

Resilient and Scalable Web Application Deployment in AWS

Project Description: -

- This project involves designing and implementing a highly available and scalable web application infrastructure on AWS.
- The architecture leverages AWS services to ensure **Fault tolerance, Load balancing, Secure user access**.
- The core of the project is to deploy a web application that can handle varying loads efficiently and maintain high availability across multiple Availability Zone (AZs).

Objectives: -

1. **High Availability** – Achieve minimal downtime for the web application by utilizing multiple Availability.
2. **Scalability** – Use AWS Auto Scaling to adjust resources automatically in response to traffic changes, ensuring efficient performance.
3. **Security** – Implement security measures focusing on security groups and secure communication.
4. **Resilience** – Develop a resilient application setup that can withstand failures and traffic spikes without manual intervention.

Core AWS Service Utilization: -

- **Virtual Private Cloud (VPC)** – Set up a custom VPC to provide an isolated network environment. This VPC will have public and private subnets across different AZs for enhanced security and availability.
- **Elastic File System (EFS)** – Leverage EFS for scalable file storage, which can be concurrently accessed by instances for storing shared application data.
- **Elastic Compute Cloud (EC2)** – Utilize EC2 instances to host the web application.
- These instances will serve as the compute resources running the application, benefiting from AWS's secure, resizable compute capacity.
- **AWS Auto Scaling** – Configure Auto Scaling to dynamically adjust the number of EC2 instances, ensuring that the application scales efficiently with demand.
- **Application Load Balancer (ALB)** – Utilize an ALB to distribute incoming traffic across multiple EC2 instances in different AZs, enhancing the fault tolerance and availability of the application.

- **Route 53** – Employ Route 53 for domain management and to route end-user requests to the application in a reliable and cost-effective manner.

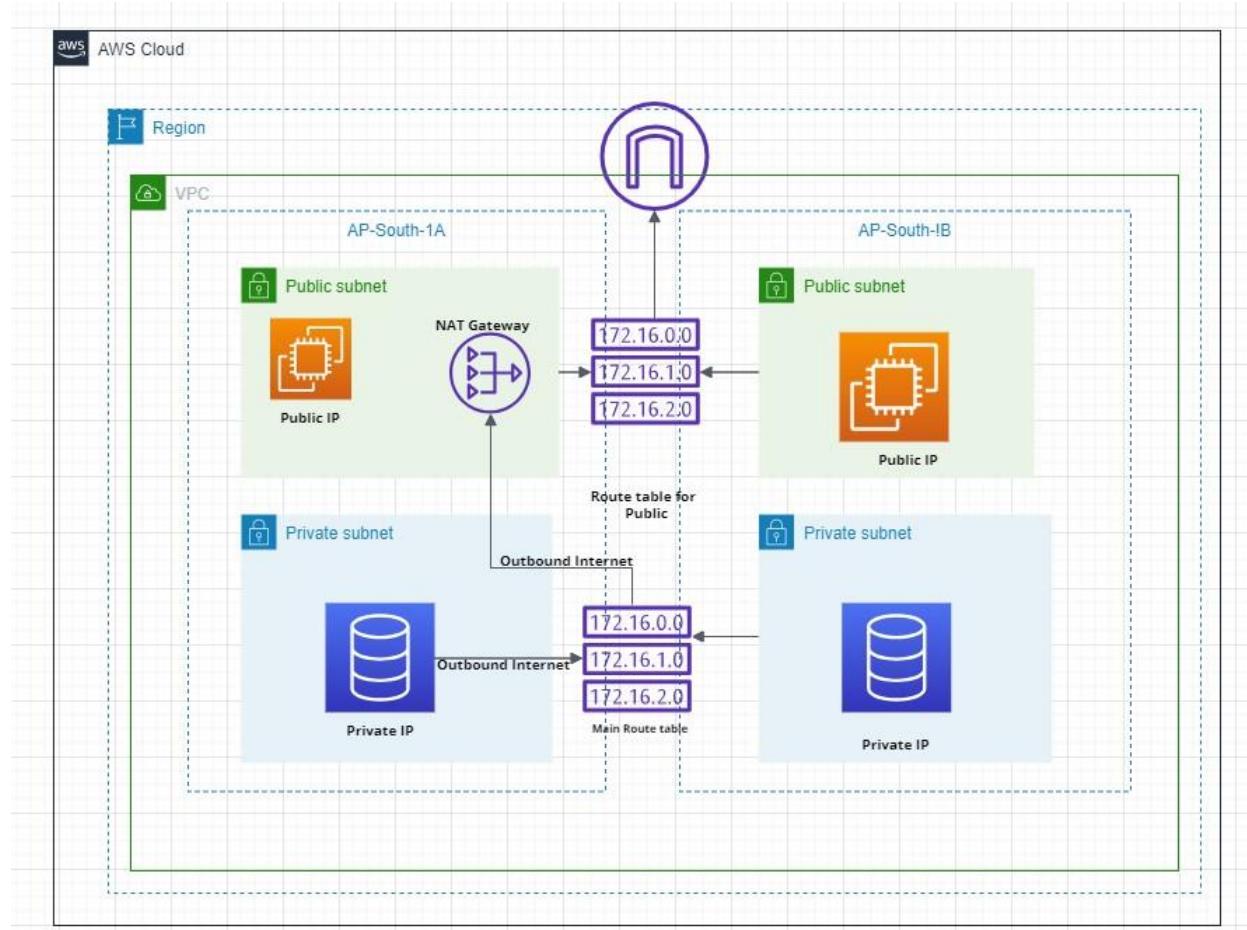
Project phases

1. **Design Phase** – Architect the solution, focusing on the application's **Security, Scalability, Availability requirements.**
2. **Implementation Phase** –
 - a. Create the VPC, subnets, and security groups.
 - b. Configure EFS.
 - c. Setup Custom AMI For Auto Scaling.
 - d. Set up and test Auto Scaling.
 - e. Deploy the ALB.
 - f. Integrate Route 53 for domain management.
3. **Testing and Optimization Phase** – Conduct functional and load testing to ensure the application's performance and scalability meet requirements.
4. **Documentation Phase** – Produce detailed documentation covering the architecture, configuration, and deployment process.

Deliverables –

- Architectural diagrams and design documentation.
- Implementation and configuration guide.
- Performance and optimization report.
- A comprehensive project presentation detailing the deployment strategy, encountered challenges, and solutions.

Phase 1: - Creation of VPC, Subnets and Security Groups



This AWS architecture is designed for high availability and resilience by leveraging multiple Availability Zones (AZs) within a VPC. The VPC spans two AZs (AP-South-1A and AP-South-1B), each containing both public and private subnets. Public subnets in each AZ (192.168.0.0/26 and 192.168.0.64/26) host internet-facing resources, ensuring accessibility even if one AZ fails. Private subnets in each AZ (192.168.0.128/26 and 192.168.0.192/26) house backend resources, maintaining functionality during an AZ outage. A NAT Gateway in the AP-South-1A public subnet allows secure internet access for private subnet instances, with the potential to add another NAT Gateway in AP-South-1B for enhanced resilience. Route tables direct traffic efficiently, with the main route table sending private subnet traffic through the NAT Gateway and public subnet traffic directly to the internet gateway. This architecture ensures continuous operation, minimal downtime, and robust disaster recovery capabilities.

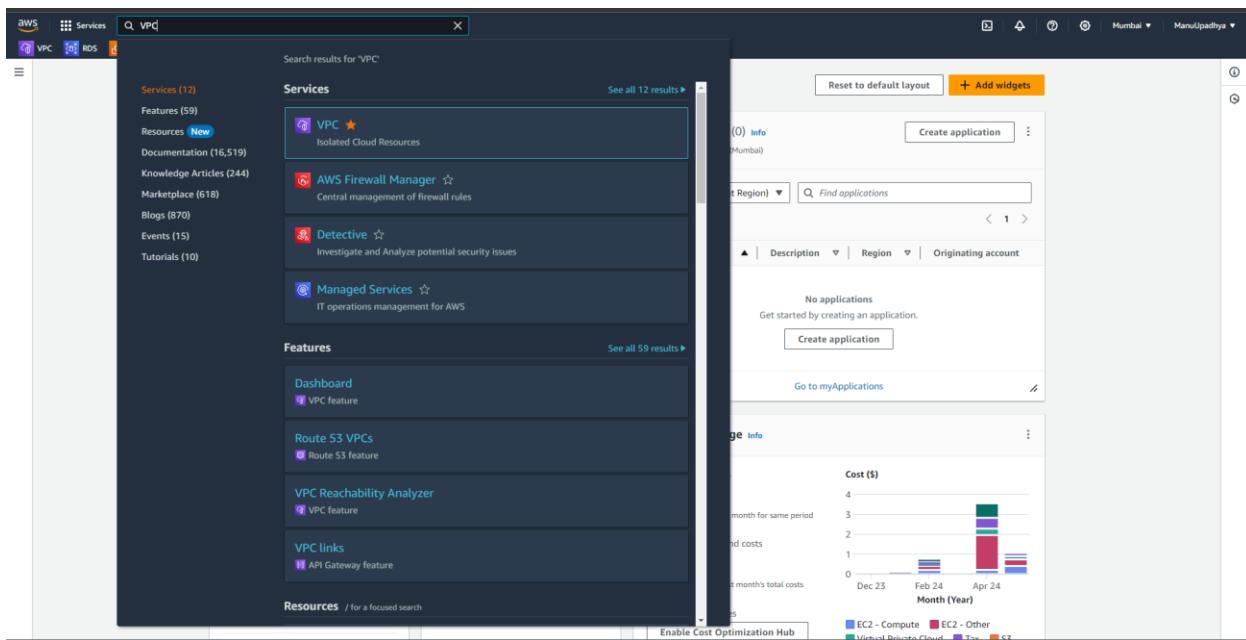
Let's break down each component in detail:

Components:

1. VPC (Virtual Private Cloud):

- The entire architecture is encapsulated within a VPC, providing isolated networking within the AWS cloud.
- The VPC has a CIDR block of 192.168.0.0/24.

Log into AWS Management Console and Navigate to VPC to Create the Virtual Private Cloud for our project.



Click on Create VPC → Provide name, IPV4 CIDR block and optional Tags.

Manoj Upadhyा

VPC > Your VPCs > Create VPC

Create VPC Info

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

VPC settings

Resources to create Info
Create only the VPC resource or the VPC and other networking resources.
 VPC only VPC and more

Name tag - optional
Creates a tag with a key of 'Name' and a value that you specify.
Capestone_Project1_VPC

IPv4 CIDR block Info
 IPv4 CIDR manual input IPAM-allocated IPv4 CIDR block
IPv4 CIDR
192.168.0.0/24
CIDR block size must be between /16 and /28.

IPv6 CIDR block Info
 No IPv6 CIDR block IPAM-allocated IPv6 CIDR block Amazon-provided IPv6 CIDR block IPv6 CIDR owned by me

Tenancy Info
Default

Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key Value - optional

CloudShell Feedback

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

Then click on create VPC.

Navigate back to VPC dashboard to check the created VPC.

aws | Services Search [Alt+S]

VPC RDS EC2 Elastic Container Service IAM CloudFront S3 Lambda

VPC dashboard Your VPCs (1/2) Info

EC2 Global View Filter by VPC: Select a 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

DNS firewall Rule groups Domain lists

Network Firewall Firewalls

CloudShell Feedback https://vpc.console.aws.amazon.com/vpc/home?region=ap-south-1

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP option set	Main route table
manoj_Default	vpc-06ab1148218fe3a8	Available	191.168.0.0/24	-	dopt-030ca0a671e63c9...	rtb-07ab5aa30040087ee
<input checked="" type="checkbox"/> Capestone_Project1_VPC	vpc-0ab54c74b6a0d76e3	Available	192.168.0.0/24	-	dopt-030ca0a671e63c9...	-

vpc-0ab54c74b6a0d76e3 / Capestone_Project1_VPC

Details Resource map CIDRs Flow logs Tags Integrations

Details

VPC ID vpc-0ab54c74b6a0d76e3	State Available	DNS hostnames Disabled	DNS resolution Enabled
Tenancy Default	DHCP option set dopt-030ca0a671e63c951	Main route table -	Main network ACL -
Default VPC No	IPv4 CIDR 192.168.0.0/24	IPv6 pool -	IPv6 CIDR (Network border group) -
Network Address Usage metrics	Route 53 Resolver DNS Firewall rule groups	Owner ID -	

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

2. Availability Zones:

- The VPC spans two Availability Zones: AP-South-1A and AP-South-1B. These zones ensure high availability and fault tolerance.

3. Subnets:

- Public Subnets:**

- AP-South-1A Public Subnet: CIDR block 192.168.0.0/26
- AP-South-1B Public Subnet: CIDR block 192.168.0.64/26.
- These subnets contain resources that need to be accessible from the internet, such as EC2 instances with public IP addresses.

○ **Private Subnets:**

- AP-South-1A Private Subnet: CIDR block 192.168.0.128/26.
- AP-South-1B Private Subnet: CIDR block 192.168.0.192/26.
- These subnets host resources that should not be directly accessible from the internet, such as databases.

Now Navigate to VPC→Subnets to create public and private subnets.

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR	Available IPv4 addresses
Public_Subnet1	subnet-0392154bc50eebab3	Available	vpc-06ab1148218fef3a8 man...	192.168.6.0/26	-	57
Private_Subnet	subnet-05dae59ca57435ac3	Available	vpc-06ab1148218fef3a8 man...	192.168.6.64/26	-	59

Click on Create Subnet.

Select the VPC ID from the dropdown.

VPC ID
Create subnets in this VPC

vpc-0ab54c74b6a0d76e3 (Capstone_Project1_VPC)
vpc-0ab54c74b6a0d76e3 (Capstone_Project1_VPC) 192.168.0/24
vpc-06ab1148218fef3a8 (manoj_Default) 192.168.6.0/24

Creation of Public Subnet 1 → Enter Subnet name (CSP_Public_Subnet1) and select the Availability Zone (Ap-South-1a) in which the subnet should be created and enter the IPv4 subnet CIDR block.

We can add optional tags if needed.

Subnet settings
Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name
Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Availability Zone [Info](#)
Choose the zone in which your subnet will reside, or let Amazon choose one for you.

IPv4 VPC CIDR block [Info](#)
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

IPv4 subnet CIDR block
 64 IPs

Tags - optional

Key	Value - optional	Remove
<input type="text" value="Name"/> <input type="button" value="X"/>	<input type="text" value="CSP_Public_Subnet1"/> <input type="button" value="X"/>	<input type="button" value="Remove"/>

You can add 49 more tags.

Click on add new subnet.

Creation of Public Subnet 2 → Enter Subnet name (CSP_Public_Subnet2) and select the Availability Zone (Ap-South-1b) in which the subnet should be created and enter the IPv4 subnet CIDR block.

Subnet 2 of 2

Subnet name

Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Availability Zone [Info](#)

Choose the zone in which your subnet will reside, or let Amazon choose one for you.



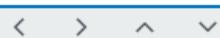
IPv4 VPC CIDR block [Info](#)

Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.



IPv4 subnet CIDR block

64 IPs



▼ Tags - optional

Key

Value - optional



Remove

Add new tag

You can add 49 more tags.

Remove

Add new subnet

Click on Add new Subnet.

Now we are going to create Private Subnets

Creation of Private Subnet 1 → Enter Subnet name (CSP_Private_Subnet1) and select the Availability Zone (Ap-South-1a) in which the subnet should be created and enter the IPv4 subnet CIDR block.

Subnet 3 of 3

Subnet name
Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Availability Zone [Info](#)
Choose the zone in which your subnet will reside, or let Amazon choose one for you.

IPv4 VPC CIDR block [Info](#)
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

IPv4 subnet CIDR block

64 IPs

< > ^ v

▼ Tags - optional

Key Value - optional [Remove](#)

[Add new tag](#)

You can add 49 more tags.

[Remove](#)

[Add new subnet](#)

Creation of Private Subnet 2 → Enter Subnet name (CSP_Private_Subnet2) and select the Availability Zone (Ap-South-1b) in which the subnet should be created and enter the IPv4 subnet CIDR block.

Subnet 4 of 4

Subnet name
Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Availability Zone [Info](#)
Choose the zone in which your subnet will reside, or let Amazon choose one for you.

IPv4 VPC CIDR block [Info](#)
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

IPv4 subnet CIDR block

64 IPs

< > ^ v

▼ Tags - optional

Key	Value - optional
<input type="text" value="Name"/>	<input type="text" value="CSP_Private_Subnet2"/>

[Add new tag](#)

You can add 49 more tags.

[Remove](#)

[Add new subnet](#)

[Cancel](#) [Create subnet](#)

Click on Create Subnet to Create all four subnets.

We can see all four subnets in Subnet Dashboard.

The screenshot shows the AWS VPC dashboard with the title "Subnets (4) Info". It displays four subnets: CSP_Private_Subnet2, CSP_Public_Subnet1, CSP_Public_Subnet2, and CSP_Private_Subnet1. Each subnet is listed with its Name, Subnet ID, State (Available), VPC, IPv4 CIDR, IPv6 CIDR, and Available IPv4 addresses. The subnets are associated with different VPCs and have specific IP ranges.

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR	Available IPv4 addresses
CSP_Private_Subnet2	subnet-01218775b95996818	Available	vpc-0ab54c74b6a0d76e3 Cap...	192.168.0.192/26	-	59
CSP_Public_Subnet1	subnet-0155158250e2c	Available	vpc-0ab54c74b6a0d76e3 Cap...	192.168.0.0/26	-	59
CSP_Public_Subnet2	subnet-09a242a3f7b3ceac	Available	vpc-0ab54c74b6a0d76e3 Cap...	192.168.0.64/26	-	59
CSP_Private_Subnet1	subnet-076f9954aa1fee063	Available	vpc-0ab54c74b6a0d76e3 Cap...	192.168.0.128/26	-	59

4. Internet Gateway:

Allows communication between instances in the VPC and the internet.

Associated with the VPC to enable direct internet access for resources in public subnets.

Navigate to Internet gateway dashboard from VPC.

The screenshot shows the AWS VPC dashboard with the title "Internet gateways (1) Info". It displays one internet gateway named "default". The gateway is attached to a specific VPC and has an owner ID of 987176295356.

Name	Internet gateway ID	State	VPC ID	Owner
default	igw-01479aa2a8bd11f98	Attached	vpc-06ab1148218feff2a8 manoj_Default	987176295356

Click on create internet gateway → give name (CSP_IGW), add an optional tag if needed.

VPC > Internet gateways > Create internet gateway

Create internet gateway Info

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

Internet gateway settings	
Name tag Creates a tag with a key of 'Name' and a value that you specify. <input type="text" value="CSP_IGW"/>	
Tags - optional A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.	
Key <input type="text" value="Name"/> <input type="button" value="X"/>	Value - optional <input type="text" value="CSP_IGW"/> <input type="button" value="X"/> <input type="button" value="Remove"/>
<input type="button" value="Add new tag"/> You can add 49 more tags.	
<input type="button" value="Cancel"/> <input type="button" value="Create internet gateway"/>	

Click on create internet gateway.

We need to attach the created Internet Gateway to VPC.

VPC dashboard						Actions ▾	
Internet gateways (1/2) <small>Info</small>						Create internet gateway	
<input type="text" value="Search"/>						<input type="button" value="View details"/> <input type="button" value="Attach to VPC"/> <input type="button" value="Detach from VPC"/> <input type="button" value="Manage tags"/> <input type="button" value="Delete internet gateway"/>	
<input type="checkbox"/>	Name	igw-01479aa2a8bd11f98	Attached	vpc-06ab1148218fef3a8	manoj_Default	987176295356	<input type="button" value="Actions ▾"/>
<input checked="" type="checkbox"/>	CSP_IGW	igw-00ad12c9a34a83047	Detached	-	987176295356	<input type="button" value="Actions ▾"/>	<input type="button" value="Create internet gateway"/>

Select the project VPC.

VPC > Internet gateways > Attach to VPC (igw-00ad12c9a34a83047)

Attach to VPC (igw-00ad12c9a34a83047) Info

VPC
Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach below.

Available VPCs	
Attach the internet gateway to this VPC: <input type="text" value="vpc-0ab54c74b6a0d76e3"/> <input type="button" value="X"/>	
<input type="button" value="AWS Command Line Interface command"/>	
<input type="button" value="Cancel"/> <input type="button" value="Attach internet gateway"/>	

Click on the attach internet gateway.

The screenshot shows the AWS VPC Dashboard. In the center, there is a detailed view of an Internet gateway named 'igw-00ad12c9a34a83047'. The 'Details' tab is selected, showing the following information:

- Internet gateway ID:** igw-00ad12c9a34a83047
- State:** Attached
- VPC ID:** vpc-0ab54c74b6a0d76e3 | Capstone Project1_VPC
- Owner:** 987176295356

Below the details, there is a 'Tags' section with a search bar and a table for key-value pairs. One tag is listed: Name: CSP_NGW.

5. NAT Gateway:

- Located in the AP-South-1A Public Subnet.
- Provides outbound internet access to instances in the private subnets while keeping them secure from incoming internet traffic.
- It has a public IP address assigned.

Navigate to NAT gateway from VPC Dashboard. Click on create NAT Gateway.

Enter name (CSP_NATGW) and select the Public Subnet, and click on “allocate Elastic IP” and add optional tag if needed.

The screenshot shows the 'Create NAT gateway' wizard. The 'NAT gateway settings' step is active. The configuration includes:

- Name:** CSP_NATGW
- Subnet:** subnet-057c0155158250e2c (CSP_Public_Subnet1)
- Connectivity type:** Public (radio button selected)
- Elastic IP allocation ID:** eipalloc-0d9e4258e7f8bdb25
- Tags:** A tag named 'Name' with value 'CSP_NATGW' is added.

Click on Create NAT Gateway.

The screenshot shows the AWS VPC Dashboard. The 'NAT gateways' section displays one entry:

Name	NAT gateway ID	Connectivity...	State	Primary public I...	Primary private I...	Primary network...	VPC
CSp_NATGW	nat-0fd5718ea56b52af	Public	Pending	-	-	192.168.0.19	eni-0658474f25cf90... vpc-0ab54c74b6a0d

We can see created NAT Gateway in the NAT Gateway Dashboard. The status of gateway will change from pending to available in 4-5 minutes.

6. Route Tables:

- **Main Route Table (for Private Subnets):**
 - Routes traffic destined for 0.0.0.0/0 (internet) to the NAT Gateway.
 - Ensures that instances in private subnets can access the internet indirectly.
- **Route Table for Public Subnets:**

Directs traffic destined for 0.0.0.0/0 to the internet gateway.

Allows instances in the public subnets to have direct access to and from the internet.

Create two route table MainRT_CSP for Private Subnets and RT_Public_CSP for Public Subnets.

Associate Public Subnets to RT_Public_CSP Route table and add route (0.0.0.0/0) via Internet gateway. This means that all network traffic (to and from) internet to subnet route via Internet gateway.

The screenshot shows the AWS VPC Route Tables page. The route table ID is rtb-023b3d18d3fbf5338 / RT_Public_CSP. The 'Subnet associations' tab is selected, showing two explicit subnet associations:

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
CSP_Public_Subnet1	subnet-057c0155158250e2c	192.168.0.0/26	-
CSP_Public_Subnet2	subnet-09a242a3f7b3ceac	192.168.0.64/26	-

Below this, there is a section for 'Subnets without explicit associations' which is currently empty.

Route table ID: rtb-023b3d18d3fbf5338

Main: No

Owner ID: vpc-0ab54c74b6a0d76e3 | Capstone_Project1_VPC

Explicit subnet associations: 2 subnets

Edge associations: -

Routes (2):

Destination	Target	Status	Propagated
0.0.0.0/0	nat-00ad12c9a34a83047	Active	No
192.168.0.0/24	local	Active	No

Associate Private Subnets to MainRT_CSP Route table and add route (0.0.0.0/0) via Nat Gateway. This means that subnet can access the internet, but internet cannot access the subnet. This is the main purpose of using NAT gateway for private subnets.

Route table ID: rtb-02629ba93a8ce831f

Main: Yes

Owner ID: vpc-0ab54c74b6a0d76e3 | Capstone_Project1_VPC

Explicit subnet associations (2):

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
CSP_Private_Subnet2	subnet-01218775b95996818	192.168.0.192/26	-
CSP_Private_Subnet1	subnet-076f954aa1fee063	192.168.0.128/26	-

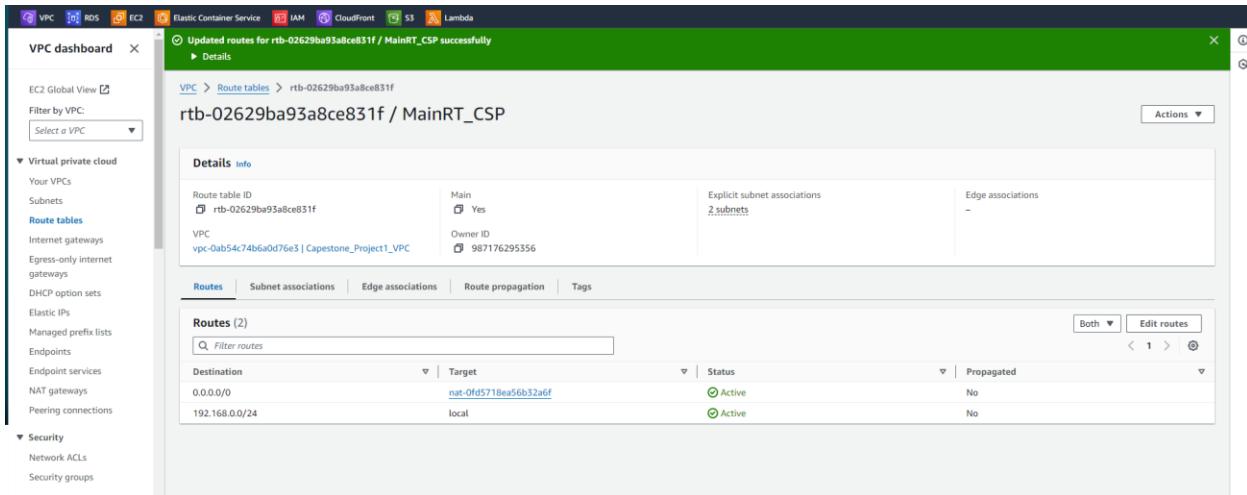
Subnets without explicit associations (0):

The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
------	-----------	-----------	-----------

No subnets without explicit associations

All your subnets are associated with a route table.



Phase 1 is almost completed we are left with security group creation we will create security group in future.

Detailed Flow of Traffic:

Outbound Internet Access for Private Subnets:

Instances in private subnets (e.g., 192.168.0.128/26 and 192.168.0.128/26) send traffic destined for the internet to the NAT Gateway via the main route table.

The NAT Gateway, located in the public subnet of AP-South-1A, translates the private IP addresses to its own public IP address and sends the traffic to the internet.

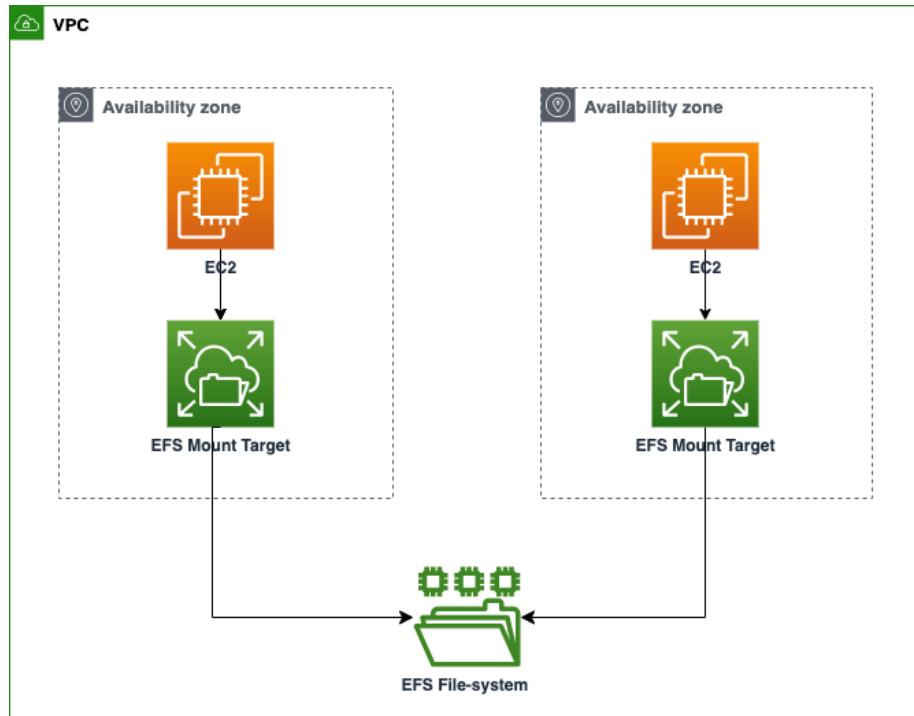
Return traffic from the internet is received by the NAT Gateway, which then translates the public IP address back to the private IP addresses of the instances and routes it accordingly.

Internet Access for Public Subnets:

Instances in public subnets (e.g., 192.168.0.0/26 and 192.168.0.64/26) route traffic destined for the internet directly through the internet gateway.

These instances can send and receive traffic directly from the internet as they have public IP addresses assigned.

Phase 2 Configure EFS



Creating Security Group for EFS

Navigate to Security group Dashboard from VPC Dashboard.

The screenshot shows the AWS VPC dashboard with the 'Security Groups' section. The 'Create security group' button is highlighted in orange at the top right of the table header.

Name	Security group ID	Security group name	VPC ID	Description	Owner
-	sg-078b1699ba5fc645	launch-wizard-1	vpc-06ab1148218feff3a8	launch-wizard-1 created 2024-05-11...	987176295356
-	sg-0d4ce3d08154406a6	default	vpc-0ab54c74b5a0d76e5	default VPC security group	987176295356
-	sg-064c464948b383949	security-group-for-inbound-nfs-d-dfl5...	vpc-06ab1148218feff3a8	[DO NOT DELETE] Security Group that...	987176295356
default	sg-08f573f41e0d5bb31	default	vpc-06ab1148218feff3a8	default VPC security group	987176295356
-	sg-0458d44e5e2ad99c	security-group-for-outbound-nfs-d-d...	vpc-06ab1148218feff3a8	[DO NOT DELETE] Security Group that...	987176295356

Click on Create Security group.

Give name, select VPC, select all traffic allowed inbound rule for as of now we will change this in future.

Created Security Group.

The screenshot shows the AWS VPC dashboard with the 'Security Groups' section selected. A success message at the top states: 'Security group (sg-0448bc35bc6043b8d | CSP_EFS_SG) was created successfully'. The main view displays the details of the new security group, including its name (sg-0448bc35bc6043b8d - CSP_EFS_SG), security group ID (sg-0448bc35bc6043b8d), owner (987176295356), and a single inbound rule allowing all traffic from 0.0.0.0/0. The 'Inbound rules' tab is active.

Creating EFS

Navigate to EFS from Amazon Management Console.

The screenshot shows the 'File systems' page in the AWS EFS console. The table header includes columns for Name, File system ID, Encrypted, Total size, Size in Standard, Size in IA, Size in Archive, Provisioned Throughput, File system state, Creation time, and Availability Zone. A large orange 'Create file system' button is centered at the bottom of the page.

Click on Create File System.

Give the name for the file system and select the VPC. Then click on Customize.

File system settings

General

Name - optional
Name your file system.
`CSP_EFS_storage`

File system type
Choose to either store data across multiple Availability Zones or within a single Availability Zone. [Learn more](#)

Regional
Offers the highest levels of availability and durability by storing file system data across multiple Availability Zones within an AWS Region.

One Zone
Provides continuous availability to data within a single Availability Zone within an AWS Region.

Automatic backups
Automatically backup your file system data with AWS Backup using recommended settings. Additional pricing applies. [Learn more](#)

Enable automatic backups

⚠️ We recommend that you create a backup policy for your file system

Lifecycle management
Automatically save money as access patterns change by moving files into the Infrequent Access (IA) or Archive storage class. [Learn more](#)

Transition into Infrequent Access (IA)	Transition into Archive	Transition into Standard
Transition files to IA based on the time since they were last accessed in Standard storage.	Transition files to Archive based on the time since they were last accessed in Standard storage.	Transition files back to Standard storage based on when they are first accessed in IA or Archive storage.
30 day(s) since last access	90 day(s) since last access	None

Encryption
Choose to enable encryption of your file system's data at rest. Uses the AWS KMS service key (`aws/elasticfilesystem`) by default. [Learn more](#)

Enable encryption of data at rest

Next step is to Mount Targets. We will select the subnet where we want to place our file system, and select the created EFS security group.

Network access

Network

Virtual Private Cloud (VPC) [Learn more](#)
Choose the VPC where you want EC2 instances to connect to your file system.
`vpc-0ab54c74b6b0d076e3 Capstone_Project1_VPC`

Mount targets
A mount target provides an NFSv4 endpoint at which you can mount an Amazon EFS file system. We recommend creating one mount target per Availability Zone. [Learn more](#)

Availability zone	Subnet ID	IP address	Security groups
ap-south-1a	subnet-076f9954aa1fee063	Automatic	<input type="button"/> Choose security groups <code>sg-0448bc35bc6043b8d CSP_EFS_SG</code>
ap-south-1b	subnet-01fc2b397074168da	Automatic	<input type="button"/> Choose security groups <code>sg-0448bc35bc6043b8d CSP_EFS_SG</code>

Add mount target

Click on next, next then create.

It will take some time to create a file system and get status available.

To check whether the EFS was created correctly and mounted properly we will navigate to EFS, by clicking on file system name in the dashboard, the go to network tab we need to have mount target state as **available** which indicates target mounted correctly.

The screenshot shows the AWS EFS console for the file system **CSP_EFS_storage (fs-09f95ea7938dfe9ae)**. The left sidebar includes links for File systems, Access points, AWS Backup, AWS DataSync, AWS Transfer, and Documentation. The main panel has tabs for General, Network, and Replication. The General tab displays settings like Performance mode (General Purpose), Throughput mode (Elastic), Lifecycle management (Transition to Infrequent Access (IA) 30 day(s) since last access, Transition to Archive 90 day(s) since last access, Transition into Standard: None), Availability zone (Regional), and DNS name (fs-09f95ea7938dfe9ae.efs.ap-south-1.amazonaws.com). The Network tab shows two mount targets: one in ap-south-1a (subnet: subnet-076f9954aa1fee063) and one in ap-south-1b (subnet: subnet-01fc2b397074168da), both marked as Available with IP addresses 192.168.0.161 and 192.168.0.253 respectively, connected to network interface IDs eni-0203bdcaa233dd104f and eni-057c83ba87cf6ab1c, and security groups sg-0448bc35bc6043b8d (CSP_EFS_SG).

Availability zone	Mount target ID	Subnet ID	Mount target state	IP address	Network interface ID	Security groups
ap-south-1a	fsmr-0676c3cc3c44de10c	subnet-076f9954aa1fee063	Available	192.168.0.161	eni-0203bdcaa233dd104f	sg-0448bc35bc6043b8d (CSP_EFS_SG)
ap-south-1b	fsmr-0b58bcc3b8d6242bc	subnet-01fc2b397074168da	Available	192.168.0.253	eni-057c83ba87cf6ab1c	sg-0448bc35bc6043b8d (CSP_EFS_SG)

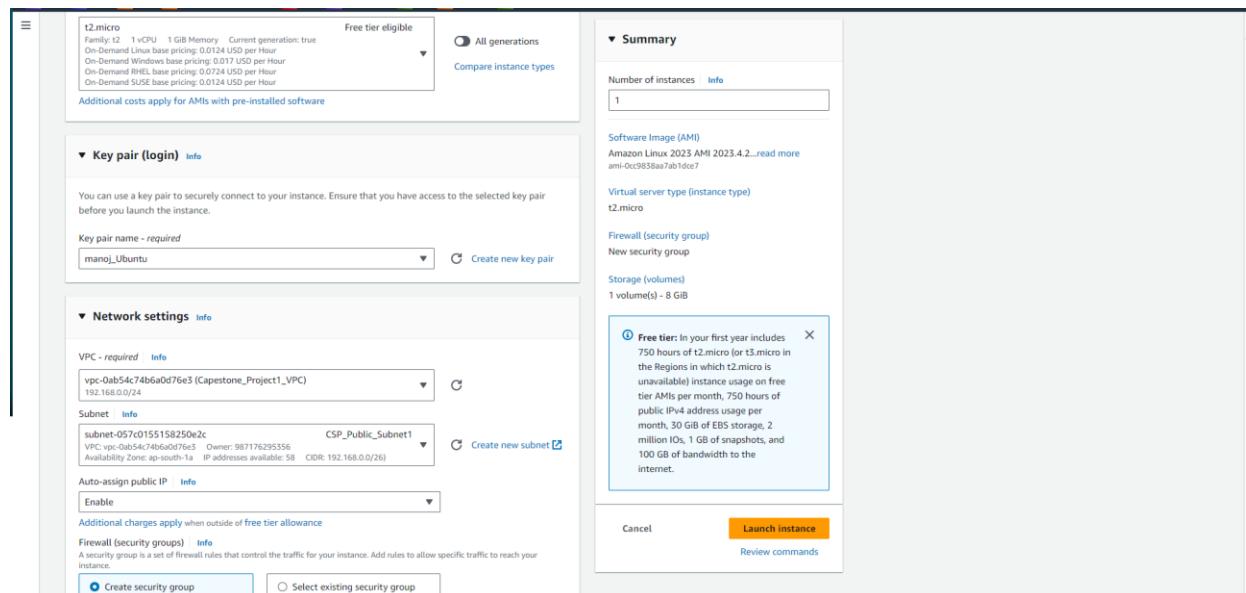
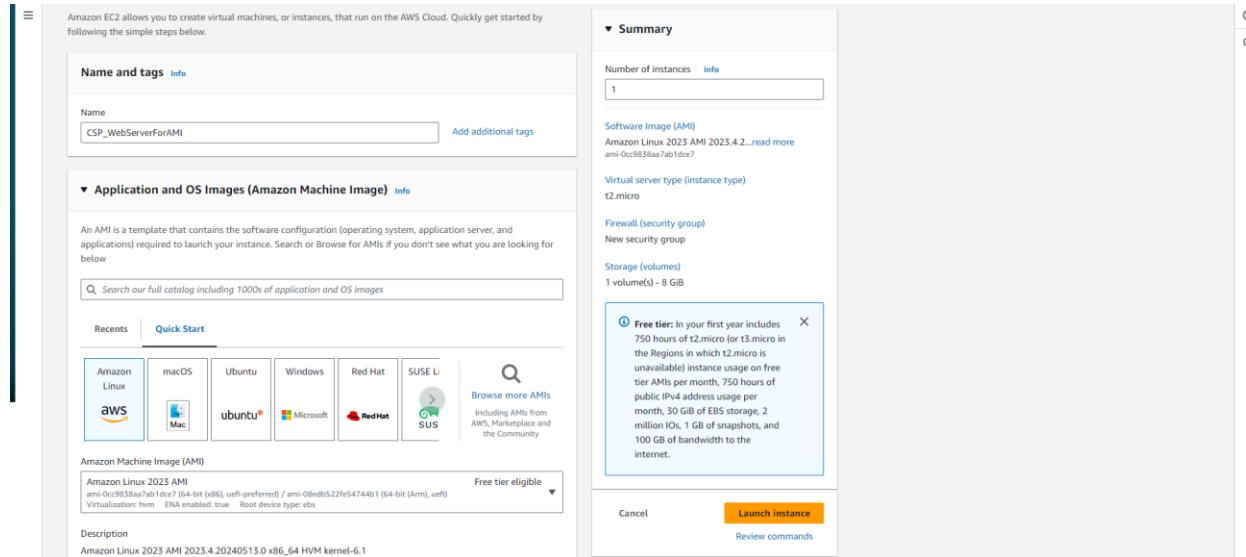
This completes our Phase 2 setup.

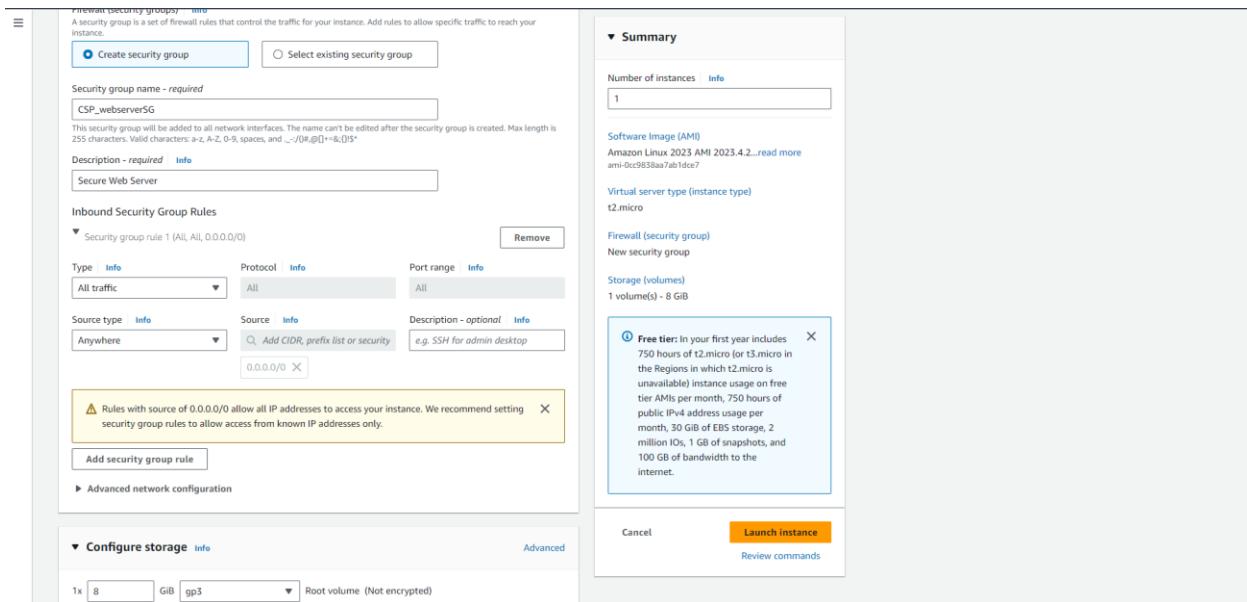
Phase 3 Setup Custom AMI for Auto Scaling

AWS Auto Scaling is a service that helps you automatically adjust the capacity of your applications running on AWS to maintain steady, predictable performance at the lowest possible cost. It involves scaling in and out (adjusting the number of instances) based on the real-time demand and predefined policies.

Before creating an AMI, we will first create an EC2 instance then from the created EC2 instance we will create an AMI.

Navigate to EC2 Dashboard, click on Launch instance.

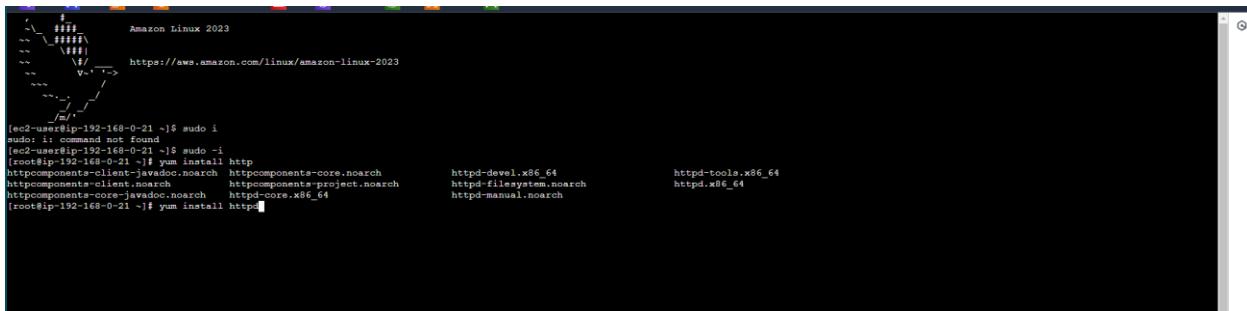




Click on Launch instance.

Log into created instance.

Installing Apache web server

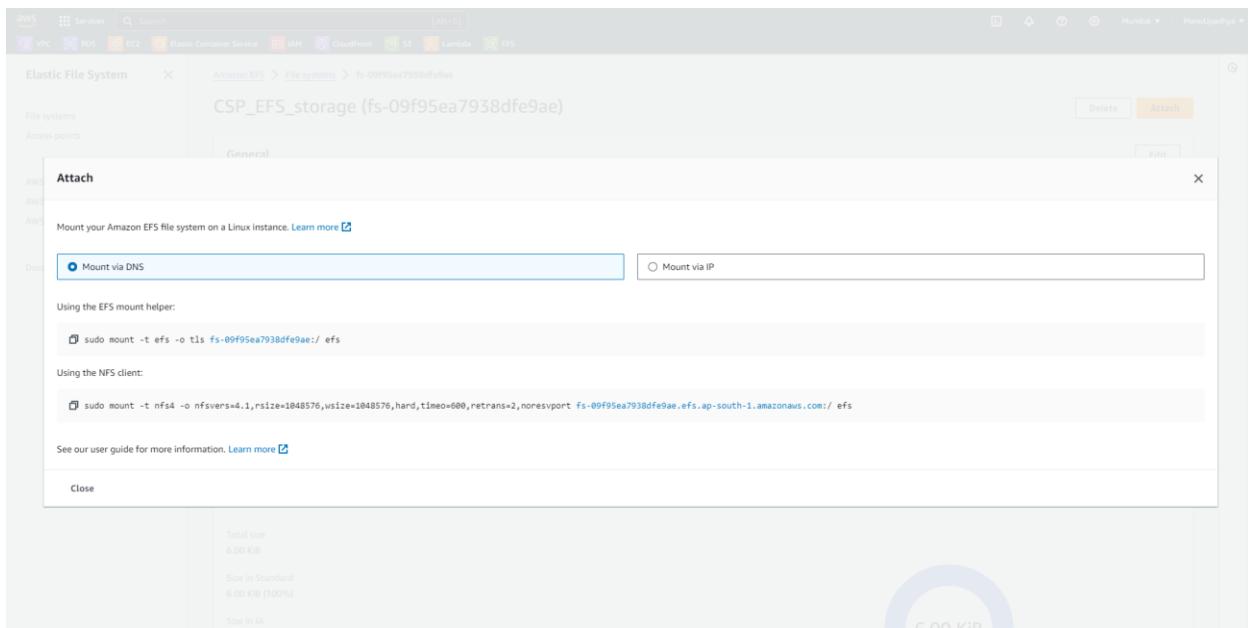


Check whether server installed correctly or not using **systemctl start httpd** command.

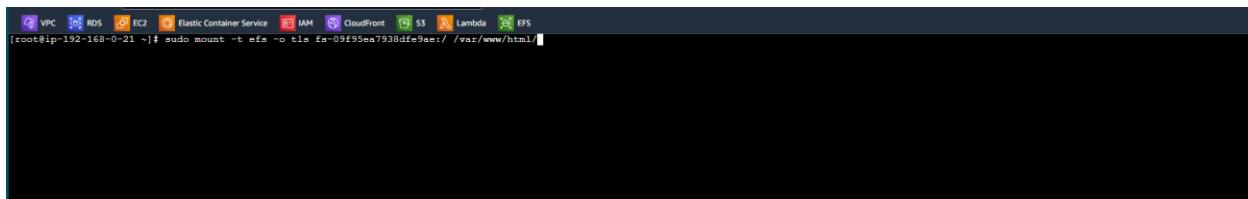
Now we will connect the created instance with created EFS file system.

Navigate to EFS → select the created file system and click on attach

Manoj Upadhyा



Now navigate to ec2 instance command line and run the command.



Now Navigate inside the /var/www/html/ and copy the web application.

```
GNU nano 5.2
[ec2-user@ip-192-168-0-21 ~]$ sudo nano index.html
index.html

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
<title>Spoon-Knife</title>
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>

```

Sample Web application created.

The last step remaining is we need to mount the EFS permanently.

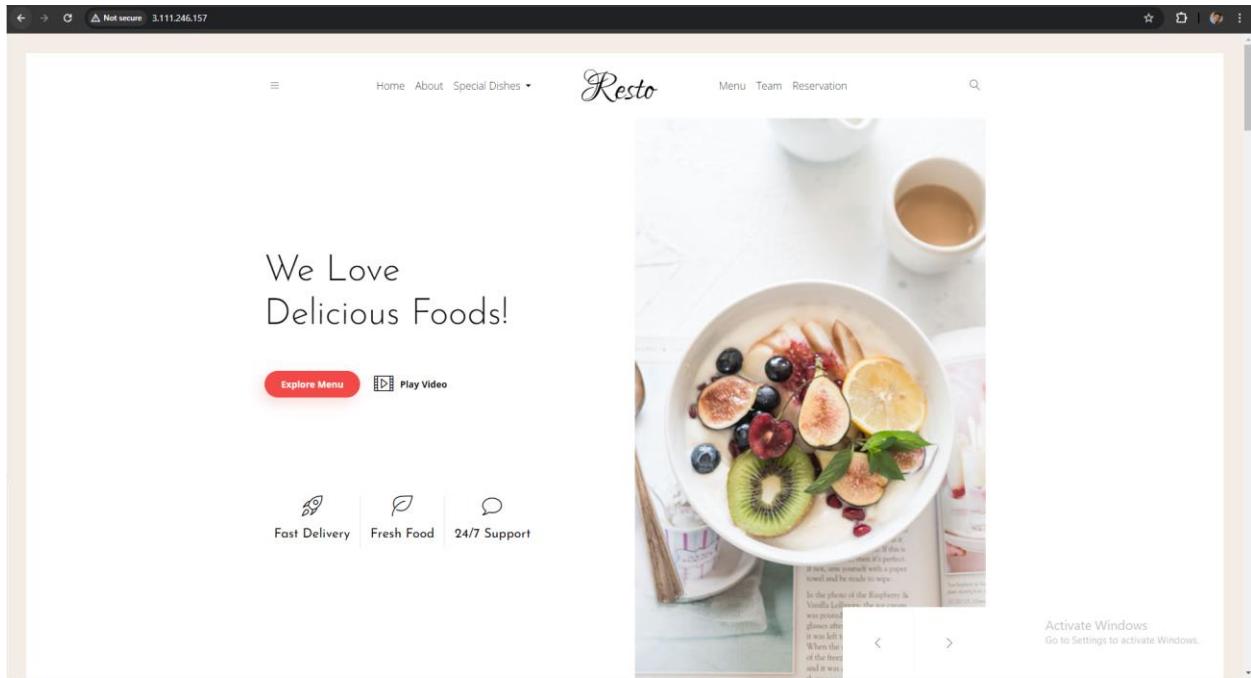
Navigate to nano /etc/fstab and add the text below into the document.

file-system-id.efs.aws-region.amazonaws.com:/ /mount/point efs defaults,_netdev 0 0

```
GNU nano 5.8                                     /etc/fstab
fs-09f95ea7938dfe9ae.efs.ap-south-1.amazonaws.com:/ /var/www/html/ efs defaults,_netdev 0 0
UUID=1600e074-aa16-449f-b780-97f8a31928c1      /          xfs    defaults,noatime 1  1
UUID=5F21-CDC5        /boot/efi      vfat   defaults,noatime,uid=0,gid=0,umask=0077,shortname=winnt,x-systemd.automount 0 2
```

Save and exit the file.

Now we need to verify our hosted web site running correctly or not.



I can open my static website using ip address. If stop and start instance, then also I am getting the website. Indicating that EFS is working correctly.

Creation of AMI from this web server: -

Now select the instance → actions → Image and templates → Create Image.

Manoj Upadhyा

The screenshot shows the AWS EC2 Instances page. A single instance, 'CSP_WebServerForAMI' (ID: i-004f327225c667c5b), is listed as 'Running'. The instance type is 't2.micro', status check is '2/2 checks passed', and it's located in 'ap-south-1a' with a public IPv4 DNS of 'ec2-3-111-246-1'. The Actions menu is open, showing options like 'Launch instances', 'Connect', 'View details', 'Manage instance state', 'Instance settings', 'Networking', 'Security', 'Image and templates' (which is highlighted in orange), 'Create image', 'Create template from instance', and 'Monitor and troubleshoot'.

The screenshot shows the 'Create image' wizard. It asks for an 'Image name' ('CSP1_AMI of Websrever') and an 'Image description - optional' ('create Image of Web Server'). Under 'Instance volumes', there is one EBS volume selected. The 'Enable' checkbox is checked. A note says 'During the image creation process, Amazon EC2 creates a snapshot of each of the above volumes.' At the bottom, there are two radio button options: 'Tag image and snapshots together' (selected) and 'Tag image and snapshots separately'.

Click on create Image.

We can see the created image in AMI Section of EC2 Dashboard.

The screenshot shows the 'Amazon Machine Images (AMIs)' page. A new AMI, 'CSP1_AMI of Websrever' (AMI ID: ami-000b4596b30261515), is listed. It was created by '987176295356/CSP1_AMI of Websrever' on '2024/05/21 11:3'. The AMI is 'Available' and has a 'Private' visibility.

Now delete the instance since it is not required for future, we create instances using created AMI.

Phase 3 is completed here.

Phase 4 Setup and test Auto Scaling

The initial step while setting up Auto Scaling is to configure the auto scaling group.

We need to create a **Launch Template** for Auto Scaling group. Click on launch template, then give name, version and select the created custom AMI.

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

Template version description
1.0
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.

Summary

- Software Image (AMI)
- Virtual server type (instance type)
- Firewall (security group)
- Storage (volumes)

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

Cancel **Create launch template**

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

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

Recents | **My AMIs** | Quick Start

Owned by me Shared with me

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

Amazon Machine Image (AMI)

CSP1_AMI of Websrever
ami-000b4596b30261515
2024-05-21T06:06:12.000Z Virtualization: hvm ENA enabled: true Root device type: ebs

Description
create Image of Web Server

Architecture x86_64 AMI ID ami-000b4596b30261515

Instance type Info | Get advice Advanced

Instance type t2.micro Free tier eligible

t2.micro Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0124 USD per Hour
On-Demand Windows base pricing: 0.017 USD per Hour
On-Demand RHEL base pricing: 0.0724 USD per Hour
On-Demand SUSE base pricing: 0.0124 USD per Hour

All generations [Compare instance types](#)

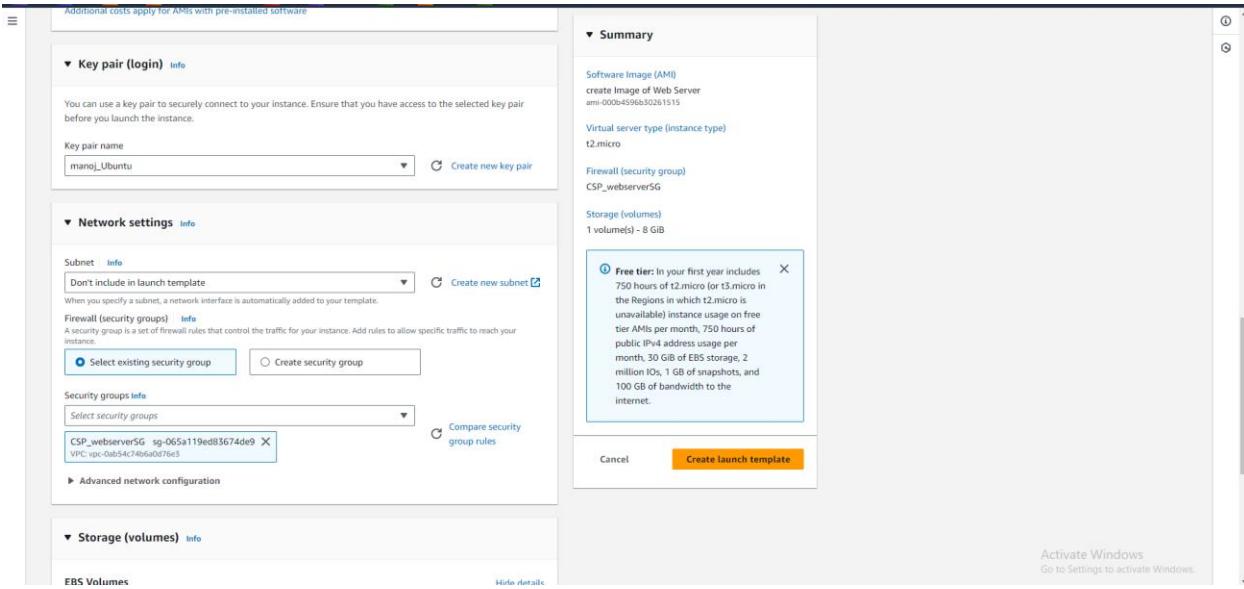
Summary

- Software Image (AMI)
create Image of Web Server ami-000b4596b30261515
- Virtual server type (instance type)
t2.micro
- Firewall (security group)
- Storage (volumes)
1 volume(s) - 8 GiB

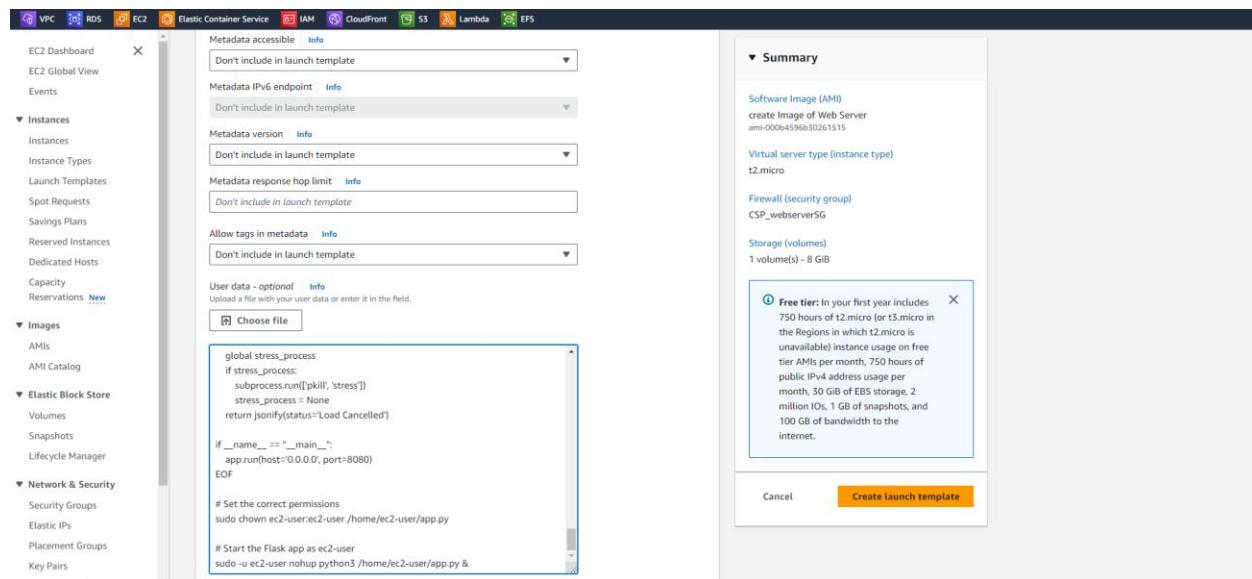
Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to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

Cancel **Create launch template**

Select instance type, key-pair, do not choose subnet while creating Launch Template, choose the created security group



Navigate to advanced details to add user data script (it is for test application used to test auto scaling).



Click on Launch template.

Now navigate back to Auto Scaling page to create Auto Scaling group.

Step 4 - optional
Configure advanced options

Step 4 - optional
Configure group size and scaling

Step 5 - optional
Add notifications

Step 6 - optional
Add tags

Step 7
Review

Name

Auto Scaling group name
Enter a name to identify the group.
CSP1_SG

Must be unique to this account in the current Region and no more than 255 characters.

Launch template info

Launch template
Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.
CSP_LT

Create a launch template [\[?\]](#)

Version
Default (1) [\[?\]](#) [\[C\]](#)

Create a launch template version [\[?\]](#)

Description
1.0

Launch template
CSP_LT [\[?\]](#)
lt-0dc07c97d82efb371

AMI ID
ami-000b4596b50261515

Key pair name
manoj_Ubuntu

Instance type
t2.micro

Security groups
-

Request Spot Instances
No

Security group IDs
sg-065a119ed83674de9 [\[?\]](#)

Additional details

Storage (volumes)
-

Date created
Thu May 23 2024 08:47:51
GMT+0530 (India Standard Time)

Give name and select created launch template, click on next.

Step 5 - optional
Add notifications

Step 6 - optional
Add tags

Step 7
Review

t2.micro

Network info

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC
Choose the VPC that defines the virtual network for your Auto Scaling group.
vpc-0ab5474b6a0d76e3 (Capestone_Project1_V... [\[?\]](#) [\[C\]](#)

Create a VPC [\[?\]](#)

Availability Zones and subnets
Define which Availability Zones and subnets your Auto Scaling group can use in the chosen VPC.

Select Availability Zones and subnets [\[?\]](#) [\[C\]](#)

ap-south-1b | subnet-01fc2b397074168da [\[?\]](#) [\[X\]](#)
(CSP_Private_Subnet2)
192.168.0.192/26

ap-south-1a | subnet-076f9954aa1feed063 [\[?\]](#) [\[X\]](#)
(CSP_Private_Subnet1)
192.168.0.128/26

Create a subnet [\[?\]](#)

Cancel Skip to review Previous **Next**

Configure network settings as required. Choose created VPC and private subnet.

Enter the Desire capacity and scaling max desired capacity.

Manoj Upadhyा

Desired capacity
Specify your group size.
0

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 0	Max desired capacity 5
Equal or less than desired capacity	
Equal or greater than desired capacity	

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.

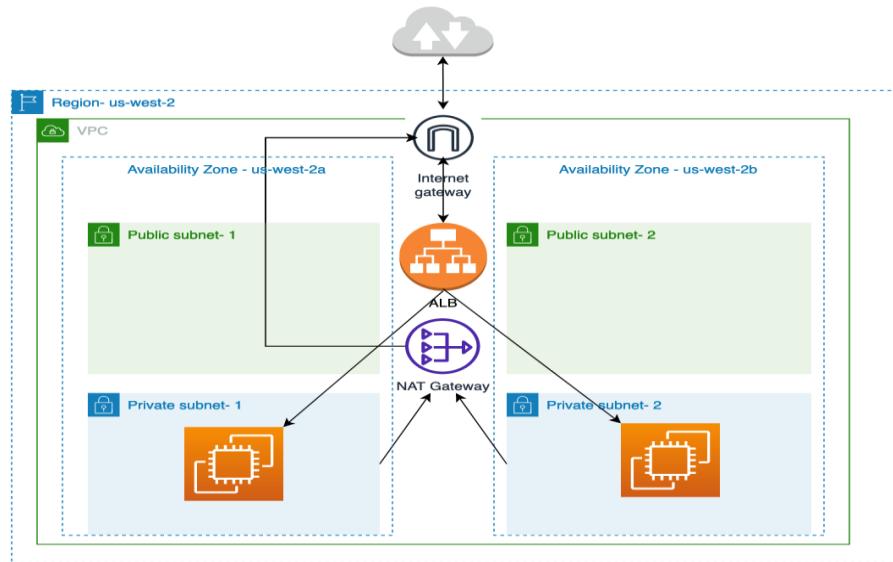
click on next→next→ next→ review and create Auto Scaling group.

Auto Scaling groups (1) Info									
<input type="text"/> Search your Auto Scaling groups									
Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones		
CSP1_SG	CSP_LT Version Default	0	-	0	0	5	ap-south-1b, ap-south-1a		

Created Auto Scaling group.

This completes the phase 4.

Phase 5 Setting Up Application Load balancer for the application.



Creation of target group for load balancer.

Navigate to AWS Management console → EC2 Dashboard → target group click on create target group.

Step 1
Specify group details

Your load balancer routes requests to the targets in a target group and performs health checks on the targets.

Step 2
Register targets

Basic configuration

Settings in this section can't be changed after the target group is created.

Choose a target type

Instances

- Supports load balancing to Instances within a specific VPC.
- Facilitates the use of Amazon EC2 Auto Scaling to manage and scale your EC2 capacity.

IP addresses

- Supports load balancing to IP 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.

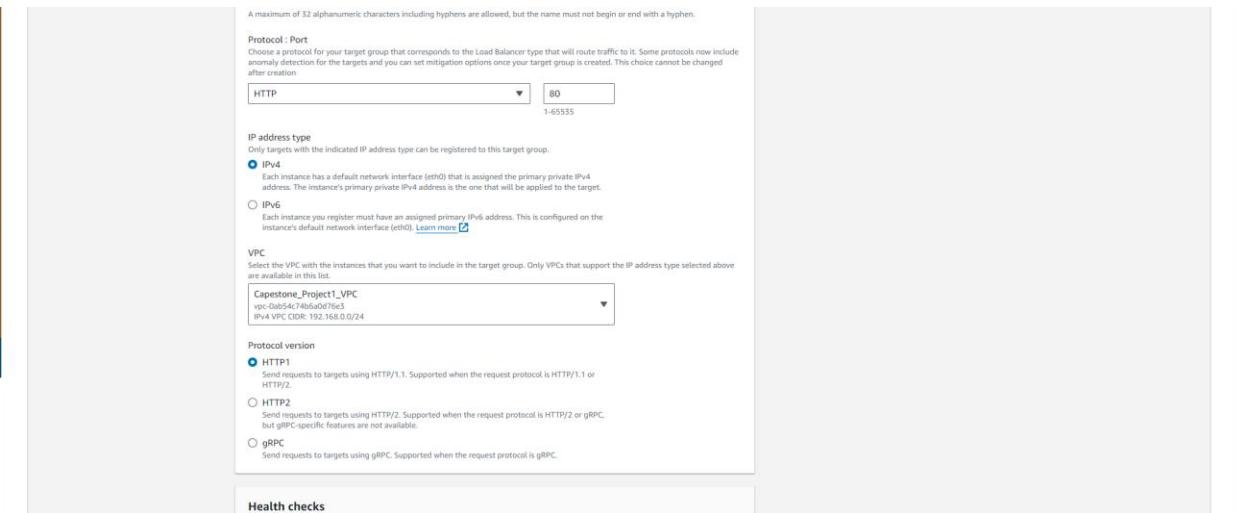
Application Load Balancer

- 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

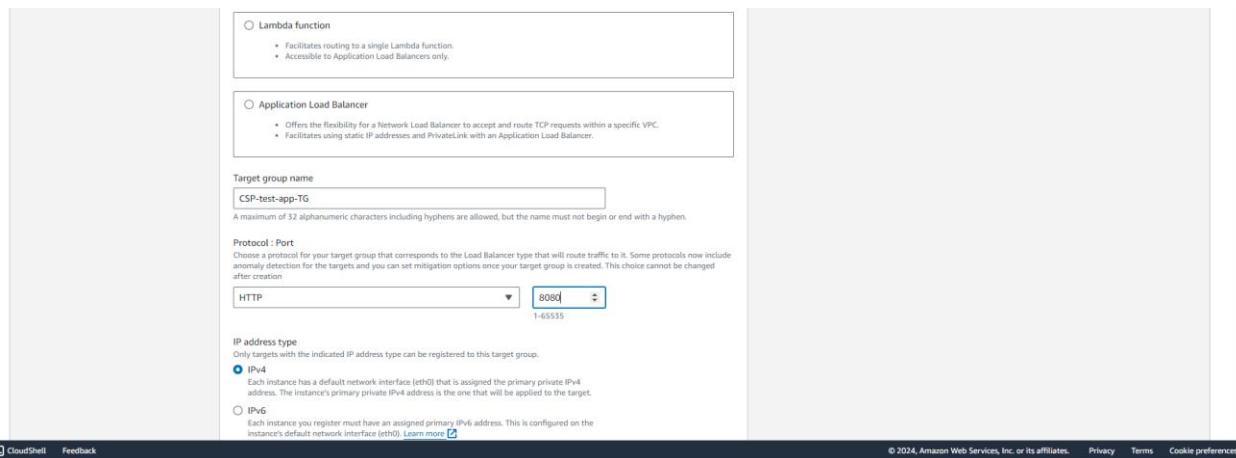
CSP_WebApp_TG

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.



Select VPC and click on create target group. Port number 80 for our application.

Now we need to create another target group for our test application.



test app will be having port 8080. Remaining steps are same as we created previous TG. Created Target groups.

The screenshot shows the AWS EC2 Target groups page. On the left, there's a navigation sidebar with sections like Instances, Images, Elastic Block Store, and Network & Security. The main area displays a table of target groups:

Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
CSP-test-app-TG	arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/CSP-test-app-TG/5678901234567890	8080	HTTP	Instance	None associated	vpc-0ab54c74b6a0d76e5
CSP-WebApp-TG	arn:aws:elasticloadbalancing:us-east-1:123456789012:targetgroup/CSP-WebApp-TG/5678901234567890	80	HTTP	Instance	None associated	vpc-0ab54c74b6a0d76e5

Below the table, it says "0 target groups selected".

Now we will Create the Application Load Balancer.

The screenshot shows the "Create Application Load Balancer" wizard, Step 1: Basic configuration. It includes sections for "How Application Load Balancers work", "Basic configuration", and "Network mapping".

Basic configuration:

- Load balancer name:** CSP_ALB
- Scheme:** Internet-facing (selected)
- IP address type:** IPv4 (selected)

Network mapping:

- VPC:** Capstone_Project1_VPC (selected)
- Mappings:**
 - ap-south-1a (aps1-az1): Subnet subnet-057c0155158250e2c (CSP_Public_Subnet1)
 - ap-south-1b (aps1-az3): Subnet subnet-09a242a3f7b3ceec (CSP_Public_Subnet2)

Manoj Upadhyा

We have not created a Security Group for Load Balancer so we will create security group now.

Choose created security group and add listener (created target groups)

The screenshot shows the 'Listeners and routing' configuration for a Load Balancer. It displays two listeners: 'Listener HTTP:80' and 'Listener HTTP:8080'. Each listener has a 'Protocol' dropdown set to 'HTTP', a 'Port' input field, and a 'Default action' section. The 'Default action' for both listeners is 'Forward to' a target group named 'CSP-WebApp-TG' (Target type: Instance, IPv4). Below each listener, there is a 'Create target group' button. A note below the first listener states: 'A listener is a process that checks for connection requests using the port and protocol you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets.' Another note below the second listener states: 'Listener tags - optional. Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.' Both sections include 'Add listener tag' buttons and a note about adding up to 50 more tags.

Click on create load balancer.

It will take few minutes to become active.

The screenshot shows the EC2 Dashboard with the 'Load balancers' section selected. It displays a table titled 'Load balancers (1)'. The table includes columns for Name, DNS name, State, VPC ID, Availability Zones, Type, and Date created. One row is shown for 'CSP-ALB', which is listed as 'Active' with a VPC ID of 'vpc-0ab54c74b6a0d76e3' and 2 availability zones. The 'Actions' dropdown menu includes an option to 'Create load balancer'.

Now navigate to Auto Scaling group to link load balancer with Auto scaling group.

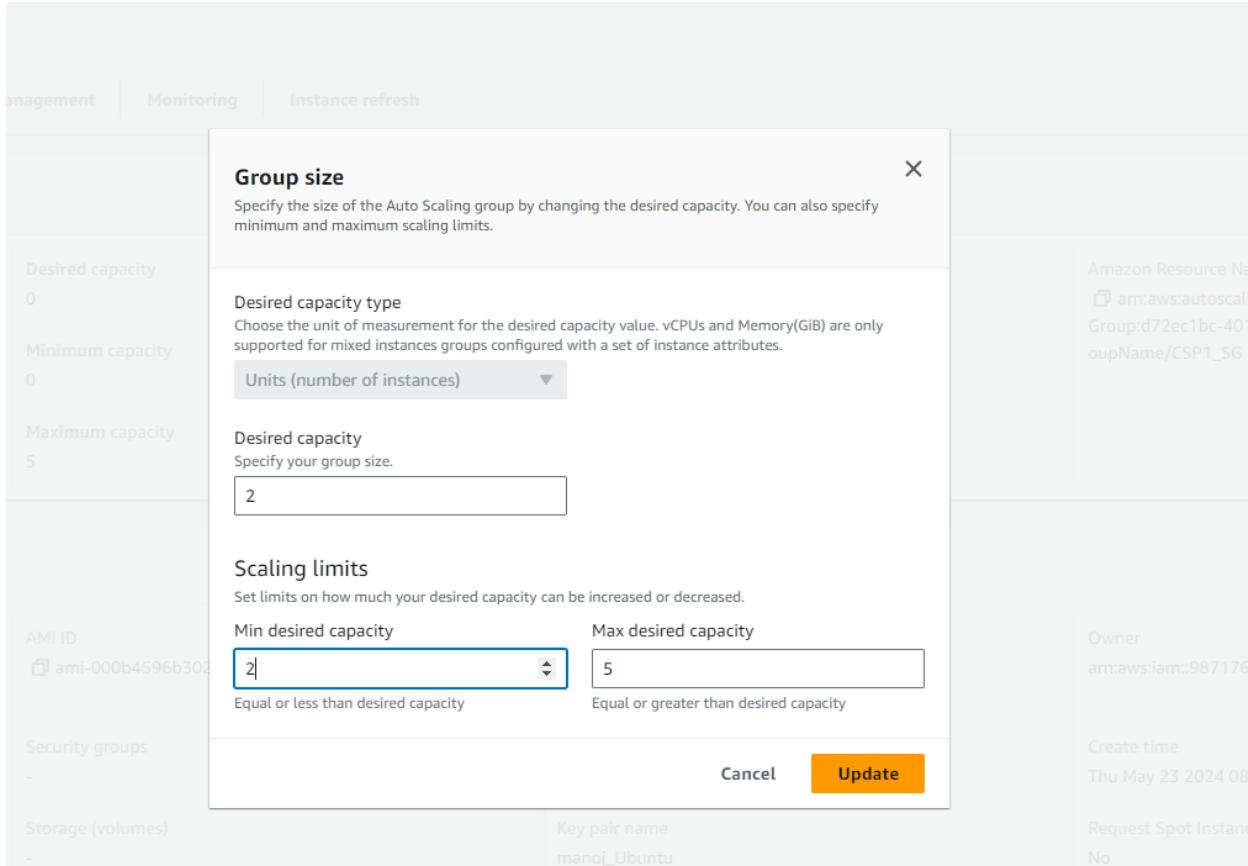
Manoj Upadhyा

click on edit of load balancing.

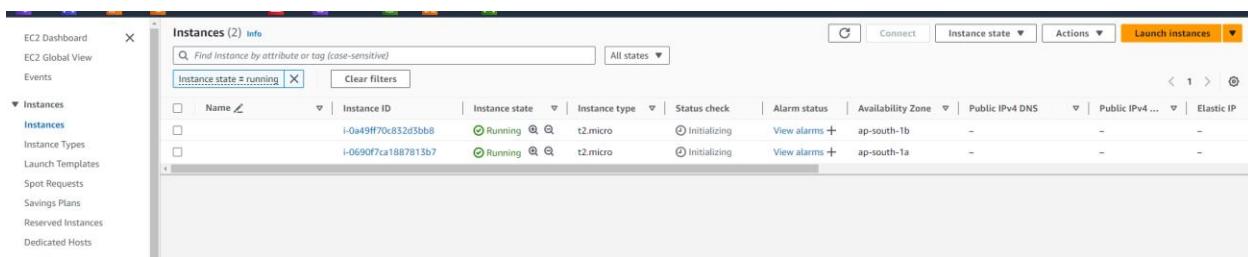
Select both target groups and click on update.

Now we will manually test.

we will manually increase the auto scaling desired capacity. Navigate to auto scaling group → Edit Group Details.



Click on Update it will create two instances. Post creation we will test the deployed application note that the Application will be deployed from AMI.



Instances are created by auto scaling group.

we will navigate to load balancing and check health of targets(launched instances/ web server)

Manoj Upadhyा

The screenshot shows the AWS EC2 Target Groups console for the 'CSP-WebApp-TG' target group. The 'Health checks' tab is selected. It lists two targets: one healthy and one unhealthy. The 'Distribution of targets by Availability Zone (AZ)' section shows traffic distribution across three AZs. The 'Health check settings' section defines an HTTP health check with a path of '/' and a 5-second timeout.

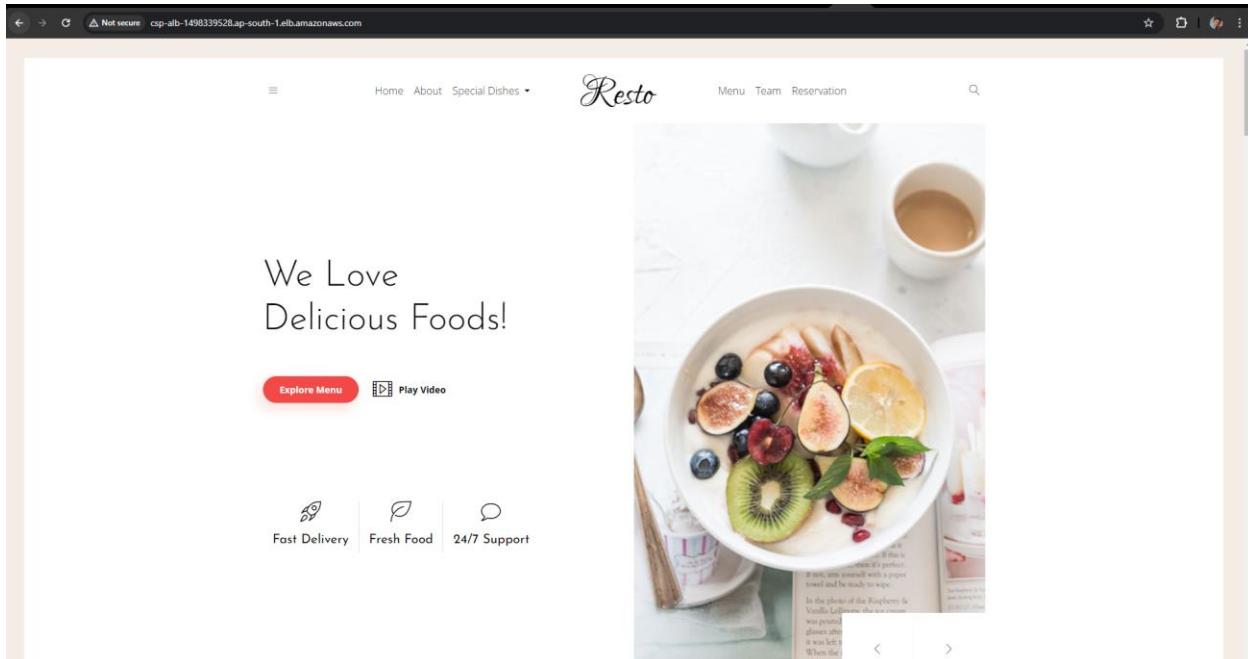
instances are healthy

Now we will copy and paste the DNs end point of load balancer in browser to get application.

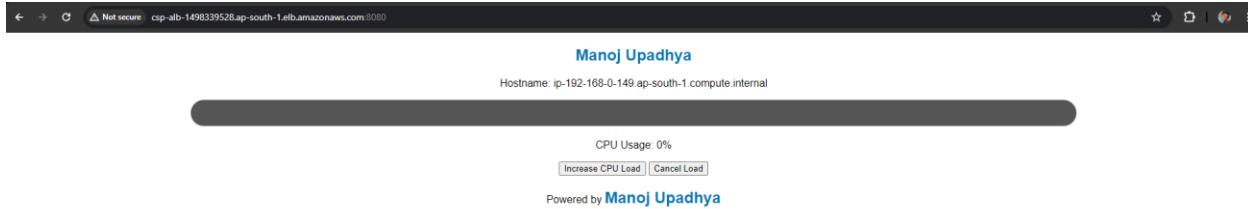
The screenshot shows the AWS Load Balancers console for the 'CSP-ALB' load balancer. The 'Listeners and rules' tab is selected, showing two listeners: 'HTTP:80' and 'HTTP:8080'. Both listeners forward traffic to the 'CSP-WebApp-TG' target group. The 'Network mapping' tab shows the association between the load balancer and the VPC.

copy the DNS Name

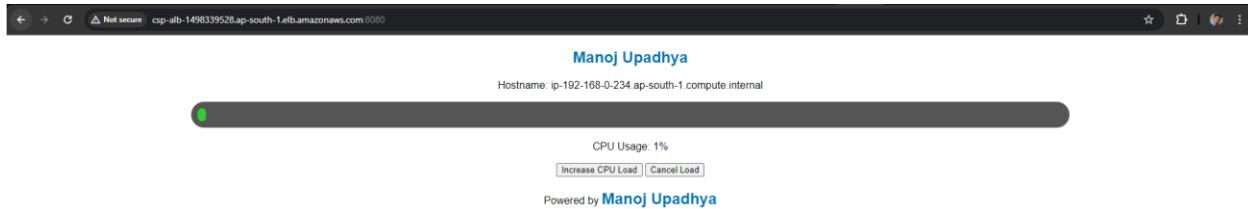
Manoj Upadhyा



Now how we will get to know the load balancer routing network traffic to which instance, here we use Test Application to check the host name ip of created EC2 instances using which we can understand to which EC2 instances the network traffic is routed.



If I refresh the page



Notice the changed host name.

Phase 5 completed here.

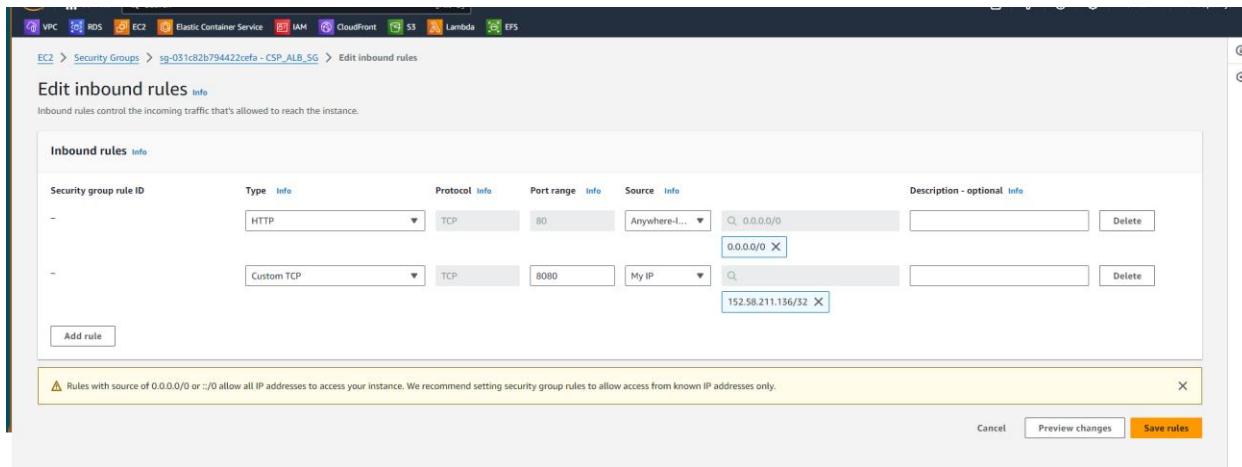
Phase 6 Setting up all Security Group according to AWS Best Practices

As you we have created 3 Security group (ALB SG, WEB SG, EFS, SG) all of them having all traffic allowed (inbound network traffics are allowed) Which is not a good practice, we will setup security group in this phase according to best practice.

ALB SG inbound rule :-

HTTP 80 from anywhere → web application ALB

HTTP 8080 only my ip → Test app Alb



click on Save.

Web SG inbound rule: -

Here we will make sure that network traffic to Web Server only comes from Application Load Balancer, there in Source section we have selected ALB SG. Added ssh rule for debugging purposes.

Manoj Upadhyा

Inbound rules info

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
-	HTTP	TCP	80	Custom sg-031c82b794422cefa	Delete
-	Custom TCP	TCP	8080	Custom sg-031c82b794422cefa	Delete
-	SSH	TCP	22	My IP 152.58.211.136/32	Delete

Add rule Cancel Preview changes Save rules

EFS SG inbound rule :-

we allow NFS traffic from Web server only.

Edit inbound rules info

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0f088dd3d3478dfb7	NFS	TCP	2049	Custom sg-065a119ed83674de9	Delete

Add rule Cancel Preview changes Save rules

No we have configured all the security group in AWS best Practices.

Now test the application if its working fine or not, if any error made application will not behave properly.

How to test?

ALB DNS name for both web application and test application.

Check the target group health.

Check accessing test app using some other device it should not open in the other device.

This completed Phase 6.

Phase 7 Integration of Route 53 for Domain Name Management.

Prerequisite for this phase is you should have your own domain name.

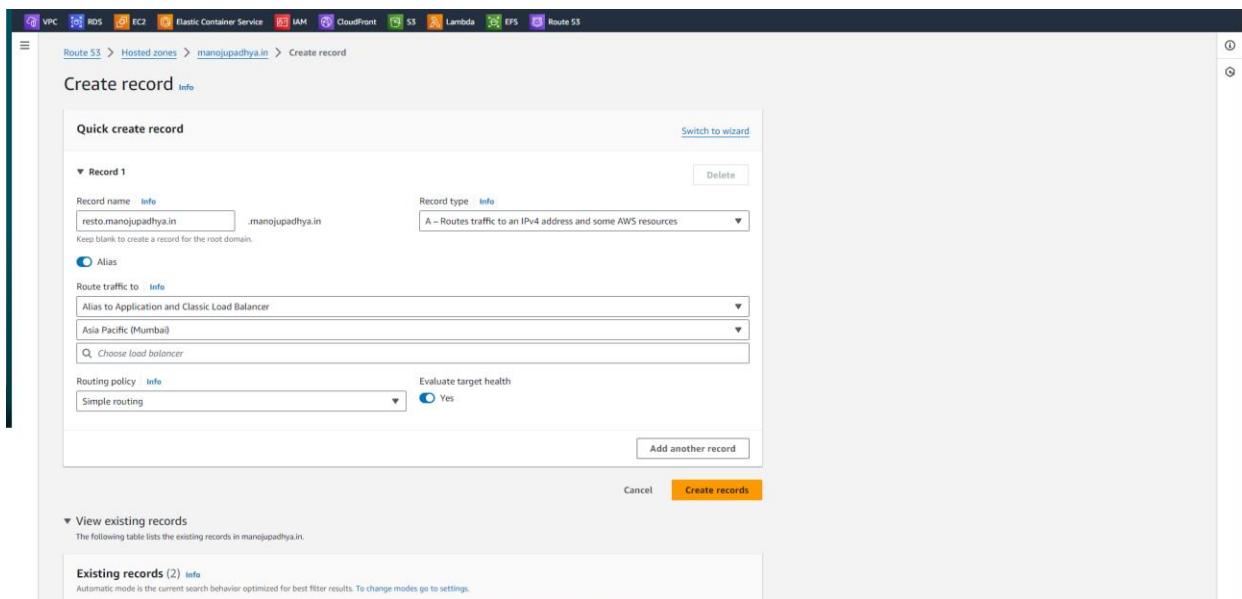
Navigate to → Route 53 in AWS Management Console.

Register a domain name, if you have purchased domain name from any other website add it to Route 53.

Create a Hosted Zone. After creating the Hosted Zone, we need to add Nameserver record present in hosted zone to our Domain name.

Now we need to create a Record for our web server purposes.

Click on create record.



In the alias section choose alias as the Load balancer so that Domain name forwards the traffic to load balancer.

It will take some time to get the record created.

Now we can access the website using DNS name directly.

Manoj Upadhyा

Route 53

Hosted zones

manojupadhyा. in

Records (3)

Record name	Type	Value/Route traffic to	TTL (s)
manojupadhyा. in	NS	Simple ns-337.awsdns-42.com. ns-1172.awsdns-18.org. ns-1948.awsdns-51.co.uk. ns-992.awsdns-60.net.	17280
manojupadhyा. in	SOA	Simple ns-537.awsdns-42.com. awsd... 900	
resto.manojupadhyा. in	A	Simple Yes dualstack.csp-alb-1498339528.ap-south-1.elb.amazonaws.com.	-

Record details

Record name: resto.manojupadhyा. in

Record type: A

Value: dualstack.csp-alb-1498339528.ap-south-1.elb.amazonaws.com.

Alias: Yes

TTL (seconds): -

Routing policy: Simple

Not secure resto.manojupadhyा. in

Home About Special Dishes Resto Menu Team Reservation

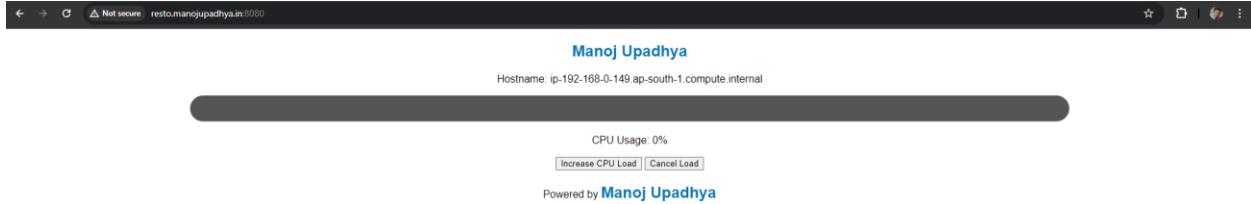
We Love Delicious Foods!

Explore Menu Play Video

Fast Delivery Fresh Food 24/7 Support



Manoj Upadhyा



We can see the test and web application using domain name.

Now we can say the Project is 100% completed. All the phases are completed.

Now we are left with Testing and Optimization Phase.

Project Functionality Testing and load Testing.

Here we use a test application for testing purposes.

- High Availability.
Navigate to EC2 Dashboard and terminate an instance. Try to access website, mean time autoscaling group should create one more instance and check these changes IP address/host name in test application as well.
- Setting up Dynamic Scaling Policy (If CPU Utilization > 60 Launch a new instance) and test using test application by increasing load.
-