

Project

CONSTRUCTING A CLOUD INFRASTRUCTURE USING AWS

2025

Prepared by
Pottipothu Divya

Table of Contents

1. Introduction:	2
2. Project Overview	2
3. Cloud Architecture Diagram	3
4. Creating a Virtual Private Cloud (VPC)	3
6.Configuring Internet Gateway for Public Access	5
8. Configuring NAT Gateway for Private Subnet Internet Access	6
9. Integrating MyNATGW with PrivateSubnet-C via Route Table	7
10.Creating NACL	8
11. Creating Amazon SNS Topic and Subscriptions	10
12.Configuring Amazon S3 for Scalable Storage	12
13.Enabling VPC Flow Logs and Storing in S3	12
14. Creating IAM Role for EC2 with S3 Full Access	13
15. Setting Up Amazon Elastic File System (MyEFS)	14
17.Creating LoadBalancer	16
19. Setting Up Auto Scaling Group (MyASG) for High Availability	19
20. Setting Up Bastion and Database Servers	20
21.Cloud watch	21
22. Cloud Trail	22
23.Checking	23
23.1 Checking WebServer	23
23.2 Checking Load Balancer	23
23.3 Checking Bastion	24
23.4 Checking DatabaseDB	25
23.5 Checking S3 files & website are accessing	27
24.Conclusion	30

1. Introduction:

In modern businesses, cloud infrastructure is essential for scalability, security, and flexibility. Most companies leverage AWS services to build secure, scalable, and cost-effective solutions. This project focuses on building a Virtual Private Cloud (VPC) in AWS with various networking, security, and storage components, ensuring efficient network segmentation and internet accessibility.

2. Project Overview

The goal of this project is to set up a cloud-based infrastructure using AWS, where businesses can securely host applications. The setup includes:

◆ Networking & Security

- **VPC (Virtual Private Cloud)** with public and private subnets
- **Internet Gateway (IG)** for external connectivity
- **NAT Gateway** to enable internet access for private subnets
- **Route Tables & Network ACLs (NACLs)** for traffic control
- **Security Groups & IAM (Identity & Access Management)** for access control

◆ Compute & Storage

- **EC2 (Elastic Compute Cloud) Instances** for scalable computing power
- **Elastic Load Balancer (ELB)** for distributing incoming traffic
- **Auto Scaling Group (ASG)** for high availability and fault tolerance
- **Amazon S3** for object storage and backups
- **Elastic File System (EFS)** for shared file storage

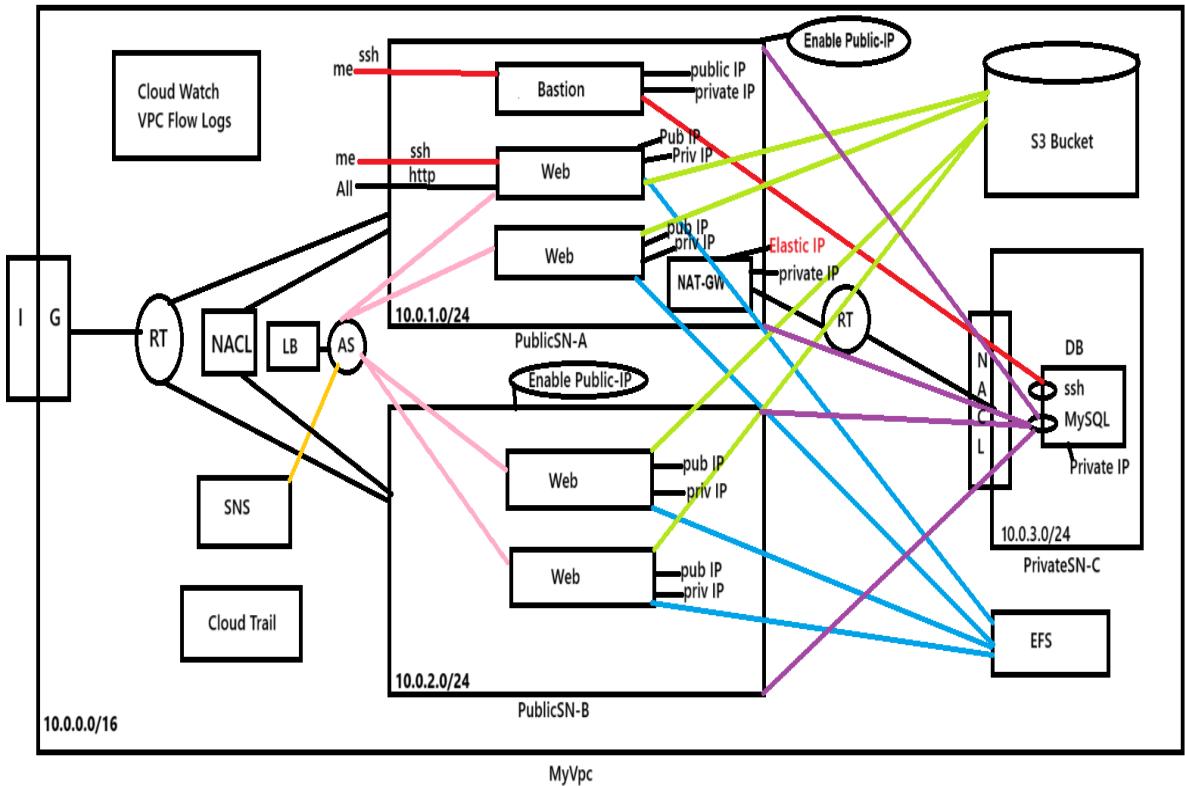
◆ Monitoring & Logging

- **Amazon CloudWatch** for real-time monitoring and alerts
- **AWS CloudTrail** for auditing API calls and activity tracking
- **VPC Flow Logs** for analyzing network traffic patterns

◆ Event-Driven & Messaging Services

- **Simple Notification Service (SNS)** for automated notifications

3. Cloud Architecture Diagram



4. Creating a Virtual Private Cloud (VPC)

A VPC (MyVpc) is created in North Virginia with a CIDR block of **10.0.0.0/16**, which provides a secure and scalable cloud network. In this setup, subnets, a route table, and an internet gateway are automatically generated, simplifying network configuration. This ensures efficient resource management, controlled communication, and enhanced security for cloud deployments.

You successfully created **vpc-04309df8fcccc9466 / MyVpc**

Your VPCs (2) Info

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR
-	vpc-009b1ff341dcb87d8	Available	Off	172.31.0.0/16	-
MyVpc	vpc-04309df8fcccc9466	Available	Off	10.0.0.0/16	-

Select a VPC above

5. Setting Up Public and Private Subnets in VPC(MyVpc)

Three subnets are created: **PublicSubnet-A (10.0.1.0/24)**, **PublicSubnet-B (10.0.2.0/24)**, and **PrivateSubnet-C (10.0.3.0/24)**. By default, all AWS subnets are private. To enable public access, **PublicSubnet-A** and **PublicSubnet-B** are configured to assign public IPs to instances. The instances launched in these subnets retain the same public IP, ensuring consistent internet connectivity, while **PrivateSubnet-C** remains isolated for internal workloads.

You have successfully created 1 subnet: **subnet-01e902484d26bd96c**

Subnets (9) Info

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
-	subnet-09f86776f7974b80f	Available	vpc-009b1ff341dcb87d8	Off	172.31.48.0/
-	subnet-056a73dd4f1025cc3	Available	vpc-009b1ff341dcb87d8	Off	172.31.80.0/
-	subnet-0efdf70e2e40703c8	Available	vpc-009b1ff341dcb87d8	Off	172.31.32.0/
PublicSubnet-A	subnet-00a50251eaace5a24	Available	vpc-04309df8fcccc9466 / MyVpc	Off	10.0.1.0/24
PublicSubnet-B	subnet-0062bd35e7aaa7c50	Available	vpc-04309df8fcccc9466 / MyVpc	Off	10.0.2.0/24
PrivateSubnet-C	subnet-01e902484d26bd96c	Available	vpc-04309df8fcccc9466 / MyVpc	Off	10.0.3.0/24

Select a subnet

6.Configuring Internet Gateway for Public Access

An **Internet Gateway (MyIGW)** is created and attached to the **VPC (MyVpc)** to enable public internet access. It allows resources in **PublicSubnet-A** and **PublicSubnet-B** to communicate with the internet while ensuring a secure network architecture. The **MyIGW** facilitates seamless inbound and outbound traffic for publicly accessible instances, enabling them to send and receive data over the internet.

The screenshot shows the AWS VPC dashboard with the 'Internet gateways' section selected. The list shows two entries: one unnamed gateway with ID `igw-0099d041829242f87` and another named `MyIGW` with ID `igw-06f67d0b709d15968`. Both are attached to the VPC `vpc-009b1ff341dcb87d8`, which is owned by user `010438484509`. The details page for `igw-06f67d0b709d15968 / MyIGW` is displayed, showing its state as `Attached` and its VPC ID as `vpc-04309df8fcccc9466 | MyVpc`.

7.Establishing Route Table Connection Between IGW and Public Subnets

The **Route Table (MyRoute)** is created and configured to enable internet access for public subnets. It is linked to the **Internet Gateway (MyIGW)** and associated with **PublicSubnet-A** and **PublicSubnet-B**. This configuration ensures that instances within these subnets can establish outbound connections to the internet while securely managing inbound traffic as required.

The screenshot shows the AWS VPC dashboard with the 'Route tables' section selected. A success message indicates that route table `rtb-0c097c175d9c36819 | MyRoute` was created successfully. The list shows three route tables: one unnamed with ID `rtb-0c01ceb82238521cb`, one unnamed with ID `rtb-07b7188bec62040fc`, and one named `MyRoute` with ID `rtb-0c097c175d9c36819`. The `MyRoute` table is associated with the Main edge and is not the default route. The details page for `rtb-0c097c175d9c36819 | MyRoute` is displayed, showing it is the Main route table.

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (2/3)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
PublicSubNet-A	subnet-00a50251eaace5a24	10.0.1.0/24	-	Main (rtb-07b7188bec62040fc)
PublicSubNet-B	subnet-0062bd35e7eaa7c50	10.0.2.0/24	-	Main (rtb-07b7188bec62040fc)
PrivateSubNet-C	subnet-01e902484d26bd96c	10.0.3.0/24	-	Main (rtb-07b7188bec62040fc)

Selected subnets

- subnet-00a50251eaace5a24 / PublicSubNet-A
- subnet-0062bd35e7eaa7c50 / PublicSubNet-B

Buttons: Cancel, Save associations

Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	No
0.0.0.0/0	Internet Gateway	-	No

Buttons: Add route, Remove, Cancel, Preview, Save changes

8. Configuring NAT Gateway for Private Subnet Internet Access

A **NAT Gateway**(MyNATGW) is created in **PublicSubnet-A** and assigned an **Elastic IP** to enable secure internet access for instances in the **private subnet**. Since private subnets do not have direct internet access, the NAT Gateway allows instances in **PrivateSubnet-C** to initiate outbound connections for tasks such as software updates, package installations, and API communications while preventing unsolicited inbound traffic.

VPC dashboard

NAT gateways (1)

Actions Create NAT gateway

Name	NAT gateway ID	Connectivity...	State	Primary public I...	Primary private I...
MyNATGW	nat-0cf0d4b7dfa3d0843	Public	Pending	-	10.0.1.54

Success message: NAT gateway nat-0cf0d4b7dfa3d0843 | MyNATGW was created successfully.

9. Integrating MyNATGW with PrivateSubnet-C via Route Table

MyNATGW is connected to **PrivateSubnet-C** through the default **route table (privateRT)**, which is automatically created during the creation of MyVPC. This configuration ensures that outbound internet traffic from PrivateSubnet-C is routed through MyNATGW while maintaining security by restricting inbound access.

The screenshot shows the AWS VPC dashboard with the 'Route tables' section. A success message at the top states: 'You have successfully updated subnet associations for rtb-07b7188bec62040fc / privateRT.' The route table list includes:

Name	Route table ID	Explicit subnet assoc...	Main	VPC
-	rtb-0c01ceb82238521cb	-	Yes	vpc-009b1ff341dcb87d8
<input checked="" type="checkbox"/> privateRT	rtb-07b7188bec62040fc	subnet-01e902484d26bd...	Yes	vpc-04309df8fcccc9466 MyVp
<input type="checkbox"/> MyRoute	rtb-0c097c175d9c36819	2 subnets	No	vpc-04309df8fcccc9466 MyVp

The screenshot shows the 'Edit subnet associations' page for route table **rtb-07b7188bec62040fc**. It lists available subnets and selected subnets.

Available subnets (1/3):

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
PublicSubNet-A	subnet-00a50251eaace5a24	10.0.1.0/24	-	rtb-0c097c175d9c36819 / MyRoute
PublicSubNet-B	subnet-0062bd35e7eaa7c50	10.0.2.0/24	-	rtb-0c097c175d9c36819 / MyRoute
<input checked="" type="checkbox"/> PrivateSubNet-C	subnet-01e902484d26bd96c	10.0.3.0/24	-	Main (rtb-07b7188bec62040fc / privat...

Selected subnets: subnet-01e902484d26bd96c / PrivateSubNet-C

Buttons: Cancel, Save associations

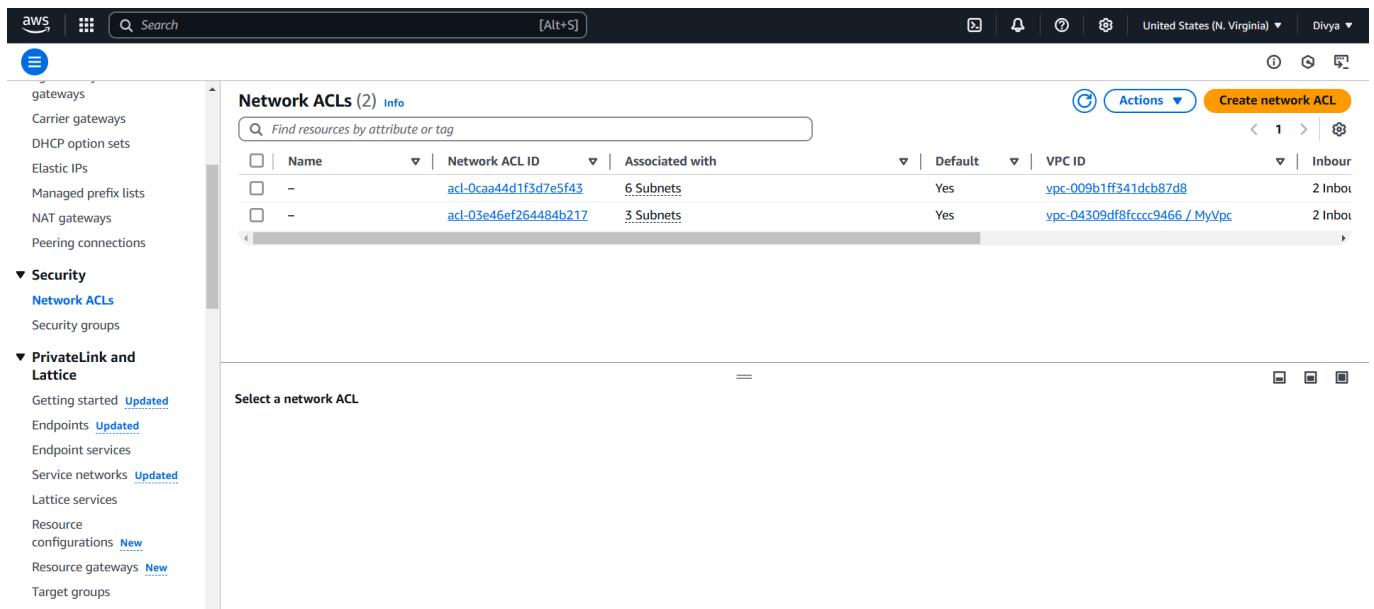
The screenshot shows the 'Edit routes' page for route table **rtb-07b7188bec62040fc**. It displays a table of routes:

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	No
0.0.0.0/0	NAT Gateway	-	No

Buttons: Add route, Remove, Cancel, Preview, Save changes

10.Creating NACL

Initially, the default NACL of MyVPC is automatically associated with all three subnets (PublicSubnet-A, PublicSubnet-B, and PrivateSubnet-C). This default NACL allows all inbound and outbound traffic by default.

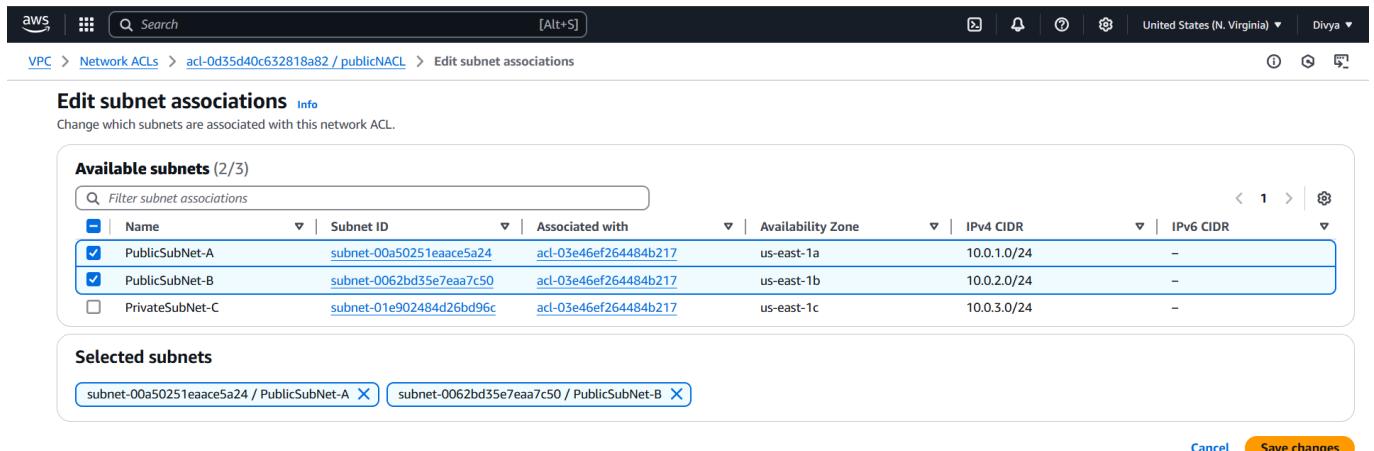


The screenshot shows the AWS VPC Network ACLs page. On the left, there's a navigation sidebar with sections like gateways, security (Network ACLs selected), and PrivateLink and Lattice. The main area displays a table titled "Network ACLs (2) Info". The table has columns for Name, Network ACL ID, Associated with, Default, VPC ID, and Inbound. Two entries are shown:

Name	Network ACL ID	Associated with	Default	VPC ID	Inbound
-	acl-0caa44d1f3d7e5f43	6 Subnets	Yes	vpc-009b1ff541dcb87d8	2 Inbound
-	acl-03e46ef264484b217	3 Subnets	Yes	vpc-04309df8fcccc9466 / MyVpc	2 Inbound

A Network Access Control List (NACL) named **publicNACL** is created within MyVPC to manage traffic flow at the subnet level. NACLs provide an additional layer of security by enforcing stateless inbound and outbound traffic rules.

This PublicNACL is associated with **PublicSubnet-A** and **PublicSubnet-B**, ensuring controlled access for internet-facing resources while maintaining security best practices.



The screenshot shows the "Edit subnet associations" page. The URL is [VPC > Network ACLs > acl-0d35d40c632818a82 / publicNACL > Edit subnet associations](#). The page title is "Edit subnet associations". It says "Change which subnets are associated with this network ACL." Below this is a table titled "Available subnets (2/3)".

Name	Subnet ID	Associated with	Availability Zone	IPv4 CIDR	IPv6 CIDR
PublicSubNet-A	subnet-00a50251eaace5a24	acl-03e46ef264484b217	us-east-1a	10.0.1.0/24	-
PublicSubNet-B	subnet-0062bd35e7eaa7c50	acl-03e46ef264484b217	us-east-1b	10.0.2.0/24	-
PrivateSubNet-C	subnet-01e902484d26bd96c	acl-03e46ef264484b217	us-east-1c	10.0.3.0/24	-

Below the table is a section titled "Selected subnets" with two items: "subnet-00a50251eaace5a24 / PublicSubNet-A" and "subnet-0062bd35e7eaa7c50 / PublicSubNet-B". At the bottom right are "Cancel" and "Save changes" buttons.

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the VPC.

Rule number	Type Info	Protocol Info	Port range Info	Source Info	Allow/Deny Info
100	SSH (22)	TCP (6)	22	117.196.244.190/32	Allow
200	HTTP (80)	TCP (6)	80	0.0.0.0/0	Allow
300	Custom TCP	TCP (6)	1024-65535	0.0.0.0/0	Allow
*	All traffic	All	All	0.0.0.0/0	Deny

[Add new rule](#) [Sort by rule number](#)

[Cancel](#) [Preview changes](#) [Save changes](#)

Edit outbound rules Info

Outbound rules control the outgoing traffic that's allowed to leave the VPC.

Rule number	Type Info	Protocol Info	Port range Info	Destination Info	Allow/Deny Info
100	SSH (22)	TCP (6)	22	117.196.244.190/32	Allow
200	HTTP (80)	TCP (6)	80	0.0.0.0/0	Allow
300	Custom TCP	TCP (6)	1024-65535	0.0.0.0/0	Allow
*	All traffic	All	All	0.0.0.0/0	Deny

[Add new rule](#) [Sort by rule number](#)

[Cancel](#) [Preview changes](#) [Save changes](#)

VPC dashboard <

Network ACLs (1/3) Info

You have successfully updated outbound rules for acl-0d35d40c632818a82 / publicNACL

Name	Network ACL ID	Associated with	Default	VPC ID	Inbound
acl-0caa44d1f3d7e5f43	6 Subnets	Yes	vpc-009b1ff341dcb87d8	2 Inbou	
acl-03e46ef264484b217	subnet-01e902484d26bd96c / PrivateSubNet-C	Yes	vpc-04309df8fcc9466 / MyVpc	2 Inbou	
publicNACL	acl-0d35d40c632818a82	2 Subnets	No	vpc-04309df8fcc9466 / MyVpc	4 Inbou

acl-0d35d40c632818a82 / publicNACL

[Details](#) [Inbound rules](#) [Outbound rules](#) [Subnet associations](#) [Tags](#)

Details

Network ACL ID acl-0d35d40c632818a82	Associated with 2 Subnets	Default No	VPC ID vpc-04309df8fcc9466 / MyVpc
---	------------------------------	---------------	---------------------------------------

The publicNACL secures PublicSubnet-A and B, enabling SSH (MyIP), HTTP (0.0.0.0/0), and ephemeral ports for responses. The default NACL protects PrivateSubnet-C, restricting traffic while allowing internal communication via the NAT Gateway.

11. Creating Amazon SNS Topic and Subscriptions

An **SNS Topic** (MyTeam) is created to enable event-driven notifications. **Subscriptions** are added to MyTeam, allowing multiple recipients (such as email addresses, SMS, or AWS services) to receive notifications whenever an event is published to the topic. This setup ensures efficient communication and automated alerts for system events or updates.

The screenshot shows the AWS SNS Topics page. A green success message at the top states: "Topic MYTeam created successfully. You can create subscriptions and send messages to them from this topic." Below this, the "MYTeam" topic details are shown, including its Name (MYTeam), ARN (arn:aws:sns:us-east-1:010438484509:MYTeam), Type (Standard), Display name (-), and Topic owner (010438484509). The "Subscriptions" tab is selected, showing 0 subscriptions. At the bottom, there are tabs for Access policy, Data protection policy, Delivery policy (HTTP/S), Delivery status logging, Encryption, and Tags. The browser's address bar shows the URL as [Amazon SNS > Topics > MYTeam](#). The operating system taskbar at the bottom indicates it's a Windows 10 environment with various pinned icons like File Explorer, Edge, and Mail.

The screenshot shows the "Create subscription" page. It has a "Details" section where the "Topic ARN" is set to arn:aws:sns:us-east-1:010438484509:MYTeam. The "Protocol" dropdown is set to "Email". The "Endpoint" field contains the email address divyapottipothu@gmail.com. A note at the bottom states: "After your subscription is created, you must confirm it." The browser's address bar shows the URL as [Amazon SNS > Subscriptions > Create subscription](#).

Subscriptions are added to **MyTeam**, and each subscriber receives a confirmation request. Once the subscription is confirmed, the recipient starts receiving notifications whenever an event is published to the topic.

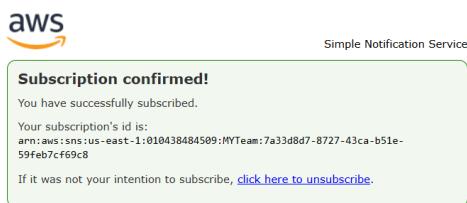
The screenshot shows a Gmail inbox with 334 unread messages. An email from "AWS Notifications <no-reply@sns.amazonaws.com>" is selected, titled "AWS Notification - Subscription Confirmation". The message body contains the following text:

You have chosen to subscribe to the topic:
arn:aws:sns:us-east-1:010438484509:MYTeam

To confirm this subscription, click or visit the link below. (If this was in error no action is necessary).
[Confirm subscription](#)

Please do not reply directly to this email. If you wish to remove yourself from receiving all future SNS subscription confirmation requests please send an email to [sns-opt-out](#).

Below the message are standard Gmail interaction buttons: Reply, Forward, and Delete.



The screenshot shows the Amazon SNS console under the "Topics" section. A specific subscription is selected, showing its details:

Subscription: 7a33d8d7-8727-43ca-b51e-59feb7cf69c8

Details	Status
ARN arn:aws:sns:us-east-1:010438484509:MYTeam:7a33d8d7-8727-43ca-b51e-59feb7cf69c8	Confirmed
Endpoint divyapottipothu@gmail.com	Protocol EMAIL
Topic MYTeam	
Subscription Principal arn:aws:iam::010438484509:root	

Below the main details, there are tabs for "Subscription filter policy" and "Redrive policy (dead-letter queue)".

12. Configuring Amazon S3 for Scalable Storage

Two **Amazon S3 buckets** are created to serve distinct purposes. The FlowLogs Storage Bucket is designated for storing VPC(MyVpc) Flow Logs, enabling network traffic monitoring and security analysis. The General Purpose Bucket is used for storing application data, backups, and other essential files. These buckets ensure efficient data organization, security, and seamless integration with AWS services for scalable and reliable storage solutions.

The screenshot shows the AWS S3 console with a success message: "Successfully created bucket 'bucket-120001xd'". Below it, there's an account snapshot and a general purpose buckets section listing two buckets: 'bucket-120001xd' and 'flowlogs-551245'. The 'Create bucket' button is visible at the top right of the table.

Name	AWS Region	IAM Access Analyzer	Creation date
bucket-120001xd	US East (N. Virginia) us-east-1	View analyzer for us-east-1	March 2, 2025, 11:14:13 (UTC+05:30)
flowlogs-551245	US East (N. Virginia) us-east-1	View analyzer for us-east-1	March 2, 2025, 11:13:10 (UTC+05:30)



13. Enabling VPC Flow Logs and Storing in S3

VPC **Flow Logs** are created for MyVPC to capture and monitor network traffic. The logs are configured to be stored in an **S3 bucket** using its ARN (Amazon Resource Name). This setup ensures centralized log storage, enabling security analysis, troubleshooting, and compliance auditing while maintaining efficient data management.

The screenshot shows the AWS VPC dashboard with a list of VPCs. A context menu is open over the 'MyVpc' entry, showing options like 'Create default VPC', 'Create flow log' (which is highlighted in blue), 'Edit VPC settings', 'Edit CIDRs', 'Manage middlebox routes', 'Manage tags', and 'Delete VPC'.

14. Creating IAM Role for EC2 with S3 Full Access

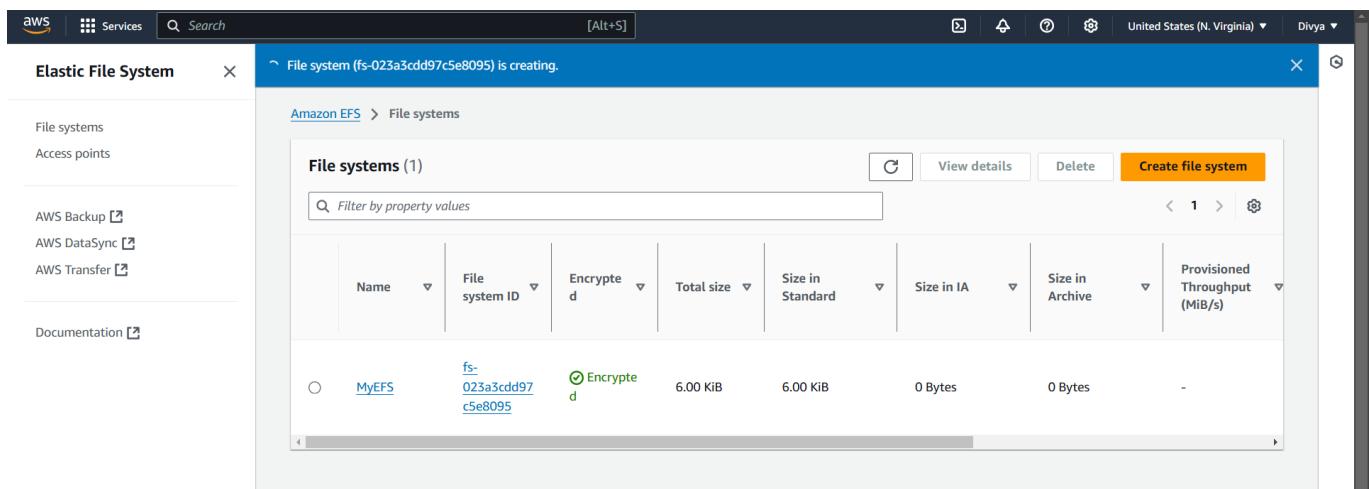
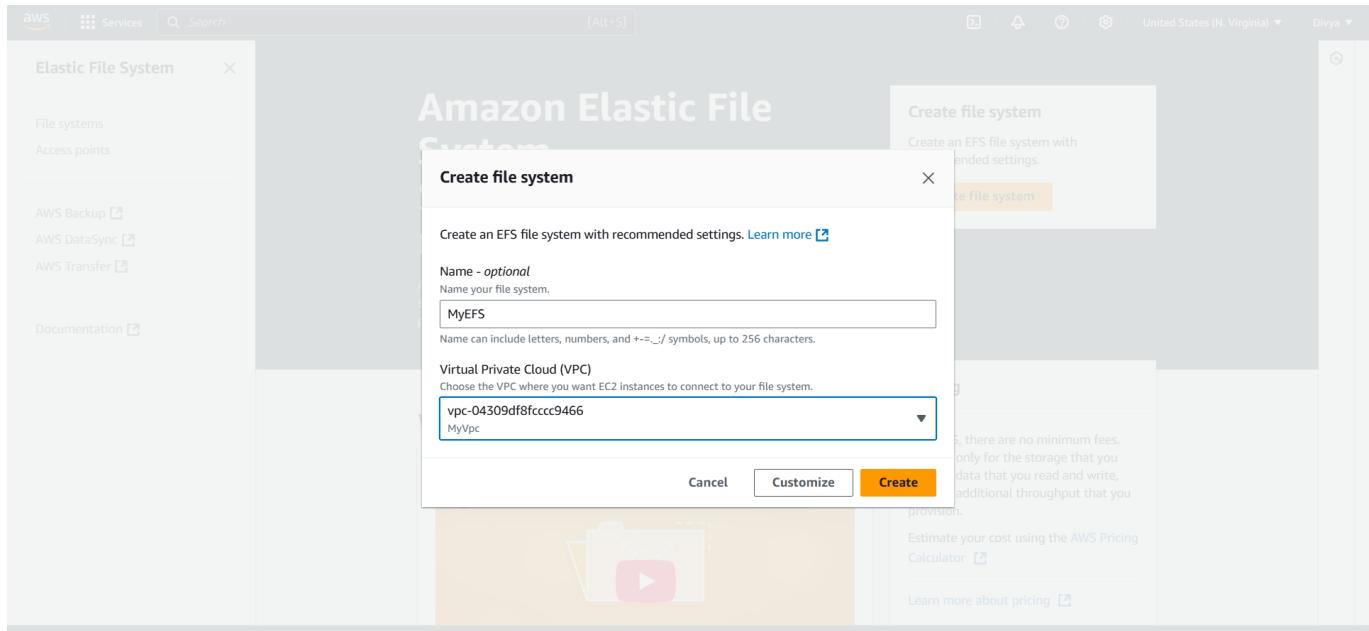
An **IAM role** (Ec2_S3fullAccess) is created to grant **EC2 instances** full access to **Amazon S3**. This role is assigned the **AmazonS3FullAccess** policy, allowing EC2 instances to read, write, and manage objects in S3 buckets without requiring manual credential configuration. By attaching this IAM role to EC2 instances, secure and seamless interaction with S3 is enabled.

The screenshot shows the 'Add permissions' step of the IAM role creation wizard. On the left, a vertical navigation bar indicates 'Step 1 Select trusted entity', 'Step 2 Add permissions' (which is selected), and 'Step 3 Name, review, and create'. The main area is titled 'Add permissions' with a 'Permissions policies (1/1037)' section. It shows a search bar with 's3full', a filter 'All types' with '1 match', and a list containing 'Policy name' (checked) and 'AmazonS3FullAccess' (checked). A detailed description for 'AmazonS3FullAccess' is visible: 'Provides full access to all buckets via the ...'. Below this is a section for 'Set permissions boundary - optional'. At the bottom right are 'Cancel', 'Previous', and 'Next' buttons.

The screenshot shows the 'Roles' page in the AWS IAM console. The sidebar includes 'Identity and Access Management (IAM)', 'Dashboard', 'Access management' (with 'Roles' selected), 'Access reports', and 'Organization activity'. The main content area displays a green success message: 'Role Ec2_S3fullAccess created.' Below this, a table lists three roles: 'AWSServiceRoleForSupport', 'AWSServiceRoleForTrustedAdvisor', and 'Ec2_S3fullAccess'. The 'Ec2_S3fullAccess' row shows it was created by 'AWS Service: support (Service-Linker)' and has 'Last activity' listed as '-'. There are 'View role', 'Delete', and 'Create role' buttons for each row. Below the table, sections for 'Roles Anywhere' (info), 'Access AWS from your non AWS workloads' (info), 'X.509 Standard' (info), and 'Temporary credentials' (info) are shown. The bottom of the screen shows a taskbar with various icons and the date '02-03-2025'.

15. Setting Up Amazon Elastic File System (MyEFS)

Amazon **Elastic File System** (MyEFS) is created to provide scalable and fully managed file storage for **Linux-based** EC2 instances. EFS allows multiple instances to access the same file system concurrently, making it ideal for shared storage, application hosting, and data persistence. It is designed to scale automatically based on demand, ensuring high availability and durability. By mounting EFS on Linux instances, applications can seamlessly read and write data without the need for manual storage management.



When creating **Amazon Elastic File System (EFS)**, enabling **DNS hostnames** in **MyVPC** is essential because EFS uses **DNS names** to allow EC2 instances to connect to the file system. Since EFS does not use IP addresses directly, enabling **DNS resolution** ensures that EC2 instances within the VPC can **mount**

and access the EFS file system using the automatically assigned DNS names. Without DNS enabled, instances would not be able to resolve the EFS mount targets, leading to connection failures.

16. Defining Security Group for Web Server

A Security Group named **WebSG** is created for EC2 instances deployed in **MyVPC**'s PublicSubnets. To ensure secure access, the inbound rules allow **SSH** (port 22) access only from **MyIP** for administrative control and **HTTP** (port 80) access from **all IP addresses** to enable public web traffic. The outbound rules remain open by default, allowing unrestricted outbound communication for the instances. This configuration ensures secure remote access while making the web server accessible to users.

EC2

Security group (sg-0d9431fbfeeb670cc | WebSG) was created successfully

sg-0d9431fbfeeb670cc - WebSG

Details

Security group name WebSG	Security group ID sg-0d9431fbfeeb670cc	Description WebSG	VPC ID vpc-04309df8fcccc9466
Owner 010438484509	Inbound rules count 2 Permission entries	Outbound rules count 1 Permission entry	

Inbound rules Outbound rules Sharing - new VPC associations - new Tags

Inbound rules (2)

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-0a8370ab791f3287b	IPv4	SSH	TCP	22
-	sqr-0deda9d263ec3bd19	IPv4	HTTP	TCP	80

17. Creating LoadBalancer

A Classic Load Balancer (MyLB) is created in MyVPC, spanning Availability Zones A and B to ensure high availability and fault tolerance. It is associated with WebSG for web traffic security and the default Security Group, which is required for EFS access. The idle timeout is set to 300 seconds, ensuring stable session management for applications relying on persistent connections. This setup efficiently distributes incoming traffic across multiple instances, improving scalability and reliability.

Basic configuration

Load balancer name
Name must be unique within your AWS account and can't be changed after the load balancer is created.

Scheme **Internet-facing**

- Serves internet-facing traffic.
- Has public IP addresses.
- DNS name is publicly resolvable.
- Requires a public subnet.

Internal

- Serves internal traffic.
- Has private IP addresses.
- DNS name is publicly resolvable.

Network mapping Info
The load balancer routes traffic to targets in the selected subnets, and in accordance with your network settings.

VPC **Info**
Select the virtual private cloud (VPC) for your targets or you can [create a new VPC](#). Only VPCs with an internet gateway are available for selection. The selected VPC cannot be changed after the load balancer is created. When selecting a VPC for your load balancer, ensure each subnet has a CIDR block with at least a /27 bitmask and at least 8 free IP addresses. [Learn more](#)

Availability Zones and subnets
Select at least one Availability Zone and one subnet for each zone. We recommend selecting at least two Availability Zones. The load balancer will route traffic only to targets in the selected Availability Zones. Availability Zones that are not supported by the load balancer or the VPC are not available for selection.

The screenshot shows the AWS Elastic Load Balancing (ELB) service in the EC2 section. On the left, there's a navigation sidebar with options like Capacity Reservations, Images, Elastic Block Store, Network & Security, Load Balancing, Auto Scaling, and more. The main area displays a table titled 'Load balancers (1/1)' with one entry: 'MyLB'. The table includes columns for Name, DNS name, State, VPC ID, Availability Zones, Type, and Date created. Below the table, a detailed view for 'Load balancer: MyLB' is shown, with tabs for Details, Listeners, Network mapping, Security, Health checks, Target instances, Monitoring, Attributes, and Tags. The 'Details' tab is selected, showing information such as Load balancer type (Classic), Status (Loading), Scheme (Internet-facing), Hosted zone (Z35SXDOTRQ7X7K), VPC (vpc-04309df8fcccc9466), Availability Zones (subnet-00a50251eaace5a24), and Date created (March 2, 2025, 13:07 (UTC+05:30)).

18. Configuring a Launch Template for EC2 with EFS and S3 Access

A **Launch Template (MyTMPT)** is created to streamline EC2 instance deployment with pre-configured settings. It is based on Amazon Linux and uses the **projectKey** key pair for secure SSH access. The instance is associated with WebSG for web security and the default Security Group for additional network controls. To enable seamless storage integration, the instance mounts an Amazon EFS file system, ensuring persistent and shared storage across multiple instances. Additionally, it is configured with an IAM role (EC2_S3_FullAccess) to securely interact with Amazon S3, allowing retrieval of web content. The Apache web server (HTTPD) is installed and configured to serve an index.html file from EFS, providing a scalable and highly available web hosting environment.

```
#!/bin/bash
sudo su -
yum update -y
mkdir /divya
#Attach EFS using NFS Client command
yum install httpd -y
cd /var/www/html
echo "Hello All .....!!!!" > index.html
service httpd start
chkconfig httpd on
```

aws Services Search ec2

United States (N. Virginia) Divya

Elastic File System > Amazon EFS > File systems > fs-023a3cdd97c5e8095

Attach

Mount your Amazon EFS file system on a Linux instance. [Learn more](#)

Mount via DNS Mount via IP

Using the EFS mount helper:

```
sudo mount -t efs -o tls fs-023a3cdd97c5e8095:/ efs
```

Using the NFS client:

```
sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport fs-023a3cdd97c5e8095.efs.us-east-1.amazonaws.com:/ efs
```

See our user guide for more information. [Learn more](#)

[Close](#)

CloudWatch Feedback

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookies

aws Search [Alt+S]

United States (N. Virginia) Divya

EC2 > [Launch templates](#) > Create launch template

Success
Successfully created MyTMPT(lt-0f2635fa6d0cec3fb).

[Actions log](#)

Next Steps

[Launch an instance](#)
With On-Demand Instances, you pay for compute capacity by the second (for Linux, with a minimum of 60 seconds) or by the hour (for all other operating systems) with no long-term commitments or upfront payments.
Launch an On-Demand Instance from your launch template.

[Launch instance from this template](#)

[Create an Auto Scaling group from your template](#)
Amazon EC2 Auto Scaling helps you maintain application availability and allows you to scale your Amazon EC2 capacity up or down automatically according to conditions you define. You can use Auto Scaling to help ensure that you are running your desired number of Amazon EC2 instances during demand spikes to maintain performance and decrease capacity during lulls to reduce costs.

[Create Auto Scaling group](#)

[Create Spot Fleet](#)
A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Because Spot Instances enable you to request unused EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly. The hourly price for a Spot Instance (of each instance type in each Availability Zone) is set by Amazon EC2, and adjusted gradually based on the long-term supply of and demand for Spot Instances. Spot instances are well-suited for data-analysis, batch jobs, background processing, and optional tasks.

[Create Spot Fleet](#)

aws Search [Alt+S]

United States (N. Virginia) Divya

EC2 > Security Groups > sg-0d9431fbfeeb670cc - WebSG

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

Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By
lt-0f2635fa6d0cec3fb	MyTMPT	1	1	2025-03-02T06:15:16.000Z	arn:aws:iam::010438

Select a launch template

19. Setting Up Auto Scaling Group (MyASG) for High Availability

The Auto Scaling Group (MyASG) is created to ensure high availability and scalability of EC2 instances in MyVPC across Availability Zones A and B. It uses the MyTMPT launch template, ensuring consistent instance configurations. The MyLB load balancer is associated with the group to evenly distribute traffic and improve fault tolerance. Health checks are configured with a threshold of 150 seconds to ensure only healthy instances handle traffic. Additionally, an SNS notification is enabled to send alerts about scaling activities, and a tag (Name: WebServer) is assigned for easy identification and management of instances within the group.

The screenshot shows the AWS Auto Scaling 'Create Auto Scaling group' wizard. The steps are:

- Step 1: Choose launch template**
 - Group details**
Auto Scaling group name: MyASG
 - Launch template**
Launch template: MyTMPT [Edit]
Version: Default
Description: MyTMPT
- Step 2: Choose instance launch options**
 - Network**
VPC: vpc-04309df8fcccc9466 [Edit]
 - Availability Zones and subnets**

Availability Zone	Subnet	Subnet CIDR range
us-east-1a	subnet-00a50251eaace5a24 [Edit]	10.0.1.0/24
- Step 5: Add notifications**
 - Notifications**
Notification 1: SNS Topic MYTeam
 - Event types**
 Launch
 Terminate
 Fail to launch
 Fail to terminate
- Step 6: Add tags**
 - Tags (1)**

Key	Value	Tag new instances
WebServer	Yes	

At the bottom, there are buttons for **Preview code**, **Cancel**, **Previous**, and a large orange **Create Auto Scaling group** button.

The screenshot shows the AWS EC2 Auto Scaling Groups page. A success message at the top indicates "MyASG, 1 Notification created successfully". Below it, the "Auto Scaling groups (1/1) Info" section shows one group named "MyASG" with a launch template "MyTMPT | Version Default". The group has a status of "Updating capacity..." with a current count of 0 and a desired capacity of 4. The minimum and maximum capacities are set to 4 and 10 respectively. The "Details" tab is selected, showing the "MyASG Capacity overview" which includes the ARN: arn:aws:autoscaling:us-east-1:010438484509:autoScalingGroup:311d811d-87bc-4537-bb1b-7607023cacfd:autoScalingGroupName/MyASG. It also displays the desired capacity (4), scaling limits (4 - 10), desired capacity type (Units (number of instances)), and status (Updating capacity).

After creating the Auto Scaling Group (MyASG), four EC2 instances are automatically launched based on the defined scaling policies. These instances are evenly distributed across Availability Zones A and B, ensuring high availability and fault tolerance.

20. Setting Up Bastion and Database Servers

A Bastion Server is deployed in MyVpc within Subnet-A to securely manage access to private instances. It is configured with an inbound rule allowing SSH access only from MyIP, ensuring restricted and controlled entry. The Database Server is provisioned in Subnet-C, which is a private subnet with public IP disabled by default for enhanced security. Remote access to the database is only allowed via custom SSH from the Bastion Server's private IP, while MySQL connections are restricted to 10.0.1.0/24 and 10.0.2.0/24, ensuring secure and controlled communication within the network.

The screenshot shows the AWS EC2 Instances Launch an instance page. A success message at the top indicates "Successfully initiated launch of instance (i-0aaafac1136b66f55)". Below it, the "Launch log" section shows the following steps and their statuses: "Initializing requests" (Succeeded), "Creating security groups" (Succeeded), "Creating security group rules" (Succeeded), and "Launch initiation" (Succeeded).

Name and tags

Name: DataBaseDB

Application and OS Images (Amazon Machine Image)

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.

Software Image (AMI)

Amazon Linux 2023 AMI 2023.6.2... [read more](#)
ami-05b10e08d247fb927

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available when used with free tier AMIs). 750

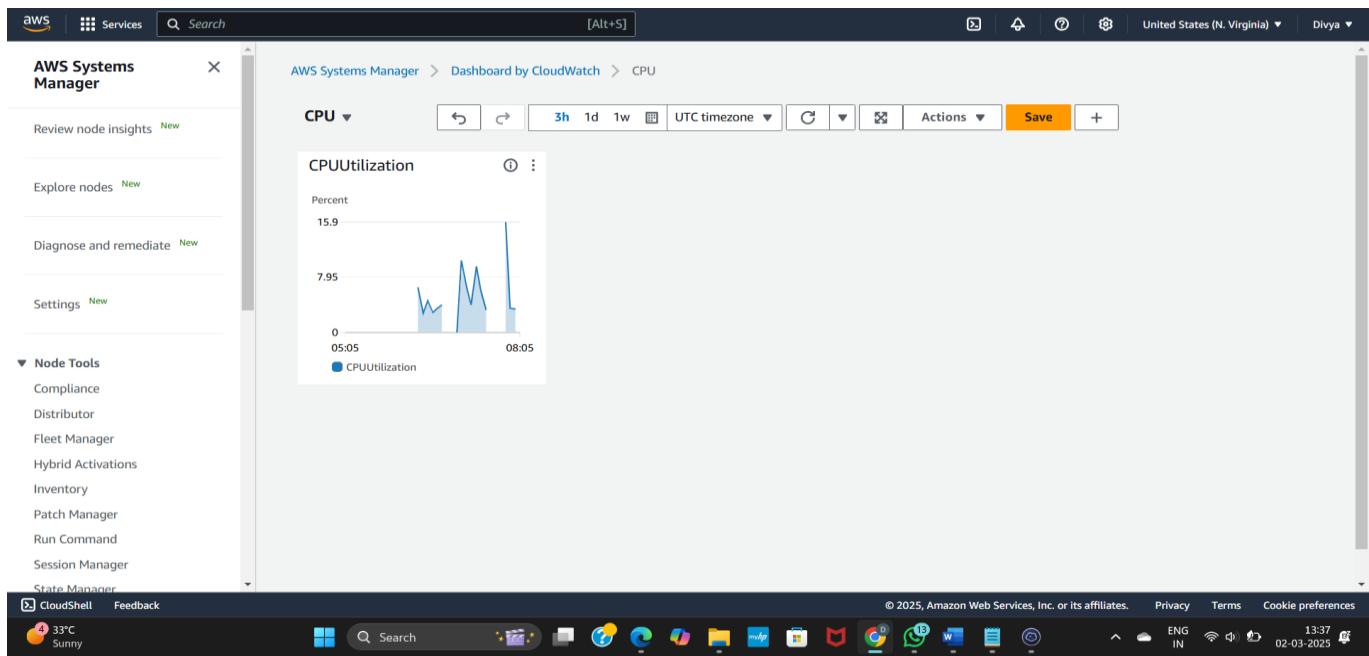
Launch instance

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
WebServer	i-04ccdf4573655bc6	Running	t2.micro	Initializing	View alarms +	us-east-1a	ec2-107-
WebServer	i-0ecf7e5fa2e2abdb6	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-100-
WebServer	i-027413bc8b9fb5e87	Shutting-d...	t2.micro	-	View alarms +	us-east-1a	ec2-44-2
WebServer	i-0c4cd6715ef67b093	Terminated	t2.micro	-	View alarms +	us-east-1a	-
BastionServer	i-0aaafac1136b66ff5	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-54-2
WebServer	i-093680616da41f8c4	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-98-8
WebServer	i-00a08bec2a11d6fbf	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-98-8
DataBaseDB	i-02830fb106105e125	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1c	-

Select an instance

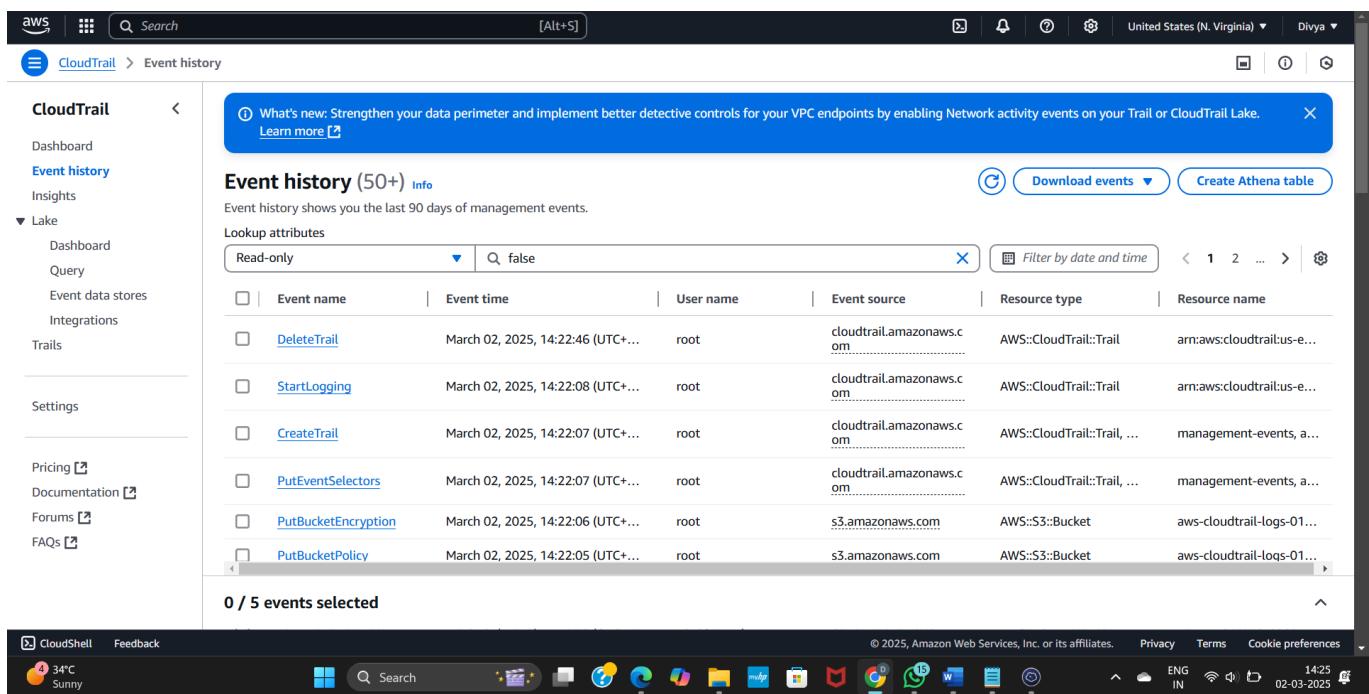
21.Cloud watch

A **CloudWatch Dashboard** is created to monitor the **CPU utilization** of **EC2** instances launched by **MyASG** (Auto Scaling Group). The process begins by creating a new dashboard in the CloudWatch Console and adding a Stacked Area widget for clear visualization. Under EC2 metrics, the monitoring is set to track instances by Auto Scaling Group, selecting MyASG and its CPU Utilization metric. Once configured, the dashboard is created and saved, ensuring real-time insights into resource performance, optimizing scaling decisions, and maintaining efficient workload management.



22. Cloud Trail

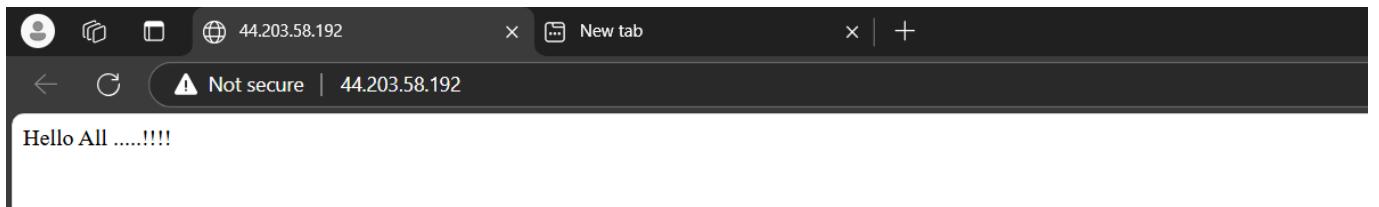
Navigating to the AWS **CloudTrail** Dashboard and accessing the **Event History** without creating a trail provides an overview of recent AWS account activity, including service usage and IAM actions. The Event History displays the last 90 days of recorded management events, such as API calls related to EC2, IAM, and S3. Since no CloudTrail trail has been created, events are only retained temporarily and are not stored for long-term auditing or analysis. Additionally, detailed insights such as **S3 object-level tracking** or **anomaly detection** are unavailable, limiting visibility into specific data access and security events.



23.Checking

23.1 Checking WebServer

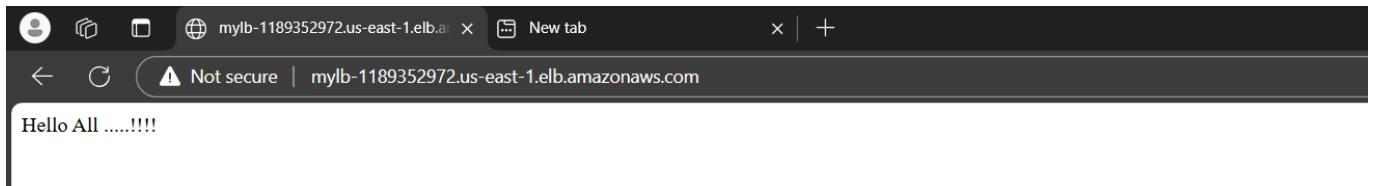
One of the web servers, with the public IP 44.203.58.192, was accessed by pasting it into the browser. The webpage loaded successfully, confirming that the web server is operational, the Apache (HTTPD) service is running, and the security group and routing configurations allow external HTTP traffic. This verifies that the Auto Scaling Group has properly launched the instance and is serving web requests as expected.



23.2 Checking Load Balancer

The Load Balancer (LB) DNS was pasted into the web browser, and the webpage loaded successfully. This confirms that the Load Balancer is correctly distributing traffic across the web servers, the target instances are healthy, and the security groups and routing configurations are properly set up. This also ensures that the Auto Scaling Group is functioning as expected, dynamically managing instances to handle incoming traffic efficiently.

A screenshot of the AWS EC2 Load balancers console. On the left, there is a navigation sidebar with options like Capacity Reservations, Images, Elastic Block Store, Network & Security, and Load Balancing. Under Load Balancing, 'Load Balancers' is selected. The main pane shows a table titled 'Load balancers (1/1)' with one entry: 'MyLB'. The table includes columns for Name, DNS name, State, VPC ID, Availability Zones, Type, and Date created. The 'Actions' button is visible at the top right of the table. Below the table, there is a detailed view for 'Load balancer: MyLB' with tabs for Details, Listeners, Network mapping, Security, Health checks, Target instances, Monitoring, Attributes, and Tags. The 'Details' tab is selected, showing information such as Load balancer type (Classic), Status (Loading...), VPC (vpc-04309df8fcccc9466), and Date created (March 2, 2025, 13:07 (UTC+05:30)).



23.3 Checking Bastion

The screenshot shows the AWS EC2 Instances page. The left sidebar includes sections for Global View, Events, Instances (with sub-options like Instance Types, Launch Templates, etc.), Images, Elastic Block Store, and Network & Security. The main area displays a table of instances:

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
<input checked="" type="checkbox"/>	BastionServer	i-0aaafac1136b66f55	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-54-2
<input type="checkbox"/>	WebServer	i-0f36056d90469cb7e	Terminated	t2.micro	-	View alarms +	us-east-1a	-
<input type="checkbox"/>	WebServer	i-01f646e6ed33182ae	Terminated	t2.micro	-	View alarms +	us-east-1b	-
<input type="checkbox"/>	WebServer	i-0119013748ad54e02	Terminated	t2.micro	-	View alarms +	us-east-1b	-
<input type="checkbox"/>	WebServer	i-093680616da41f8c4	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-98-8
<input type="checkbox"/>	WebServer	i-00a08bec2a11d6fbf	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1b	ec2-98-8

Below the table, a detailed view is shown for the Bastion Server instance (i-0aaafac1136b66f55). It includes fields for Instance ID, Public IPv4 address (54.210.235.209), Instance state (Running), Hostname type (IP name: ip-10-0-1-192.ec2.internal), Private IP DNS name (IPv4 only) (ip-10-0-1-192.ec2.internal), Private IP4 addresses (10.0.1.192), and Public IPv4 DNS (ec2-54-210-235-209.compute-1.amazonaws.com).

Connected to the Bastion Server using its public IP (54.210.235.209) in Termius, utilizing the projectKey for secure SSH access.

After connecting to the Bastion Server, internet access was verified by executing the following commands:

- sudo su
- yum install httpd -y

The installation was completed successfully, confirming that the Bastion Server has outbound internet access and can download and install packages from external repositories.

```

[ec2-user@ip-10-0-1-192 ~]$ sudo su
[root@ip-10-0-1-192 ec2-user]# yum install httpd -y
Amazon Linux 2023 Kernel Livepatch repository
Dependencies resolved.
=====
Package           Architecture Version      Repository  Size
=====
Installing:
httpd            x86_64      2.4.62-1.amzn2023 amazonlinux 48 k
Installing dependencies:
apr              x86_64      1.7.5-1.amzn2023.0.4 amazonlinux 129 k
apr-util         x86_64      1.6.3-1.amzn2023.0.1 amazonlinux 98 k
generic-logos-htpd noarch      18.0.0-12.amzn2023.0.3 amazonlinux 19 k
httpd-core       x86_64      2.4.62-1.amzn2023 amazonlinux 1.4 M
httpd-filesystem noarch      2.4.62-1.amzn2023 amazonlinux 14 k
httpd-tools      x86_64      2.4.62-1.amzn2023 amazonlinux 81 k
libbrotli        x86_64      1.0.9-4.amzn2023.0.2 amazonlinux 315 k
mailcap          noarch      2.1.49-3.amzn2023.0.3 amazonlinux 33 k
Installing weak dependencies:
apr-util-openssl x86_64      1.6.3-1.amzn2023.0.1 amazonlinux 17 k
mod_http2        x86_64      2.0.27-1.amzn2023.0.3 amazonlinux 166 k
mod_lua          x86_64      2.4.62-1.amzn2023 amazonlinux 61 k
=====
Transaction Summary
=====
Install 12 Packages

Total download size: 2.3 M
Installed size: 6.9 M
Downloading Packages:
(1/12): apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64.rpm 476 kB/s | 17 kB 00:00
(2/12): apr-1.7.5-1.amzn2023.0.4.x86_64.rpm 2.6 MB/s | 129 kB 00:00
(3/12): apr-util-1.6.3-1.amzn2023.0.1.x86_64.rpm 1.8 MB/s | 98 kB 00:00
(4/12): generic-logos-htpd-18.0.0-12.amzn2023.0.3.noarch.rpm 847 kB/s | 19 kB 00:00
(5/12): httpd-2.4.62-1.amzn2023.x86_64.rpm 2.1 MB/s | 48 kB 00:00
(6/12): httpd-filesystem-2.4.62-1.amzn2023.noarch.rpm 418 kB/s | 14 kB 00:00
(7/12): httpd-core-2.4.62-1.amzn2023.x86_64.rpm 28 MB/s | 1.4 MB 00:00
(8/12): httpd-tools-2.4.62-1.amzn2023.x86_64.rpm 1.3 MB/s | 81 kB 00:00
(9/12): libbrotli-1.0.9-4.amzn2023.0.2.x86_64.rpm 6.3 MB/s | 315 kB 00:00
(10/12): mailcap-2.1.49-3.amzn2023.0.3.noarch.rpm 785 kB/s | 33 kB 00:00
(11/12): mod_http2-2.0.27-1.amzn2023.x86_64.rpm 1.3 MB/s | 166 kB 00:00
(12/12): mod_lua-2.4.62-1.amzn2023.x86_64.rpm 61 kB/s | 61 kB 00:00
=====

```

23.4 Checking DatabaseDB

The projectKey was securely **uploaded** to the Bastion Server from the local system using SFTP (Secure File Transfer Protocol). This was done to facilitate secure SSH access to the Database Server from the Bastion.

Name	Date Modified	Size	Kind
idphoto.jpg	1/4/2024, 3:30 PM	18.86 kB	jpg
larger_number.cpp	11/1/2023, 9:45 AM	870.00 Bytes	cpp
larger_number.exe	11/1/2023, 9:47 AM	69.15 kB	exe
new_csp_ppt.pptx	11/23/2023, 10:47 PM	126.70 kB	pptx
password.pdf	7/24/2024, 10:01 PM	71.86 kB	pdf
portfolio.html	12/2/2024, 8:35 PM	7.78 kB	html
ProjectKey.pem	3/2/2025, 11:33 AM	1.64 kB	pem
redhat_python_certificate.pdf	11/6/2023, 9:06 PM	46.70 kB	file
resume_photo.jpg	11/6/2023, 9:47 PM	51.55 kB	jpg
rtr47FB.tmp	5/24/2023, 10:10 PM	1.00 GB	tmp
sif.cpp	11/15/2023, 3:17 PM	1.00 kB	cpp
System Volume Information	1/1/1970, 5:30 AM	0 Bytes	file

Name	Date Modified	Size	Kind
..			
ProjectKey.pem	3/2/2025, 1:56 PM	1.64 kB	pem

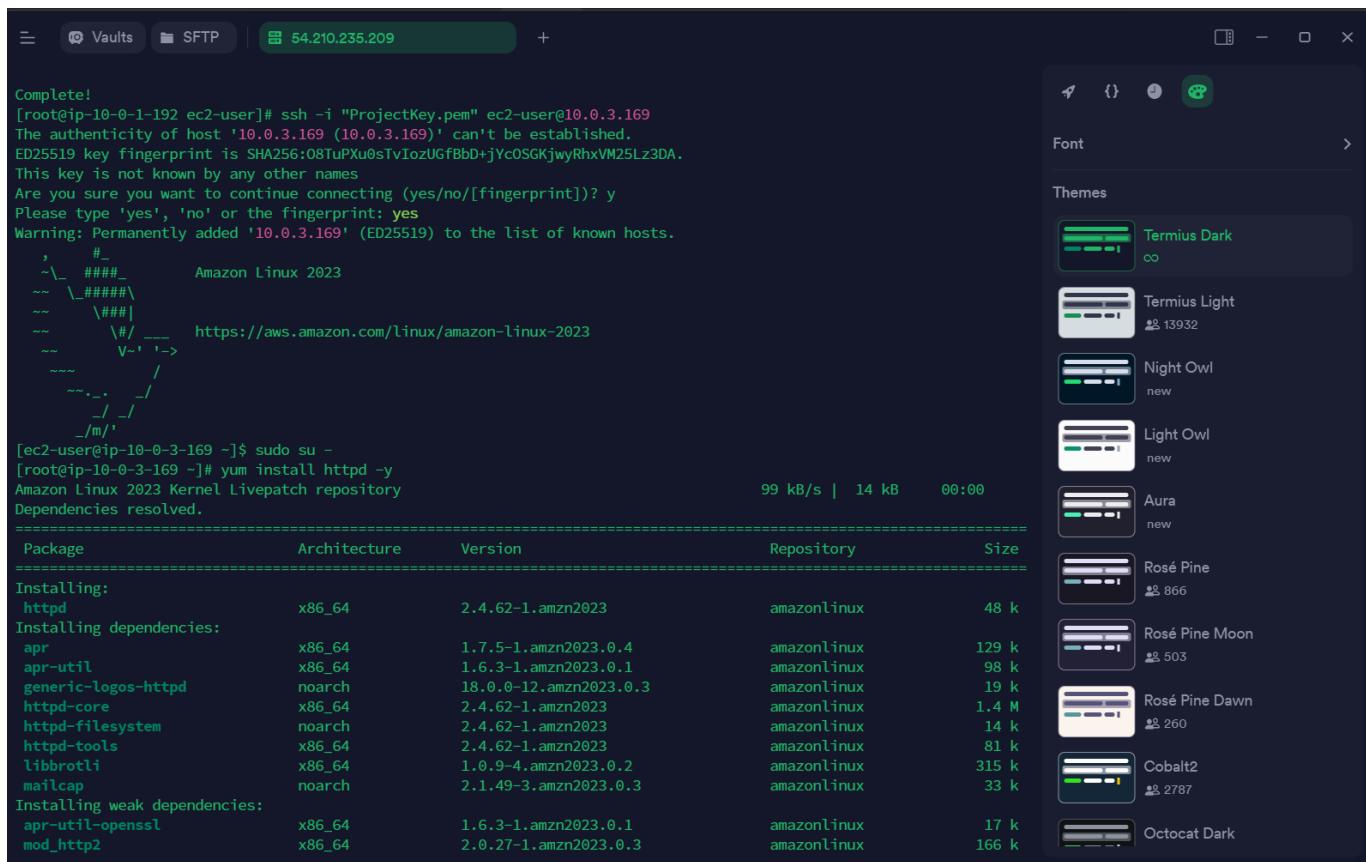
Screenshot of the AWS EC2 Connect to instance page for instance i-02830fb106105e125 (DataBaseDB). The SSH client tab is selected. It shows instructions for connecting via SSH:

- Open an SSH client.
- Locate your private key file. The key used to launch this instance is ProjectKey.pem
- Run this command, if necessary, to ensure your key is not publicly viewable.
chmod 400 "ProjectKey.pem"
- Connect to your instance using its Private IP:
10.0.3.169

Example:
ssh -i "ProjectKey.pem" ec2-user@10.0.3.169

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Connected to **DataBaseDB(10.0.3.169)** using the SSH command with the **ProjectKey.pem**. To verify internet access, attempted to install **HTTPD** using sudo yum install httpd -y. The installation completed successfully, confirming that **DataBaseDB** has outbound internet connectivity.



```

Complete!
[root@ip-10-0-1-192 ~]# ssh -i "ProjectKey.pem" ec2-user@10.0.3.169
The authenticity of host '10.0.3.169 (10.0.3.169)' can't be established.
ED25519 key fingerprint is SHA256:08TuPXu0sTvIozUGfBbD+jYcOSGKjwyRhxVM25Lz3DA.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '10.0.3.169' (ED25519) to the list of known hosts.

,
  _#
 ~\_ #####_      Amazon Linux 2023
 ~~ \#####\_
 ~~  #####|_
 ~~  \#/ ___  https://aws.amazon.com/linux/amazon-linux-2023
 ~~  V~`  `->
 ~~   /`_
 ~~ .` _/
 ~~ /` _/
 ~~ /` _/
 ~~ /` _/
 [ec2-user@ip-10-0-3-169 ~]$ sudo su -
[root@ip-10-0-3-169 ~]# yum install httpd -y
Amazon Linux 2023 Kernel Livepatch repository
Dependencies resolved.
=====
Package           Architecture Version       Repository  Size
=====
Installing:
httpd            x86_64      2.4.62-1.amzn2023  amazonlinux  48 k
Installing dependencies:
apr              x86_64      1.7.5-1.amzn2023.0.4  amazonlinux 129 k
apr-util         x86_64      1.6.3-1.amzn2023.0.1  amazonlinux  98 k
generic-logos-httpd  noarch    18.0.0-12.amzn2023.0.3  amazonlinux  19 k
httpd-core       x86_64      2.4.62-1.amzn2023  amazonlinux  1.4 M
httpd-filesystem noarch    2.4.62-1.amzn2023  amazonlinux  14 k
httpd-tools      x86_64      2.4.62-1.amzn2023  amazonlinux  81 k
libbrotli        x86_64      1.0.9-4.amzn2023.0.2  amazonlinux 315 k
mailcap          noarch    2.1.49-3.amzn2023.0.3  amazonlinux  33 k
Installing weak dependencies:
apr-util-openssl x86_64      1.6.3-1.amzn2023.0.1  amazonlinux  17 k
mod_http2        x86_64      2.0.27-1.amzn2023.0.3  amazonlinux 166 k

```

Instances (1/14) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
webServer	i-0aa08ec2a110b0r	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1d	ec2-98-8
	i-046866d4f3a461301	Terminated	t2.micro	-	View alarms +	us-east-1b	-
	i-073b02de4c8ca0702	Terminated	t2.micro	-	View alarms +	us-east-1b	-
	WEBSERVER	Terminated	t2.micro	-	View alarms +	us-east-1b	-
	 DataBaseDB	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1c	-

i-02830fb106105e125 (DataBaseDB)

Instance summary

Instance ID	Public IPv4 address	Private IPv4 addresses
i-02830fb106105e125	-	10.0.3.169
IPv6 address	Instance state	Public IPv4 DNS
-	Running	-
Hostname type	Private IP DNS name (IPv4 only)	
IP name: io-10-0-3-169.ec2.internal	io-10-0-3-169.ec2.internal	

23.5 Checking S3 files & website are accessing

Amazon S3

General purpose buckets

Account snapshot - updated every 24 hours All AWS Regions

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

[View Storage Lens dashboard](#)

General purpose buckets (2) Info All AWS Regions

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
bucket-120001xd	US East (N. Virginia) us-east-1	View analyzer for us-east-1	March 2, 2025, 11:14:13 (UTC+05:30)
flowlogs-551245	US East (N. Virginia) us-east-1	View analyzer for us-east-1	March 2, 2025, 11:13:10 (UTC+05:30)

After connecting to a web server using its public IP in Termius, the command `aws s3 ls` was executed to list the available S3 buckets associated with the AWS account. The output displayed the names and timestamps of the buckets, confirming that the instance has the necessary AWS credentials and permissions to access Amazon S3.

```
[ec2-user@ip-10-0-2-88 ~]$ sudo su -
Last login: Sun Mar  2 07:52:55 UTC 2025
[root@ip-10-0-2-88 ~]# aws s3 ls
2025-03-02 05:44:13 bucket-120001xd
2025-03-02 05:43:10 flowlogs-551245
[root@ip-10-0-2-88 ~]#
```

23.6 Checking EFS Storage on EC2

Instances (1/14) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
WebServer	i-0c4cd6715ef67b093	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-3-91
BastionServer	i-0aaafac1136b66f55	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a	ec2-54-2
WebServer	i-0f36056d90469cb7e	Terminated	t2.micro	-	View alarms +	us-east-1a	-
WebServer	i-01f646e6ed33182ae	Terminated	t2.micro	-	View alarms +	us-east-1b	-
WebServer	i-0119013748ad54e02	Terminated	t2.micro	-	View alarms +	us-east-1b	-

i-0c4cd6715ef67b093 (WebServer)

Details Status and alarms Monitoring Security Networking Storage Tags

Instance summary

Instance ID: i-0c4cd6715ef67b093
IPv6 address: -
Instance state: Running

Public IPv4 address copied: 3.91.200.127 | open address

Private IPv4 addresses: 10.0.1.38
Public IPv4 DNS: ec2-3-91-200-127.compute-1.amazonaws.com

- EFS allows multiple EC2 instances to access the same file system simultaneously.
- Since both servers (connected via public IPs 98.80.13.198 and 3.91.200.127) are showing the same files (fl.txt, kumari, durgarao) in /divya, it confirms they are mounted to the same EFS volume.
- Any change made on one server (e.g., creating a file or directory) is instantly reflected on the other, proving shared storage.

This setup is commonly used for scalable and high-availability architectures, ensuring seamless file access across multiple instances.

```

98.80.13.198
[ec2-user@ip-10-0-2-88 ~]$ sudo su -
Last login: Sun Mar  2 07:52:55 UTC 2025
[root@ip-10-0-2-88 ~]# aws s3 ls
2025-03-02 05:44:13 bucket-120001xd
2025-03-02 05:43:10 flowLogs-551245
[root@ip-10-0-2-88 ~]# cd /divya
[root@ip-10-0-2-88 divya]# ls
durgarao f1.txt
[root@ip-10-0-2-88 divya]# touch kumari
[root@ip-10-0-2-88 divya]# ls
durgarao f1.txt kumari
[root@ip-10-0-2-88 divya]# 

3.91.200.127
[ec2-user@ip-10-0-1-38 ~]$ sudo su -
Last login: Sun Mar  2 07:52:57 UTC 2025
[root@ip-10-0-1-38 ~]# cd /divya
[root@ip-10-0-1-38 divya]# ls
[root@ip-10-0-1-38 divya]# touch f1.txt
[root@ip-10-0-1-38 divya]# mkdir durgarao
[root@ip-10-0-1-38 divya]# ls
durgarao f1.txt
[root@ip-10-0-1-38 divya]# ls
durgarao f1.txt kumari
[root@ip-10-0-1-38 divya]# 

```

Font:

Themes:

- Termius Dark
- Termius Light
- Night Owl
- Light Owl
- Aura
- Rosé Pine
- Rosé Pine Moon
- Rosé Pine Dawn
- Cobalt2
- Octocat Dark

23.7 Checking Flow Logs of MyVpc

Amazon S3 Copy S3 URI

02/

Objects Properties

Objects (67)

C Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	010438484509_vpcflowlogs_us-east-1_fl-0477bd1d1f05179c3_20250302T0550Z_1da1844e.log.gz	gz	March 2, 2025, 11:29:22 (UTC+05:30)	558.0 B	Standard
<input type="checkbox"/>	010438484509_vpcflowlogs_us-east-1_fl-0477bd1d1f05179c3_20250302T0550Z_95ca8fc1.log.gz	gz	March 2, 2025, 11:24:21 (UTC+05:30)	876.0 B	Standard
<input type="checkbox"/>	010438484509_vpcflowlogs_us-east-1 fl-	gz	March 2, 2025, 11:29:22	1.2 KB	Standard

24. Conclusion

In this project, we successfully built a scalable, secure, and highly available AWS cloud infrastructure using multiple AWS services. The deployment included networking, security, compute, storage, monitoring, and automation components, ensuring efficient resource management and seamless application hosting.

Key highlights of the project:

- Secure and Scalable Network Design: Implemented VPC, subnets, Internet Gateway, NAT Gateway, and NACLs to ensure controlled and secure traffic flow.
- Optimized Compute and Storage: Used EC2, Auto Scaling, Load Balancer, S3, and EFS for flexible, scalable, and fault-tolerant application hosting.
- Robust Security & Access Control: Configured IAM roles, Security Groups, and Bastion Host to enhance security while maintaining operational efficiency.
- Effective Monitoring & Logging: Enabled CloudWatch, CloudTrail, and VPC Flow Logs for real-time tracking, security auditing, and performance optimization.
- Automation & Notifications: Integrated Launch Templates, Auto Scaling Group, and SNS for dynamic resource management and event-driven alerts.

This AWS infrastructure ensures **high availability, fault tolerance, and secure access control**, making it a **reliable and scalable cloud solution** for hosting business applications.

Overall, this project demonstrates how **AWS cloud services can be leveraged to build a robust and production-ready infrastructure**, empowering businesses with **cost-effective, scalable, and secure cloud solutions**.