AWS Backup & Disaster Recovery

Table of Contents

1. Objective

Design and implement a robust backup and disaster recovery (DR) strategy.

- Automate backups using AWS Backup for RDS, EC2, and EFS.
- Implement cross-region replication for DR using S3 Replication and RDS Multi-AZ.
- Create failover mechanisms using Route 53 and AWS Elastic Disaster Recovery.
- Test recovery scenarios using AWS Resilience Hub.
- Optimize costs using lifecycle policies and AWS Storage Gateway.

2. Project Scope

The PoC will focus on establishing a resilient and cost-effective backup and disaster recovery plan using AWS services. The scope includes:

- Backup Automation: Implement AWS Backup for RDS, EC2, and EFS to ensure regular and secure backups.
- Cross-Region Replication: Utilize S3 Replication and RDS Multi-AZ for redundancy across AWS regions.
- Failover Strategy: Configure Route 53 and AWS Elastic Disaster Recovery for seamless failover and minimal downtime.
- Disaster Recovery Testing: Use AWS Resilience Hub to simulate and validate recovery scenarios.
- Cost Optimization: Implement lifecycle policies and AWS Storage Gateway to manage storage costs effectively.
- Security and Compliance: Ensure backups are encrypted and access-controlled while maintaining compliance with industry standards.

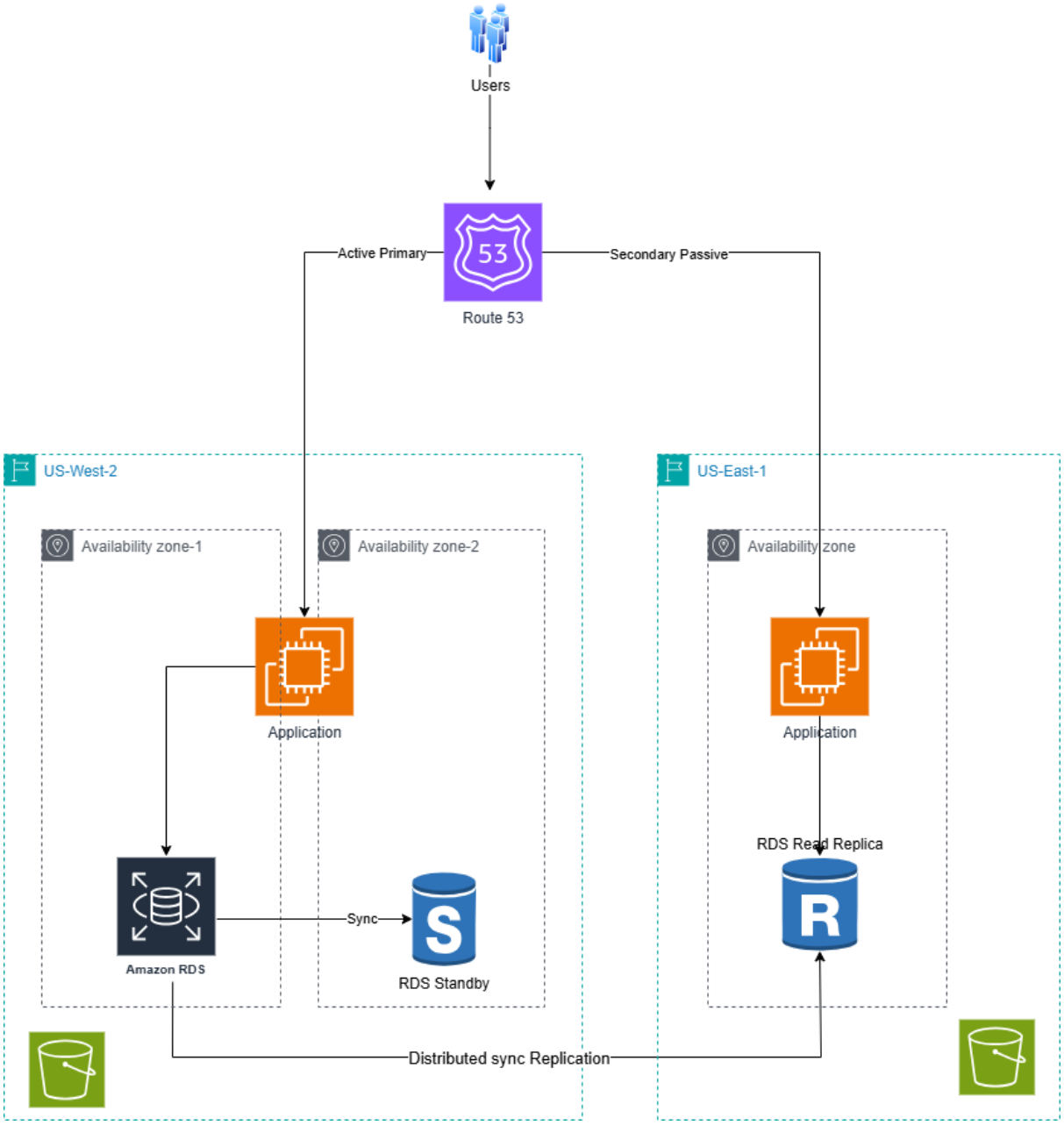
This PoC will serve as a foundation for a fully scalable and production-ready backup and disaster recovery solution.

3. Architecture Diagram

The architecture for this PoC will include:

- AWS Backup for automated backups across RDS, EC2, and EFS.
- S3 Replication for cross-region storage redundancy.

- RDS Multi-AZ for high availability.
- Route 53 failover policies for seamless DNS-based failover.
- AWS Elastic Disaster Recovery (AWS DRS) for rapid recovery of workloads.
- AWS Resilience Hub for disaster recovery testing and validation.



4. Basic Explanation about PoC

The PoC aims to demonstrate the effectiveness of AWS Backup and Disaster Recovery in ensuring business continuity. This includes:

- Automating backup processes to prevent data loss.

- Implementing cross-region replication to provide redundancy and resilience.
- Setting up failover mechanisms for high availability.
- Testing disaster recovery scenarios to validate response plans.

A. Why Use AWS Backup & Disaster Recovery?

AWS Backup & Disaster Recovery provides:

- Automated Backup Management – Centralized backup across AWS services.
- Cross-Region Replication – Ensures data is available even if a region fails.
- Cost Efficiency – Storage lifecycle policies reduce backup costs.
- Security & Compliance – Data encryption, IAM access controls, and compliance with regulations.
- Minimal Downtime – Failover mechanisms ensure quick recovery.
- Scalability – Works for small and large enterprises alike.

B. Business Benefits

Implementing AWS Backup & Disaster Recovery helps businesses:

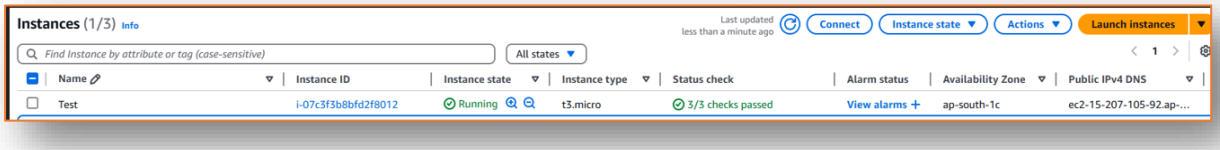
- Ensure Business Continuity – Reduces downtime and data loss.
- Improve Disaster Preparedness – DR testing ensures readiness for failures.
- Enhance Security & Compliance – Protects data with encryption and access controls.
- Reduce Operational Overhead – Automates backup processes, reducing manual effort.
- Optimize Costs – Efficiently manages storage and backup expenses through lifecycle policies.

5. let's setup configuration steps.

1. Log in to AWS Console

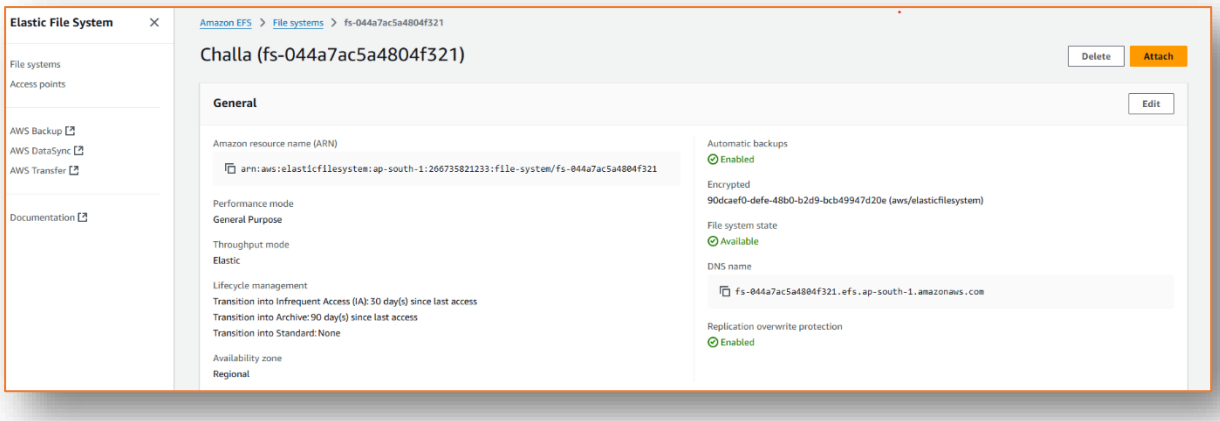
1. Go to the AWS Management Console.
2. Search for EC2 in the search bar and click EC2.

3. Launch EC2 Instance
    1. Go to AWS Console > EC2 > Launch instance.
4. Choose an AMI
    1. Select Amazon Linux 2 or Ubuntu 22.04 LTS.
5. Select Instance Type
    1. Choose t2. micro (Free Tier Eligible).
6. Configure Instance
    1. Set VPC, subnet, and enable public IP (if needed).
7. Set Storage & Tags
    1. Default 8 GiB (gp2 SSD), add tag (e.g., Name: My-EC2-Instance).
8. Configure Security Group
    1. Allow SSH (22, My IP) and HTTP (80, Anywhere).
9. Launch & Connect
    1. Create/download key pair, launch instance, and connect via SSH.



2. Create EFS and Attach to EC2

1. Go to the AWS Management Console
2. Open AWS Console.
3. Navigate to Amazon EFS.
4. Click on "Create file system"
5. Choose the VPC where your EC2 instance is located.
6. Select Availability Zones and Subnets.
7. Choose General Purpose Performance Mode.
8. Enable Encryption (optional).
9. Configure Security Group
10. Ensure the security group allows NFS (port 2049) for the EC2 instance.
11. Click "Create" and wait for the file system to be available.



12. Once created, open an EC2 instance.
13. Attach the EFS to EC2:
    - Connect to EC2 using SSH.
    - Install NFS client:

```
$ sudo yum install -y amazon-efs-utils
```

    - Create a Mount Directory

```
$ sudo mkdir -p /mnt/efs
```

    - Mount the EFS:

```
$ sudo mount -t efs fs-0e9a26376c6033b5a.efs.ap-south-1.amazonaws.com:/ /mnt/efs
```

    - Verify the mount:

```
$ df -h
```

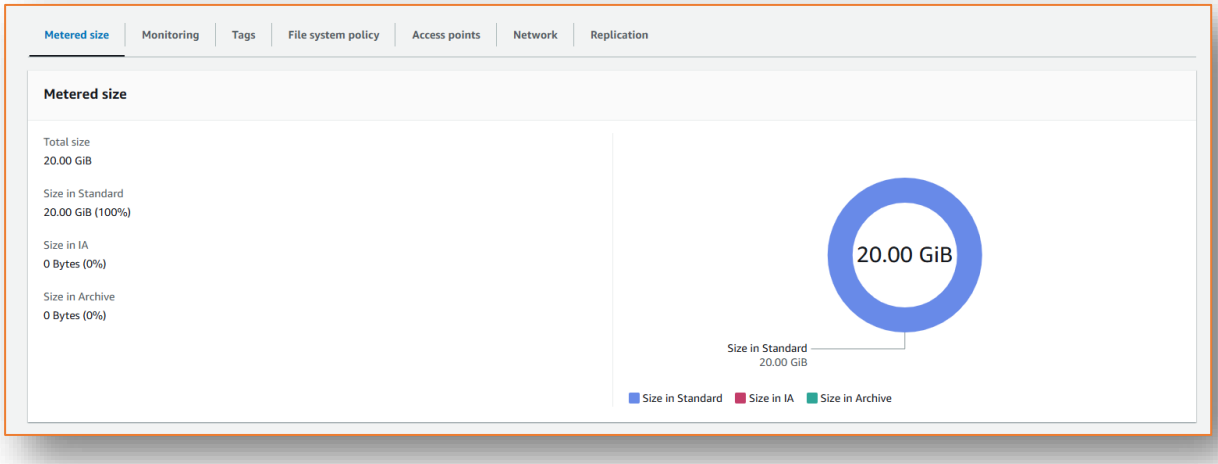
```
[root@ip-172-31-19-116 ~]# df -hT
Filesystem      Type      Size  Used Avail Use% Mounted on
devtmpfs        devtmpfs  455M     0  455M   0% /dev
tmpfs           tmpfs     464M     0  464M   0% /dev/shm
tmpfs           tmpfs     464M  452K  464M   1% /run
tmpfs           tmpfs     464M     0  464M   0% /sys/fs/cgroup
/dev/nvme0n1p1  xfs        20G  8.1G   12G  41% /
tmpfs           tmpfs      93M     0   93M   0% /run/user/0
127.0.0.1:/     nfs4      8.0E   20G  8.0E   1% /mnt/efs
```

    - If you see files and folders.

```
$ ls -lh /mnt/efs
```

```
[root@ip-172-31-19-116 ~]# ls -lh /mnt/efs
total 21G
drwxr-xr-x 4 root root 6.0K Feb 25 05:15 home
drwxr-xr-x 6 root root 6.0K Feb 25 06:06 Shop-Verse
-rw-r--r-- 1 root root  20G Feb 25 05:30 testfile
[root@ip-172-31-19-116 ~]# du -sh /mnt/efs
21G     /mnt/efs
[root@ip-172-31-19-116 ~]#
```

- Once you have attached ec2 instance with EFS then it will reflect the EFS like below



- you cannot see the AWS ec2 instance UI you can only see the console EFS attached or not.

3. AWS RDS Configuration Steps

1 Go to AWS RDS Console

- Sign in to AWS Console.
- Navigate to RDS → Click Create database.

2 Choose Database Engine

- Select MySQL, PostgreSQL, SQL Server, or Oracle.
- Choose the latest version.

3 Set Instance Details

- Enter DB Instance Name (e.g., mydatabase).
- Set Master Username & Password.

4 Select Instance Type & Storage

- Choose Instance Type (e.g., t3.micro for free tier).

- Set Storage Size (EX 20GB).

5 Configure Network & Security

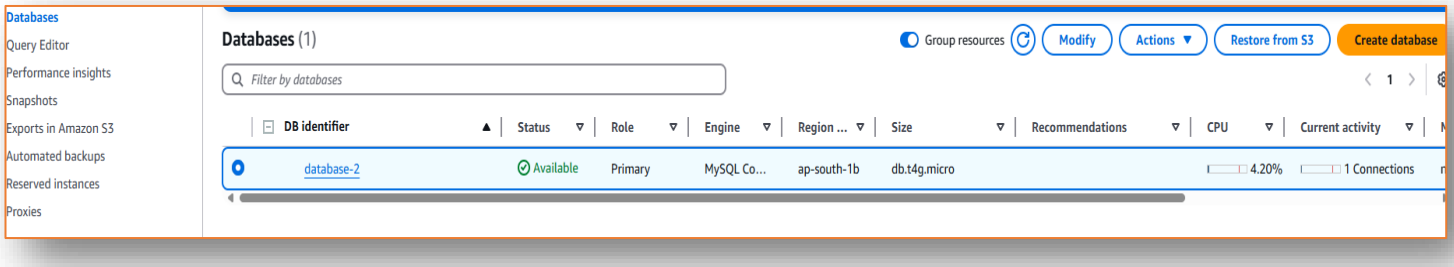
- Select VPC (default or custom).
- Public Access: Yes (for external access) / No (for private access).
- Security Group: Allow DB Port (3306 for MySQL).
- Enable Encryption at Rest (recommended).

6 Set Backup & Monitoring

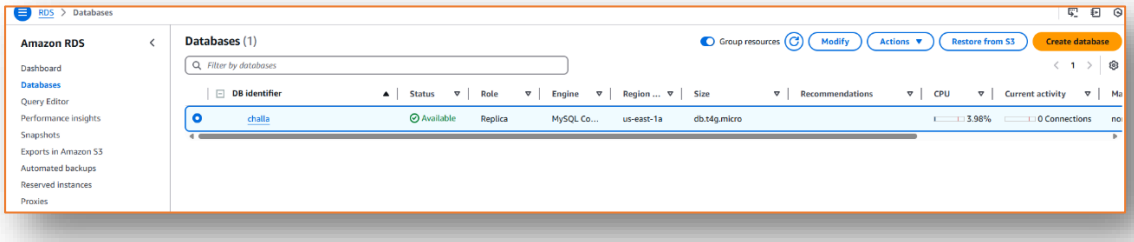
- Choose Backup Retention Period (EX 7 days).
- Enable Performance Insights (optional).
- Enable Multi-AZ Deployment for high availability.

7 Review & Create

- Verify all settings.
- Click Create database → Wait for it to launch.



With Multi-AZ Database with Read Replication server.



Automated EC2, RDS, EFS backup with Different Region.

1 Create Backup Vault

- Go to AWS Backup → Backup Vaults → Click Create vault.
- Choose a name and enable Cross-Region Replication.

2 Set Up Backup Plan

- Go to AWS Backup → Backup Plans → Click Create Plan.
- Define schedule, retention period, and destination region.

3 Assign Resources

- Add EC2, RDS, and EFS to the backup plan using tags or resource IDs.

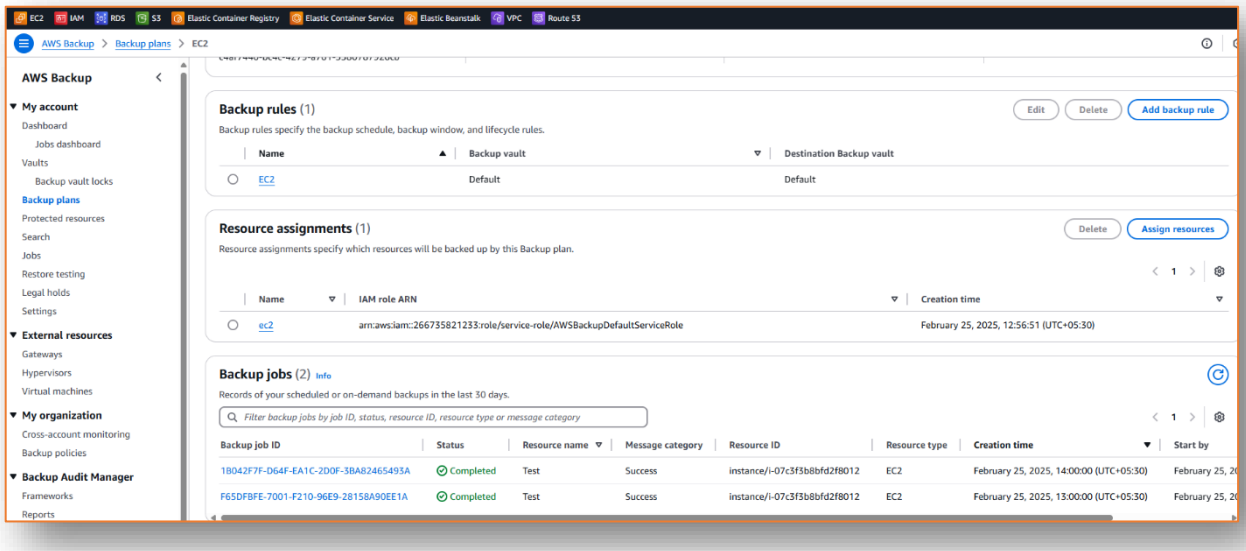
4 Configure IAM Permissions

- Ensure AWS Backup service role has access for cross-region replication.

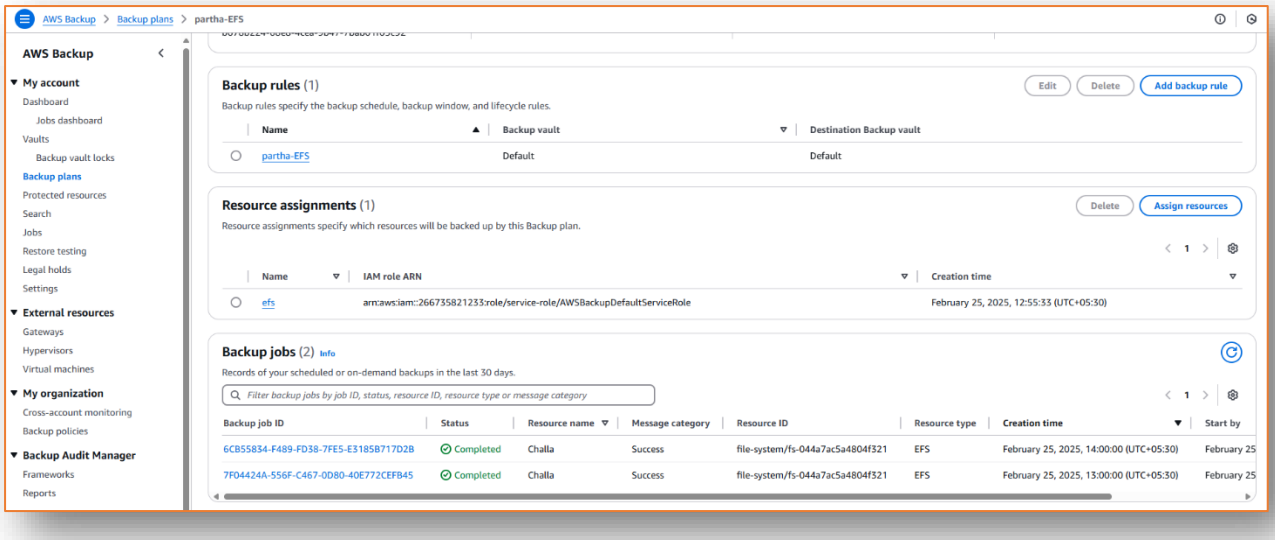
5 Monitor & Restore

- Check backup jobs in AWS Backup Console.
- Restore backups to the desired region when needed.

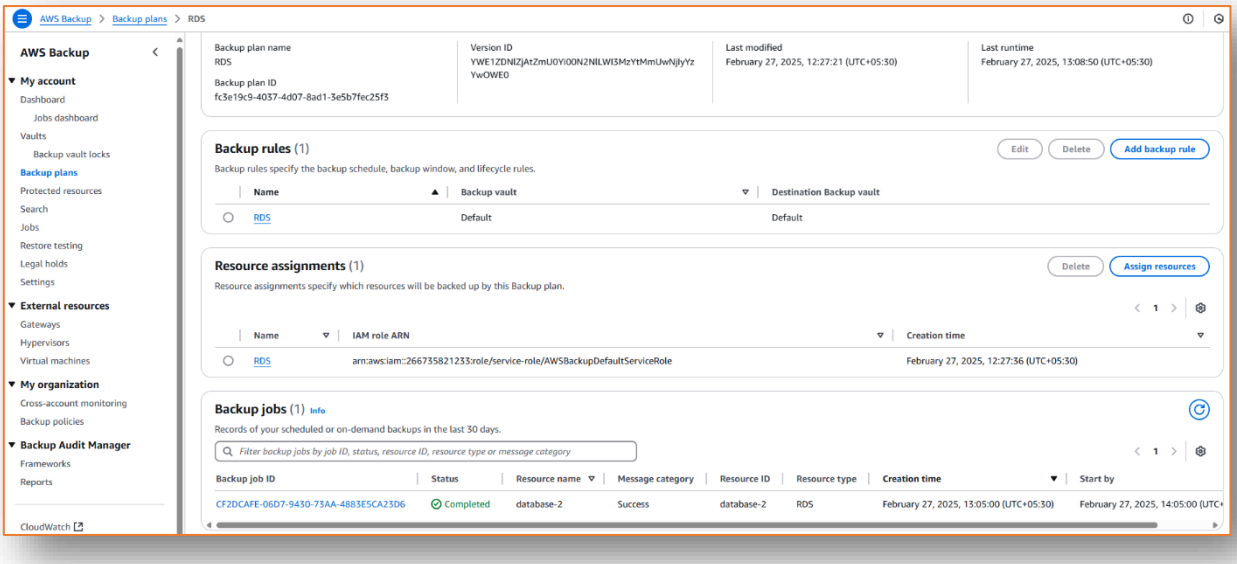
EC2 instance backup.



EFS Backups



RDS Backups.



Whenever you need to be restored the server at the time first you need IAM role aws backup permissions

IAM role Permissions.

Open IAM Management Console

1. Log in to the AWS Management Console.
2. In the search bar, type IAM and click on IAM.
3. On the left menu, click Roles.

Step 2: Create a New IAM Role

4. Click Create role.
5. Under Trusted entity type, select AWS Service.
6. In the Use case section, select AWS Backup.
7. Click Next.

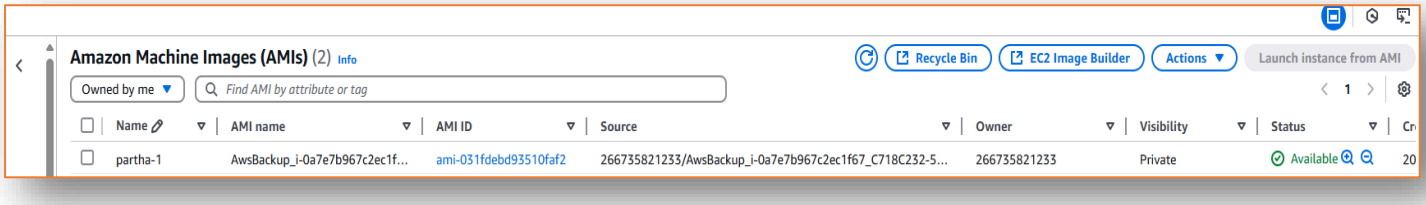
Step 3: Attach Required Policies

8. In the Permissions policies section, search for the following policies:
    - AWSBackupServiceRolePolicyForBackup
    - AWSBackupServiceRolePolicyForRestores
9. Select both policies by checking their boxes.
10. Click Next.

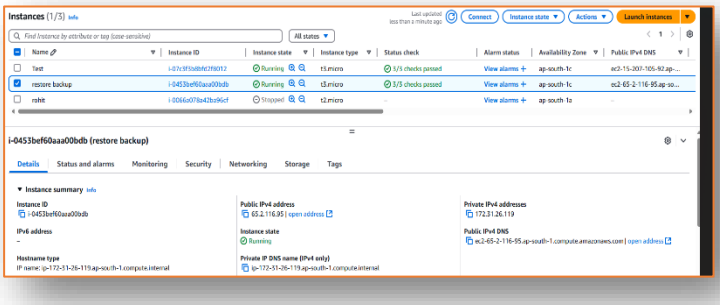
Step 4: Name and Create the Role

11. Enter a Role name: AWSBackupRestoreRole.
12. (Optional) Add a description:
    - "This role allows AWS Backup to restore EC2 and RDS backups."
13. Click Create role.

EC2 instance the cross region used aws backup.



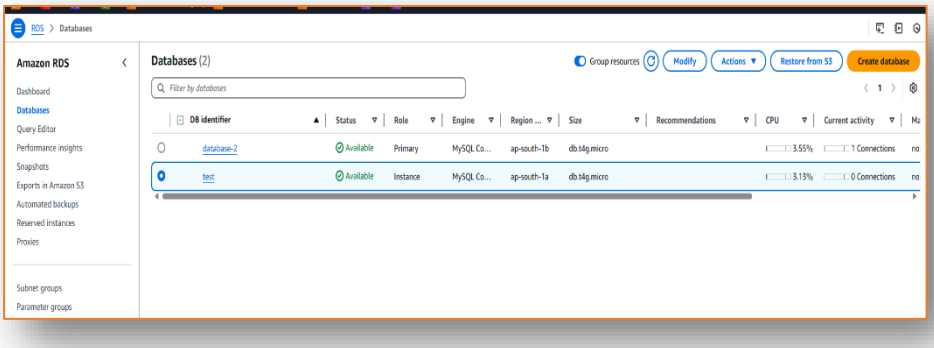
Ec2 instance restore the within region.



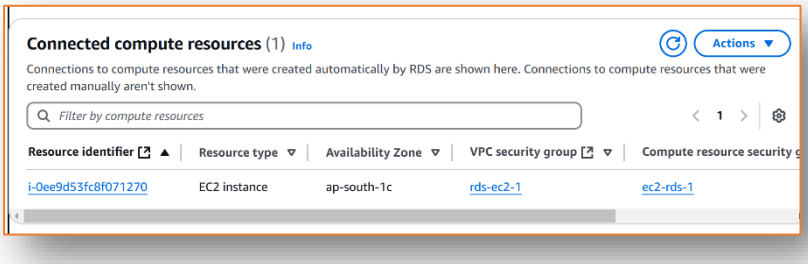
EFS restored the AWS Backup.

```
sh-4.2$ sudo -i
[root@ip-172-31-19-116 ~]# ls -lh /mnt/efs
total 21G
drwxr-xr-x 5 root root 6.0K Feb 25 06:33 aws-backup-restore_2025-02-25T09-19-22-789155545Z
drwxr-xr-x 4 root root 6.0K Feb 25 05:15 home
drwxr-xr-x 6 root root 6.0K Feb 25 06:06 Shop-Verse
-rw-r--r-- 1 root root  20G Feb 25 05:30 testfile
[root@ip-172-31-19-116 ~]#
```

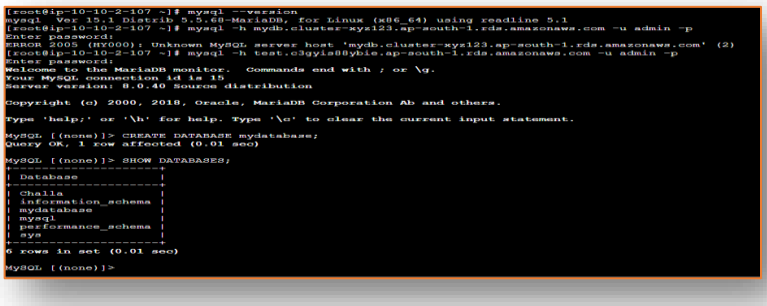
RDS Restore the AWS Backup.



AWS EC2 instance with RDS connection



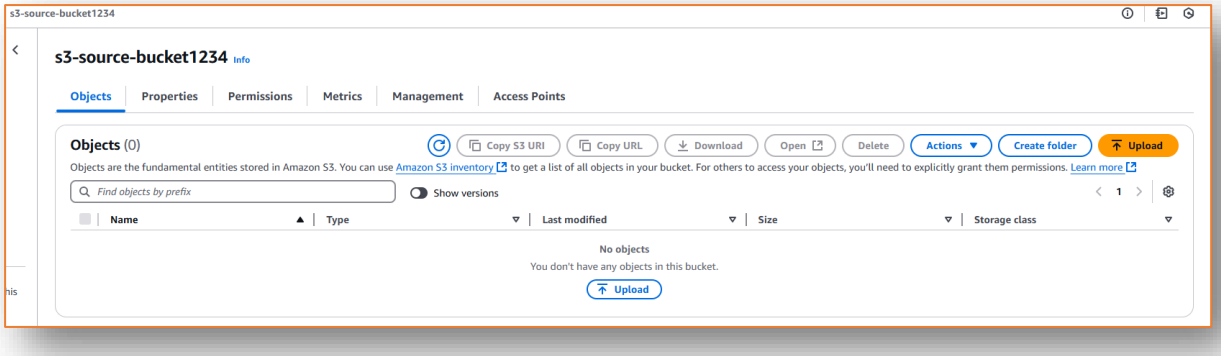
Whenever you need to connect your database, you can install my sql and you need to be a database endpoint and username and password



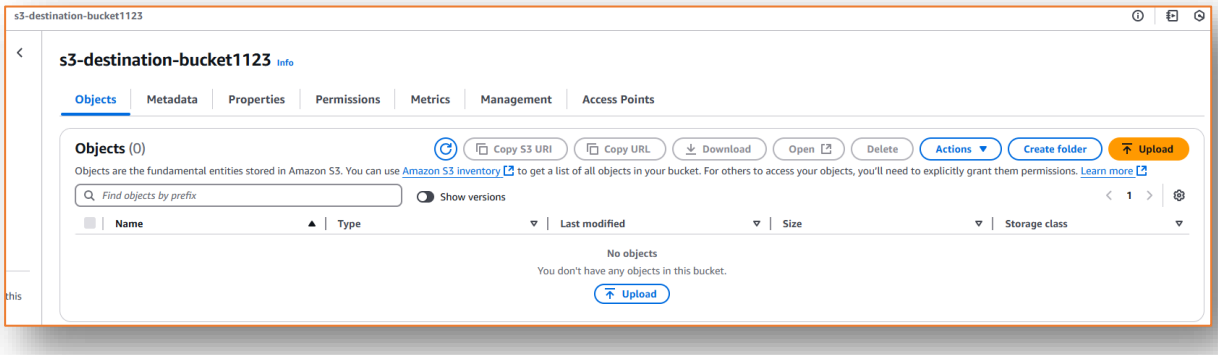
S3 Cross-Region Replication (CRR) for DR

Step 1: Create Source and Destination Buckets

1. Go to S3 Console → AWS S3
2. Click Create Bucket.
    - Bucket Name: source-bucket-name
    - Region: Choose the Primary Region (EX ap-south-1).
    - Enable Versioning (required for CRR).
    - Click Create Bucket.

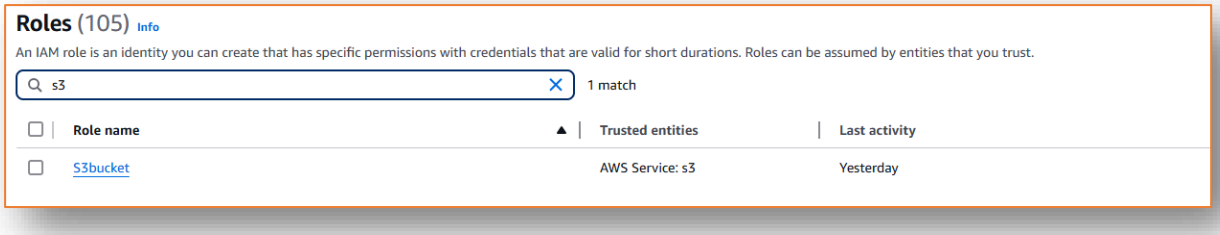


3. Repeat the steps to create a Destination Bucket in a different region.
    - Bucket Name: destination-bucket-name
    - Region: Choose a different AWS Region (EX us-west-1).
    - Enable Versioning.
    - Click Create Bucket.



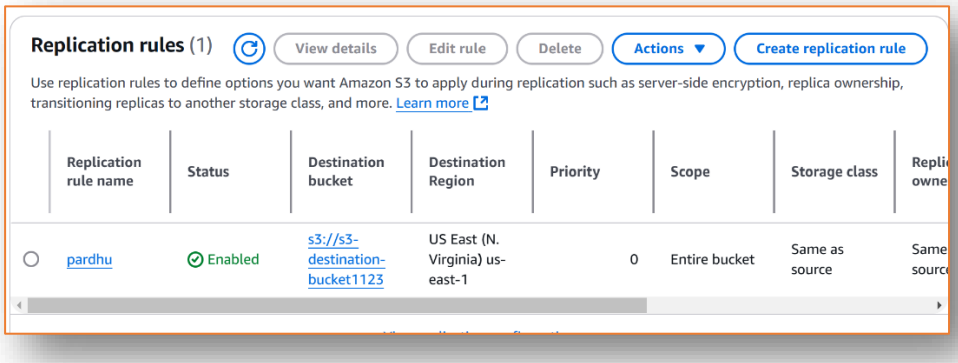
Step 2: Create an IAM Role for Replication

1. Go to IAM Console.
2. Click Roles → Create Role.
3. Select AWS Service → Choose S3.
4. Attach the policy:
    - Use the AmazonS3FullAccess policy OR
    - Create a custom policy with replication permissions.
5. Name the Role: **S3ReplicationRole**.
6. Click Create Role.



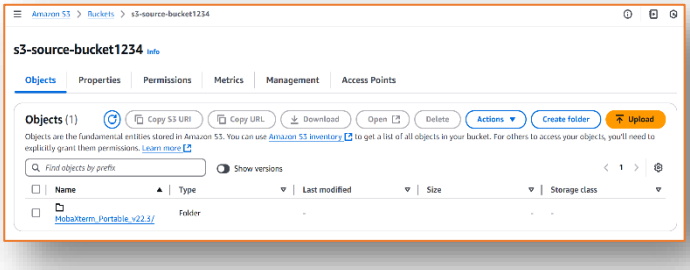
Step 3: Configure Replication Rule

1. Go to Source Bucket → Management → Replication Rules.
2. Click Create Replication Rule.
3. Name the rule (e.g., CRR-Rule).
4. Select Entire Bucket or Specific Prefix.
5. Choose Destination Bucket.
6. Select the IAM Role (S3ReplicationRole).
7. Click Save.

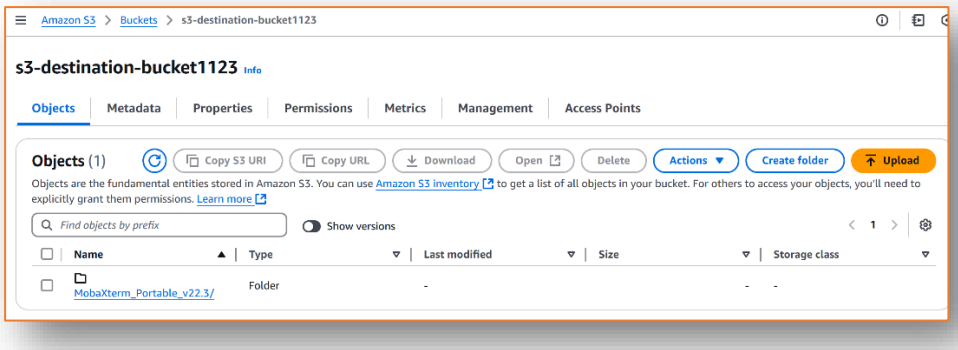


Whenever I have upload file in source bucket it is automatically to replicate file to store destination bucket.

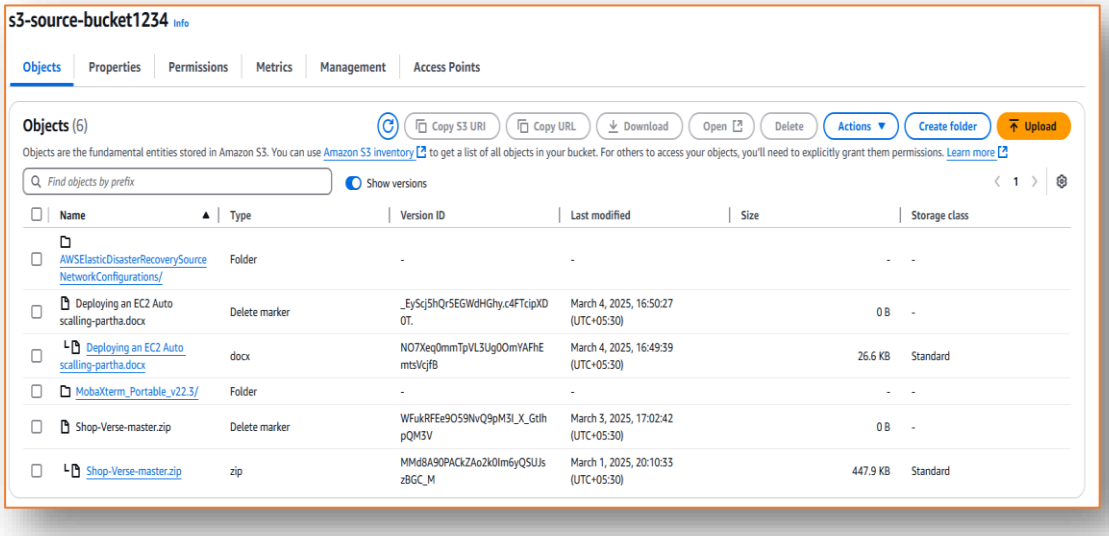
This source bucket



This is destination bucket.



If you have deleted the file in source file and destination file, then you can retrieve the file like s3 bucket versioning.

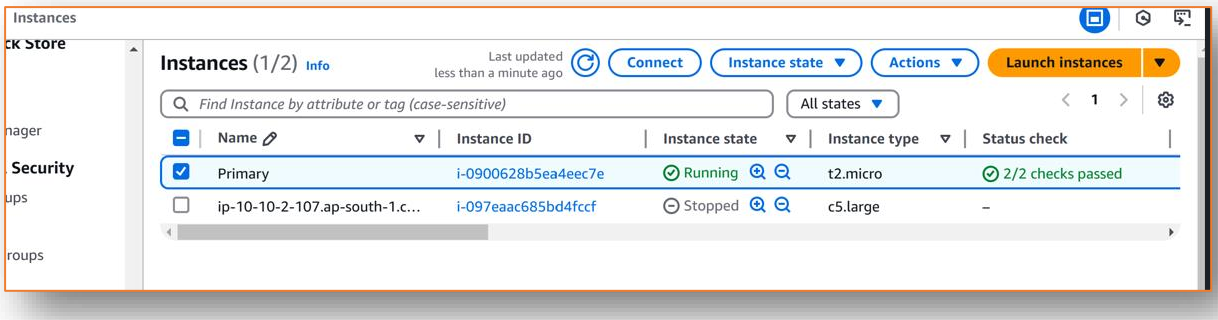


Failover Mechanisms Using Route 53 with Load Balancer (Mumbai & Singapore Regions).

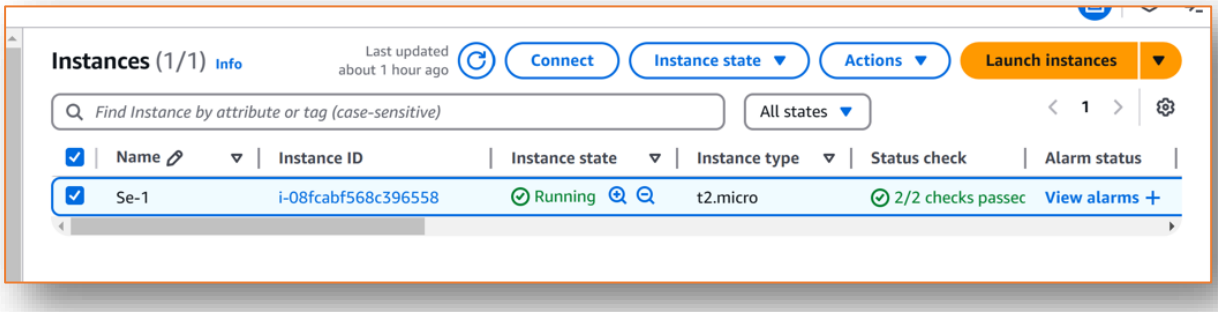
Before you should start this task, you can create it.

- **Mumbai Region (ap-south-1)** and **Singapore Region (ap-southeast-1)**
- Two EC2 instances in each region (acting as web servers) and with attached RDS database.
- Application Load Balancers (ALB) set up in each region with target group.
- A domain name managed in Route 53.

Mumbai region Ap-south-1 EC2 instance primary with web application small.



Singapore region Ap-southeast-1 EC2 instance secondary web application small

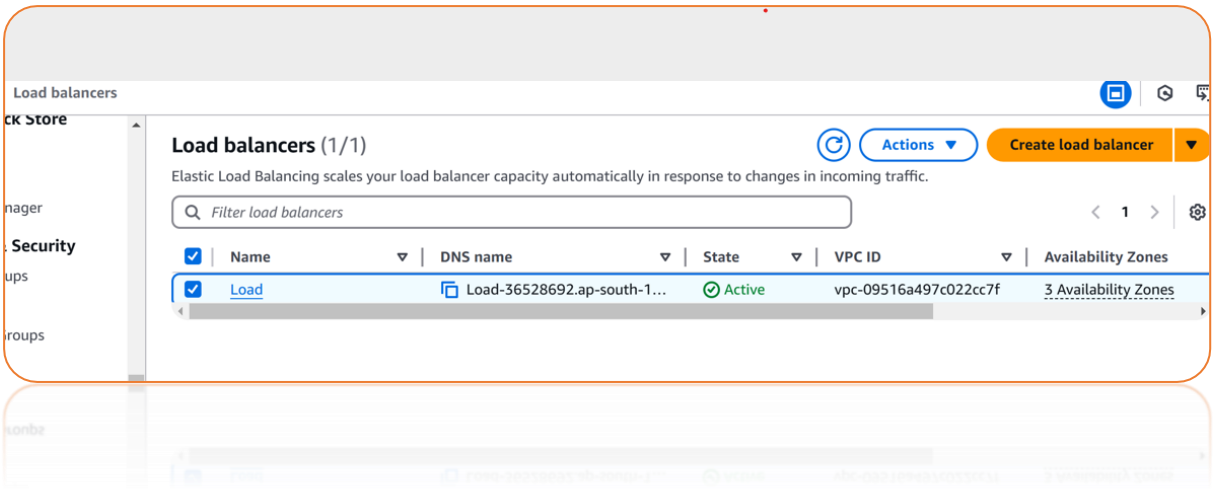


Set Up Elastic Load Balancers (ALB)

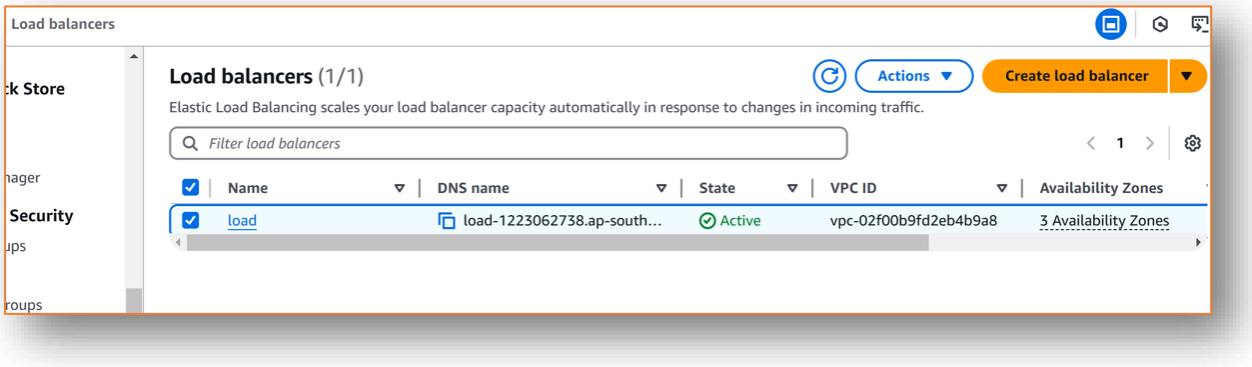
Step 1: Create Load Balancers in Mumbai and Singapore Regions

1. Open the **AWS Management Console → EC2 → Load Balancers**.
2. Click **Create Load Balancer**.
3. Choose **Application Load Balancer (ALB)**.
4. Configure as follows:
    - **Name:** Mumbai-ALB (for Mumbai), Singapore-ALB (for Singapore)
    - **Scheme:** Internet-facing
    - **VPC & Subnets:** Choose appropriate VPC and public subnets.
    - **Security Group:** Allow HTTP/HTTPS traffic.
5. Click **Create Load Balancer**.

This is Mumbai region load balancer.



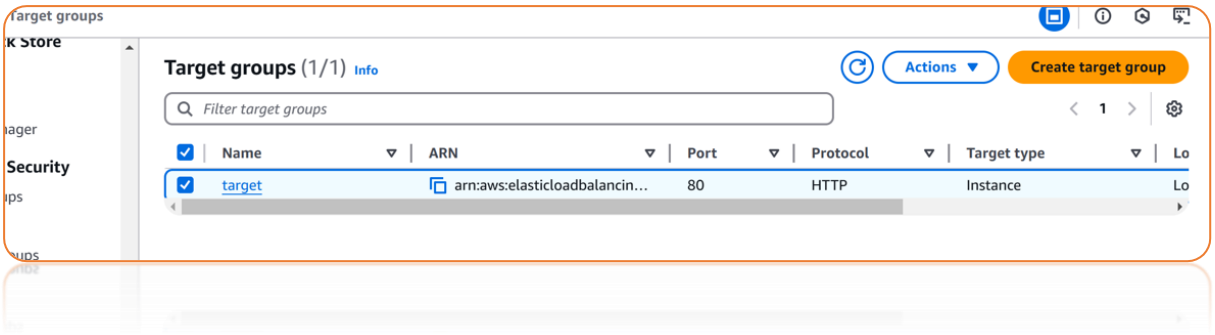
This is Singapore region load balancer.



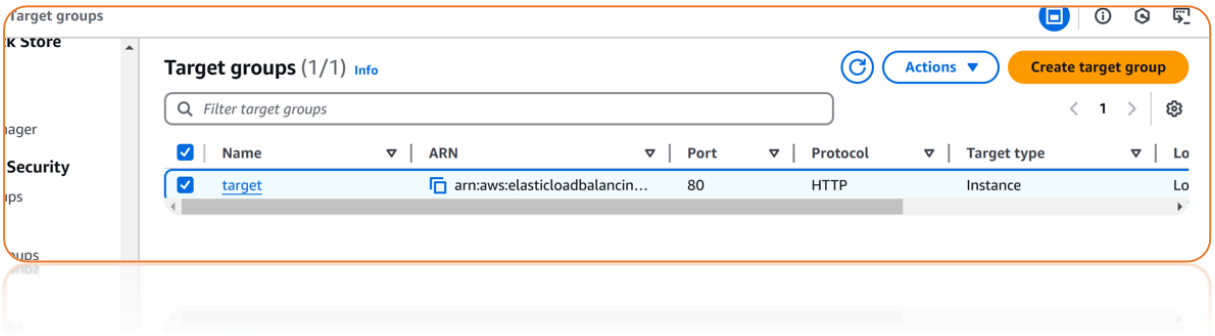
Step 2: Create Target Groups and Register EC2 Instances

1. Navigate to **Target Groups**.
2. Click **Create Target Group**.
3. Choose **Instances** and configure:
    - **Name:** Mumbai-TG, Singapore-TG
    - **Protocol:** HTTP (or HTTPS if SSL configured)
    - **VPC:** Select the corresponding VPC.
4. Click **Create** and then **Register Targets** by selecting EC2 instances.
5. Click **Save**.

Target group Mumbai region.

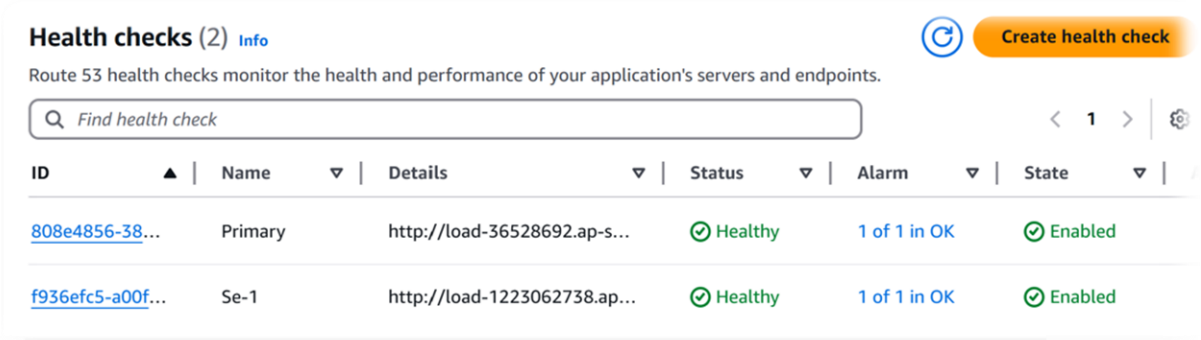


Target group Singapore region.



Step 3: Set Up Health Checks

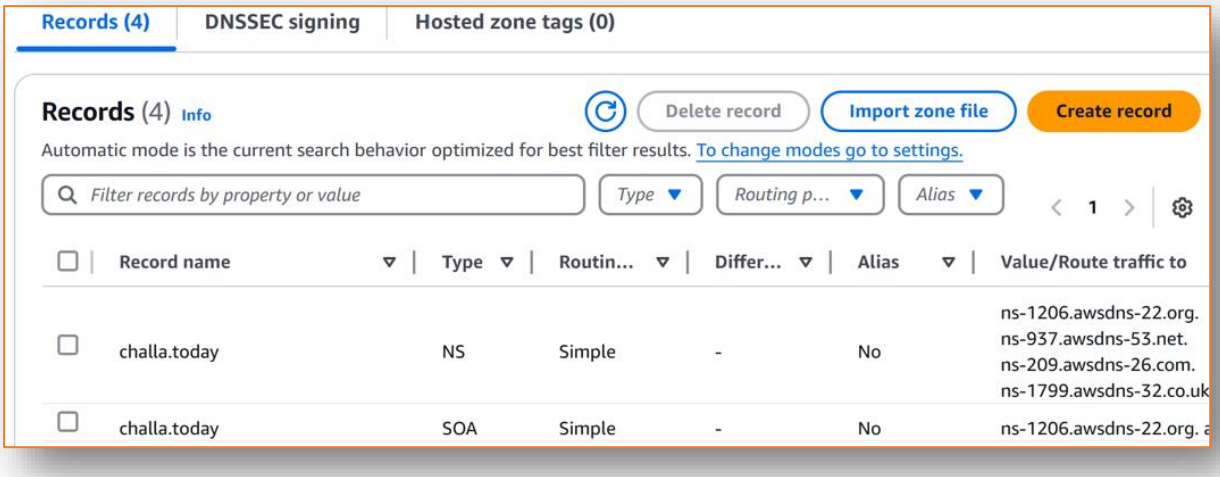
1. Open **Route 53** → **Health Checks** → **Create Health Check**.
2. Configure the health check:
    - **Name:** primary,Secondary
    - **Endpoint:** Mumbai ALB DNS, Singapore ALB DNS
    - **Protocol:** HTTP
    - **Request Path:** / (or a health check path like /health)
    - **Failure Threshold:** 3 (default)
    - Click **Create**.
    - With create aws CloudWatch alarm



3. Configure Route 53 for Failover

Step 1: Create a Hosted Zone

1. Open **Route 53 Console** → **Hosted Zones**.
2. Click **Create Hosted Zone**.
3. Enter your domain name (Challa.Today).
4. Click **Create**.



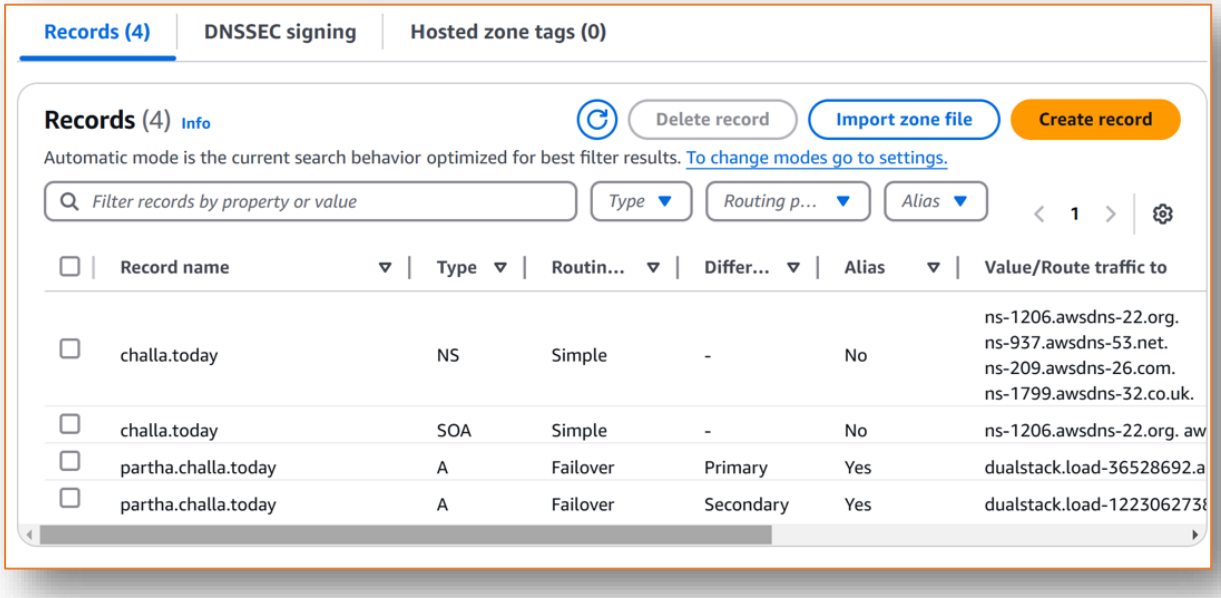
Step 2: Create Failover DNS Records

1. Open the **Hosted Zone** → Click **Create Record**.
2. **Create the Primary Record (Mumbai):**
    - **Name:** Partha.challa.today
    - **Type:** A
    - **Alias:** Yes
    - **Value:** Mumbai ALB DNS Name
    - **Routing Policy:** Failover (Primary)
    - **Health Check:** Attach a new health check (explained below)
    - Click **Create Record**.

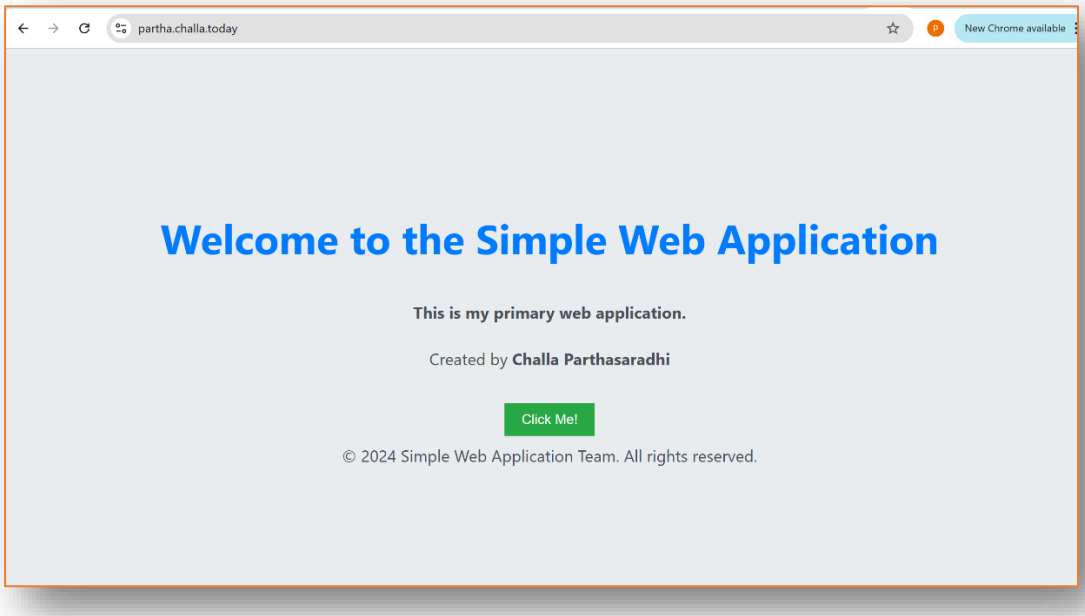
3. **Create the Secondary Record (Singapore):**

- **Name:** partha.challa.today
- **Type:** A
- **Alias:** Yes
- **Value:** Singapore ALB DNS Name
- **Routing Policy:** Failover (Secondary)
- **Health Check:** Attach a new health check (explained below)
- Click **Create Record**.

- Primary and secondary hosted zones.

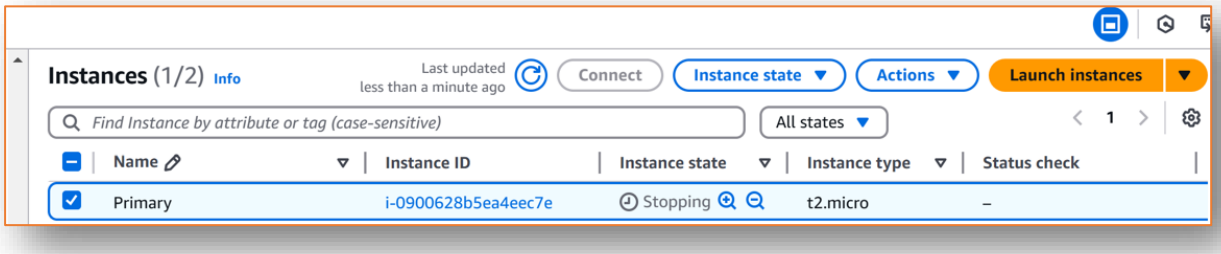


- You can copy the domain and open the browser then pasted it would be open website and you can able to see the web site.

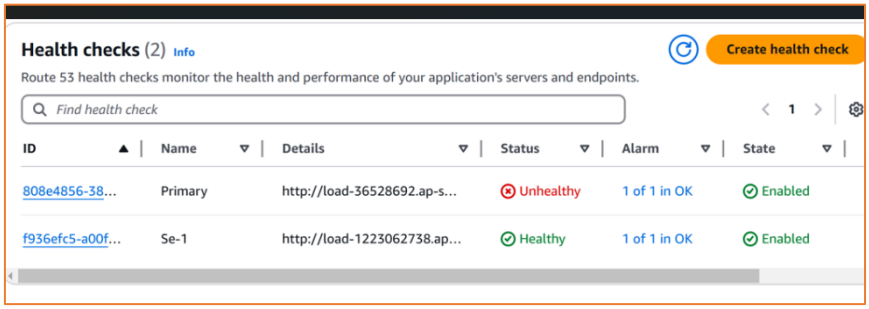


- You can test the traffic redirect or not you can check.
- First You need to be ec2 instance stop the server then you can open my secondary web application

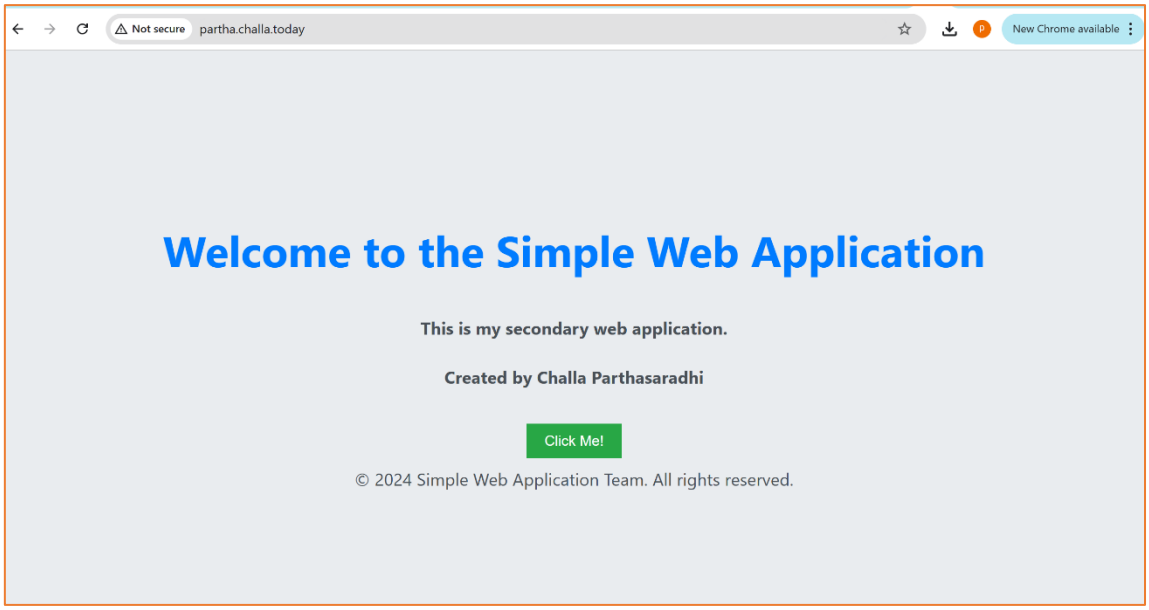
Ec2 instance stop the server.



- You can check route53 health check primary health is unhealth because of server is stop

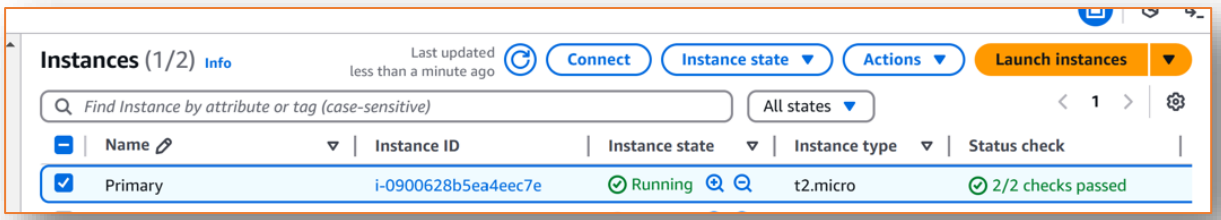


- You can check web site is open secondary web application because of route traffic redirect.

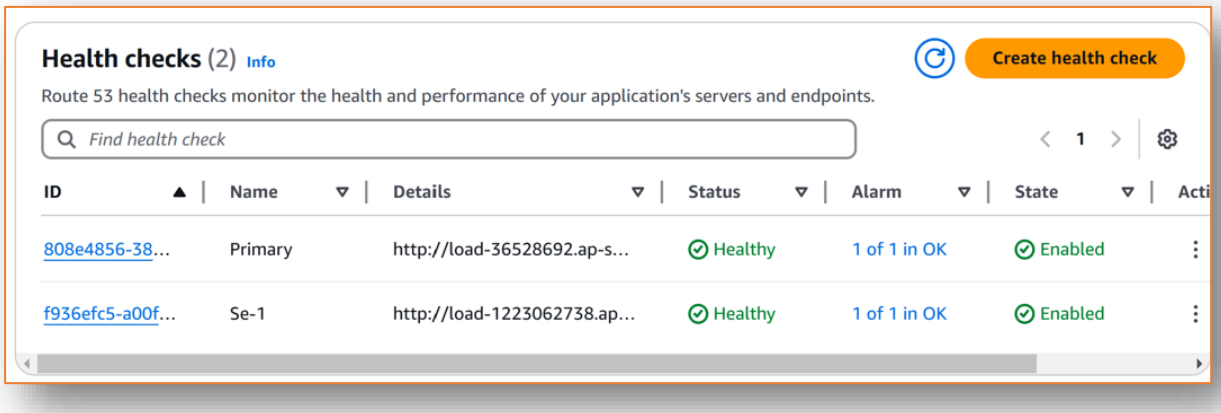


- Once you tested that all then you can up the server then you can access the primary web application.

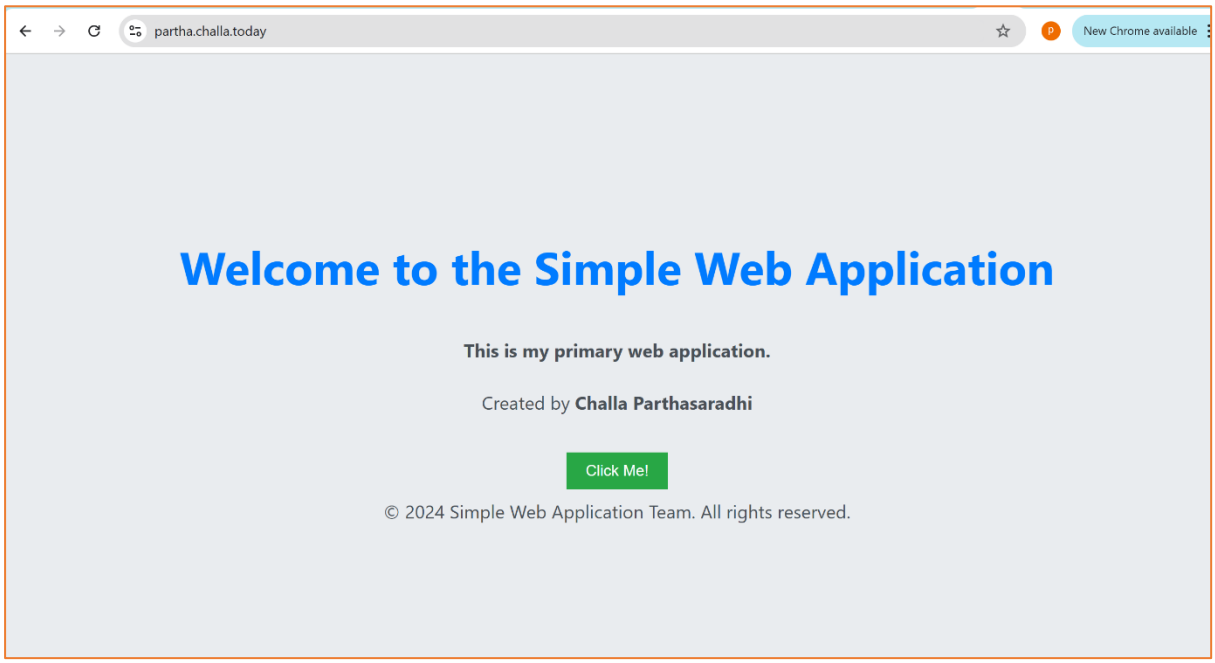
EC2 instance.



- And you should check health check it primary ec2 instance health check is health is good.



- Once primary application server is active then route traffic redirect you can see the primary web application.



AWS Elastic Disaster Recovery (AWS DRS) for EC2 Instances within a region available zone and cross region available zone.

- when you have need disaster recovery you can make them to do it and to create one role to attach the ec2 instance which server you need you can for attached then you need to install the Disaster recovery agent

IAM role Permissions.

Open IAM Management Console

1. Log in to the AWS Management Console.
2. In the search bar, type IAM and click on IAM.
3. In the left menu, click on Roles.

2 Create a New IAM Role

4. Click Create role.
5. Under Trusted entity type, select AWS Service.
6. In the Use case section, select Elastic Disaster Recovery (drs.amazonaws.com).
7. Click Next.

3 Attach Required Policies

8. In the Permissions policies section, search for the following policies:
    - AWSElasticDisasterRecoveryReplicationServerPolicy

- AWSElasticDisasterRecoveryConversionServerPolicy
- AWSElasticDisasterRecoveryEc2InstancePolicy
- AmazonSSMManagedInstanceCore
- AmazonS3FullAccess
- AmazonSSMFullAccess
- AWSElasticDisasterRecoveryAgentPolicy
- AWSElasticDisasterRecoveryFailbackPolicy
- AWSElasticDisasterRecoveryRecoveryInstancePolicy

9. Select all three policies by checking their boxes.
10. Click Next.

4 Name and Create the Role

11. Enter a Role name: AWS-DisasterRecovery.
12. (Optional) Add a description:

- "This role allows AWS Elastic Disaster Recovery to replicate and restore EC2 instances."

13. Click Create role.

- Next you can connect to the EC2 instance and to install the AWS DR agent nothing, but it is replicate the ec2 server where you need to take the server within available zone or cross region
- You need to enter the security group custom TCP port number 1500, and https and http add

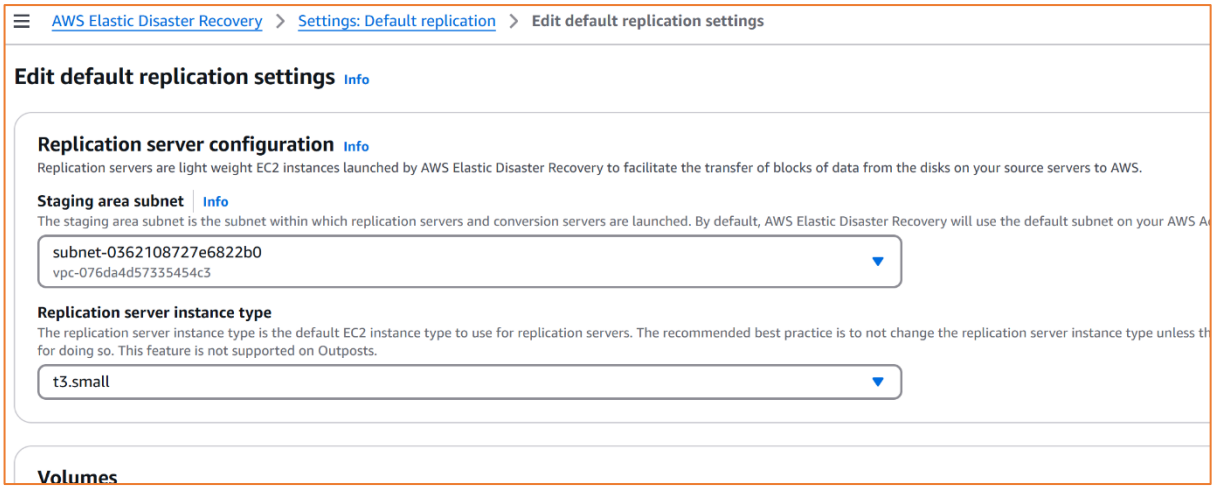
AWS DR installation commands follow the AWS document.

https://docs.aws.amazon.com/drs/latest/userguide/linux-agent.html

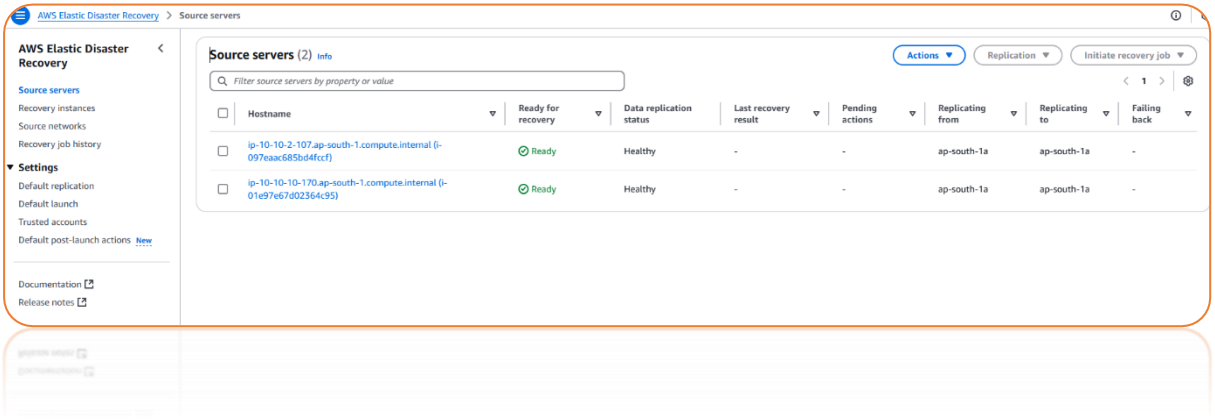
Configure Replication source server (Mumbai Region, Different AZ)

Configure Replication Settings

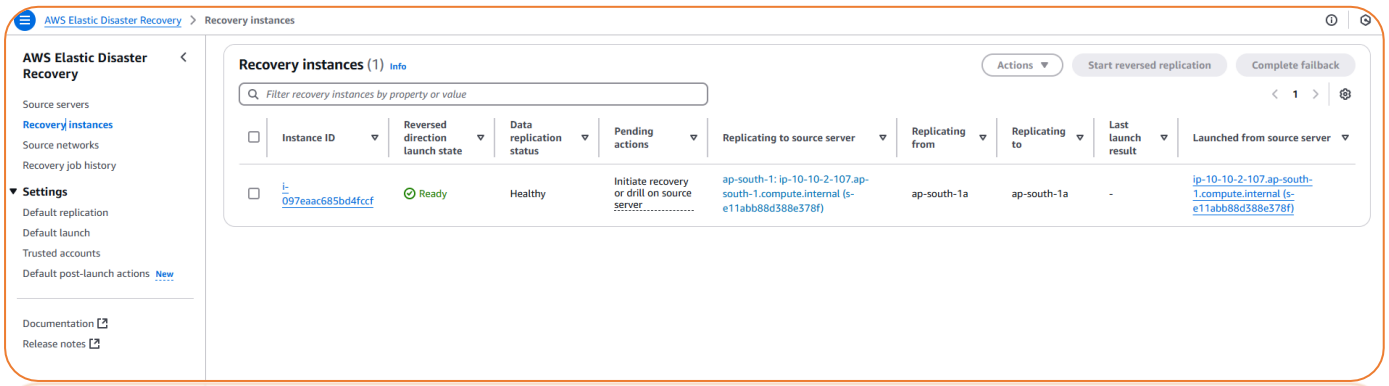
- Select the **source server** in AWS DRS.
- Click **Edit replication settings**.
- Choose the **same AWS region** but a **different availability zone (AZ)** for disaster recovery.
- Configure:
  - **Replication subnet** (must be in the target AZ).
  - **Security group** (ensure it allows necessary traffic).
  - **IAM role** (grant necessary permissions for replication).
- Click **Save changes**.



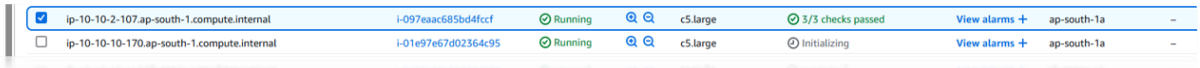
This is my AWS Disaster recovery Source server



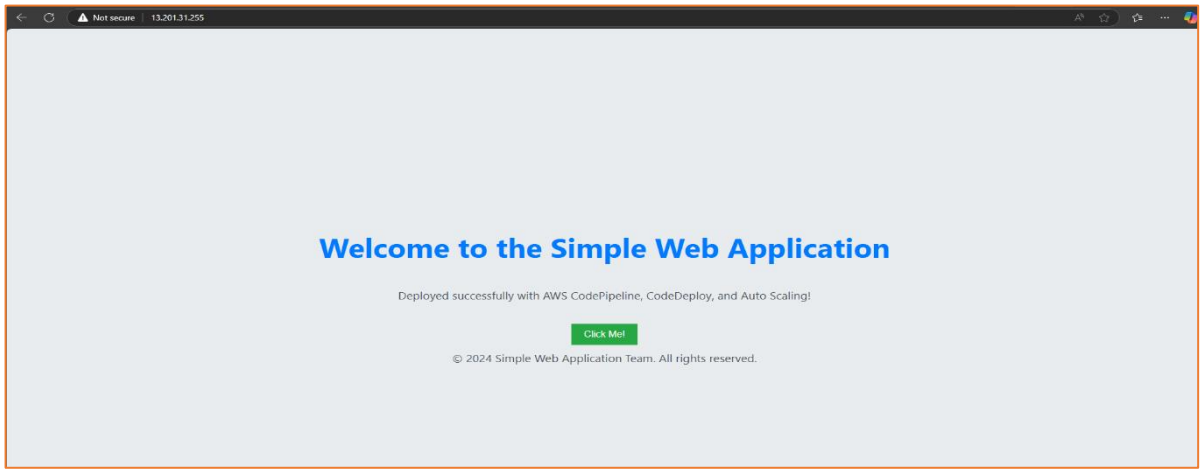
This is recovery instance.



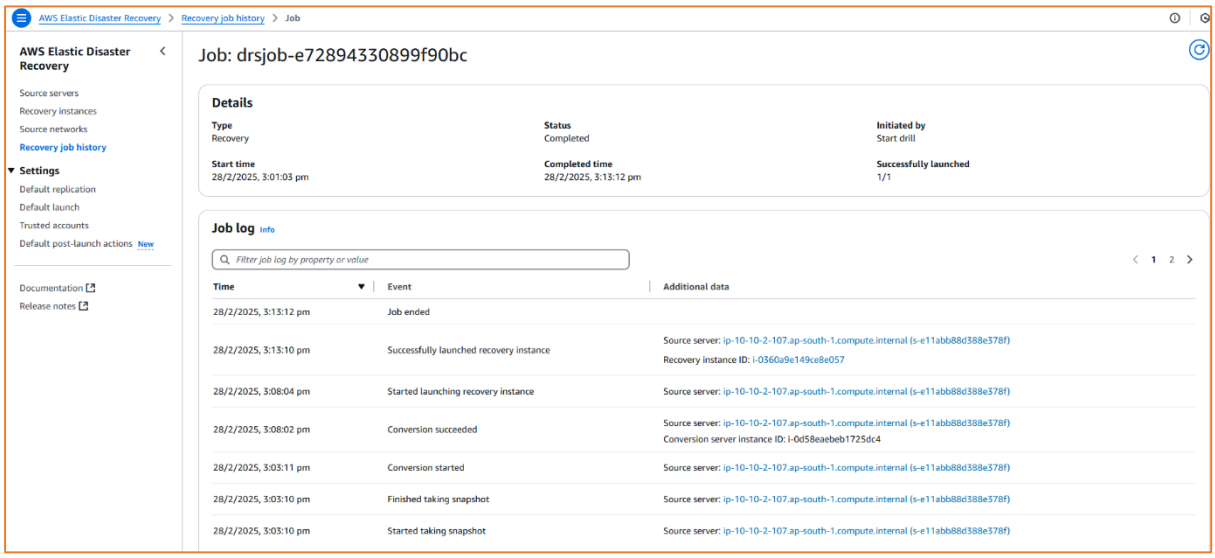
This is my recovery Ec2 instance for testing.



You can copy the ec2 instance Ip address then you can open it browser and pasted it you can access the web application.



If you need to check history of recovery you can check, go to the recovery history then click on.

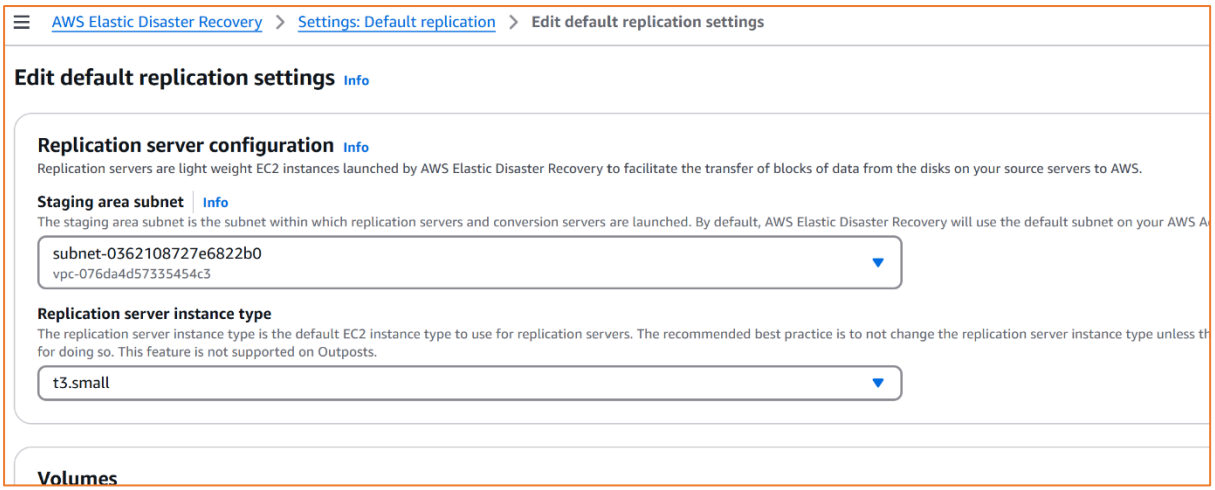


Cross region Configure Replication Singapore region targeting (Different AZ).

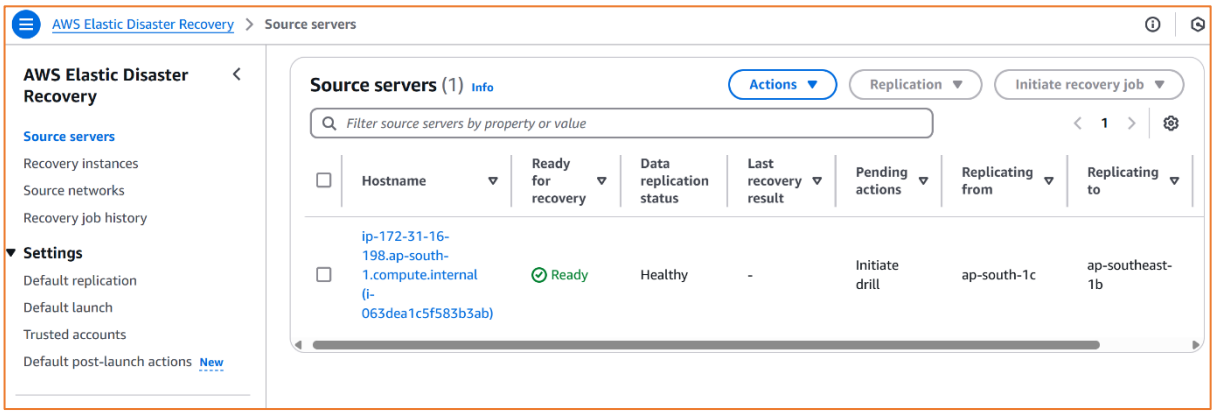
Configure Replication Settings

- Select the **source server** in AWS DRS.
- Click **Edit replication settings**.
- Choose the **same AWS region** but a **different availability zone (AZ)** for disaster recovery.
- Configure:
    - **Replication subnet** (must be in the target AZ).
    - **Security group** (ensure it allows necessary traffic).
    - **IAM role** (grant necessary permissions for replication).

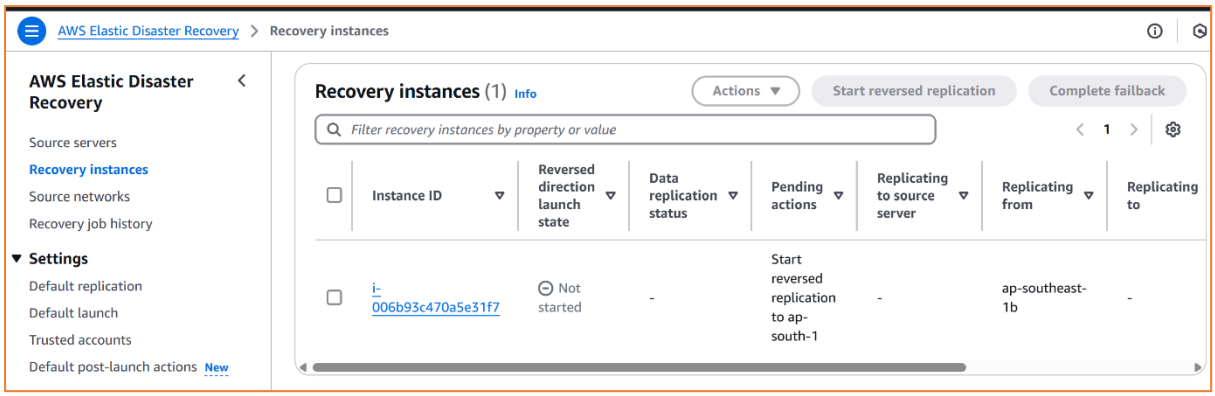
- Click **Save changes**.



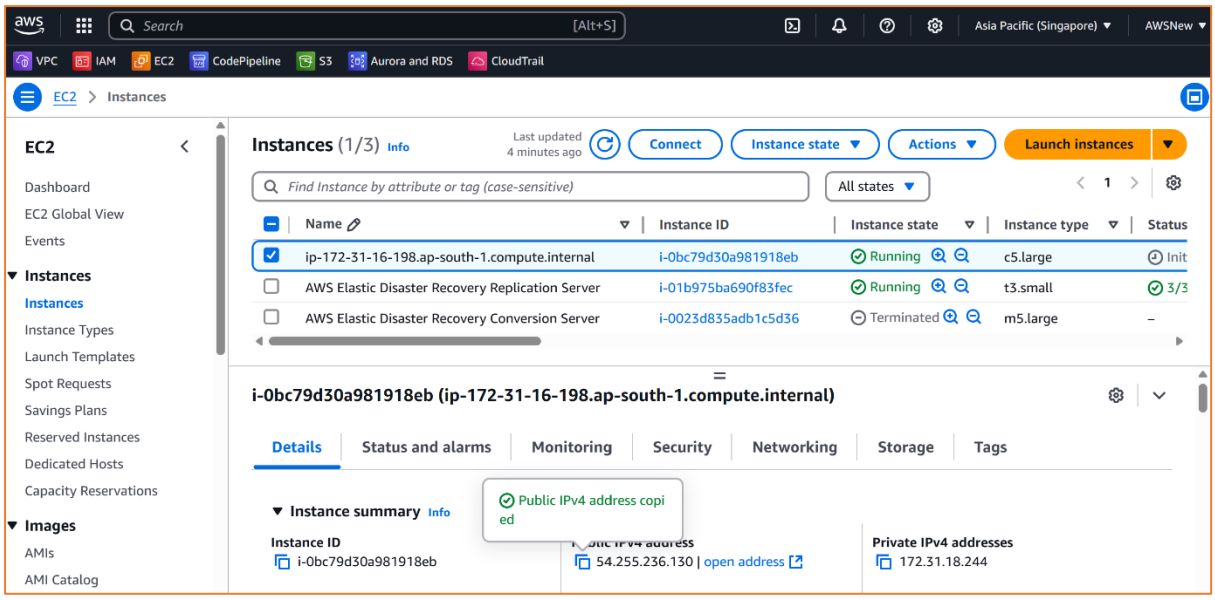
This is my AWS Disaster recovery Source servermumbai region to replicate from ap-south-1 to replicating Singapore southeast-1



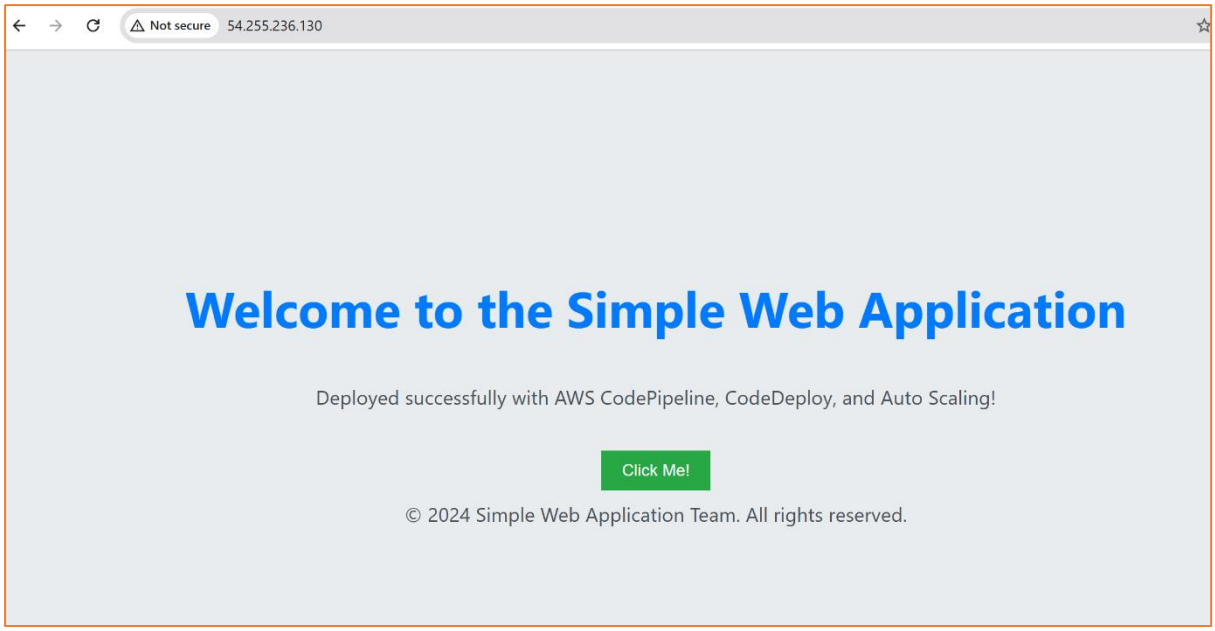
This is recovery instance.



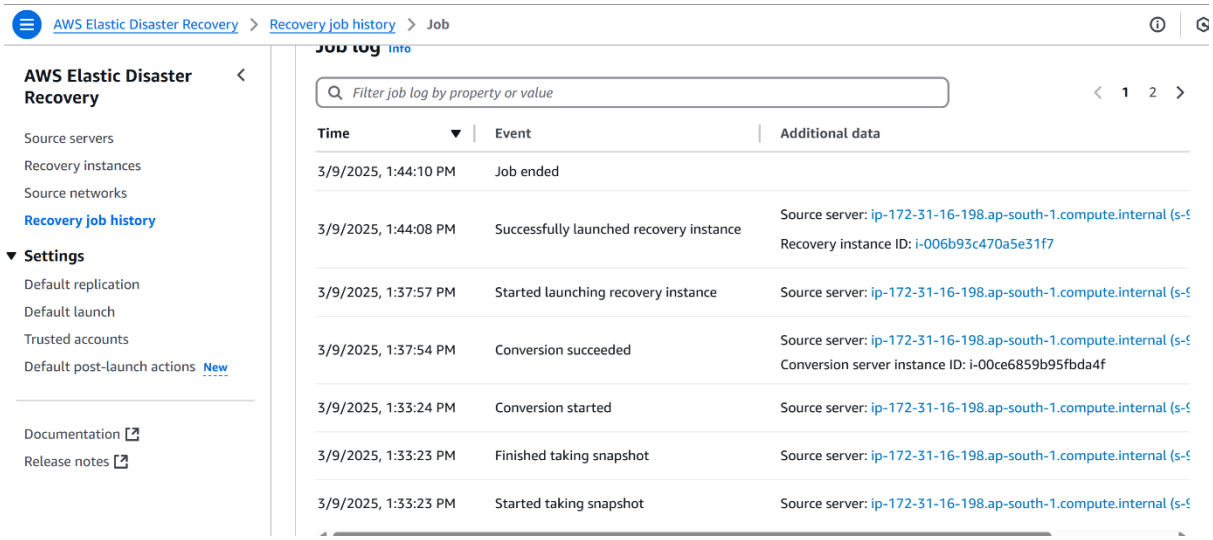
This is my recovery Ec2 instance for testing.



You can copy the ec2 instance Ip address then you can open it browser and pasted it you can access the web application.



If you need to check history of recovery you can check, go to the recovery history then click on.



AWS Resilience Hub.

first you need to create an AWS Resource group tags you need to tag the one ec2 or load balancer whatever you need server you need to tag that one then only you can be able to see the EC2 or ALB. AWS resilience hub.

Open IAM Management Console

1. Log in to **AWS Console**.
2. Search for **IAM** and open it.
3. Click **Roles** from the left menu.
4. Click **Create role**.

2 Set Up IAM Role for Resilience Hub

5. Under **Trusted entity type**, select **AWS Service**.
6. Under **Use case**, select **Resilience Hub**.
7. Click **Next**.

3 Attach Required Policies

8. In the **Permissions policies** section, search for and attach:
    - ☑ **AmazonResilienceHubFullAccess**
    - ☑ **AWSResilienceHubServiceRolePolicy**
9. Click **Next**.

4 Name and Create the Role

10. Enter **Role name**: AWS-ResilienceHub-Role.
11. (Optional) Add a **description**.
12. Click **Create role**.

Add EC2 to AWS Resilience Hub

1. Open **AWS Resilience Hub** from the AWS Console.
2. Click **Add application**.
3. Select **Import from AWS Resource Groups**.
4. Choose **EC2-Resilience-Group** (the group you created).
5. Click **Next**.
6. Review the selected resources and click **Create application**.

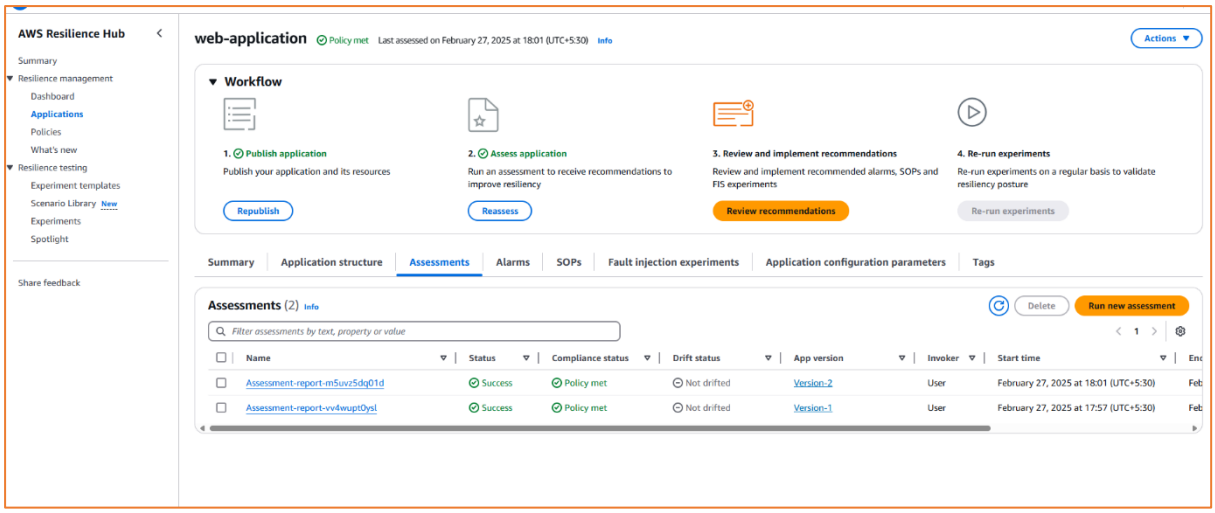
4 Define Resilience Policy

1. Click **Create resilience policy**.
2. Set **Recovery Time Objective (RTO)** and **Recovery Point Objective (RPO)**.
    - Example:
        - **RTO:** 5 minutes (Maximum downtime allowed).
        - **RPO:** 10 minutes (Data loss tolerance).
3. Select **Failure Scenarios**:
    - **EC2 instance**.
    - **Availability Zone**.
    - **Autoscaling**.
4. Click **Create policy**.

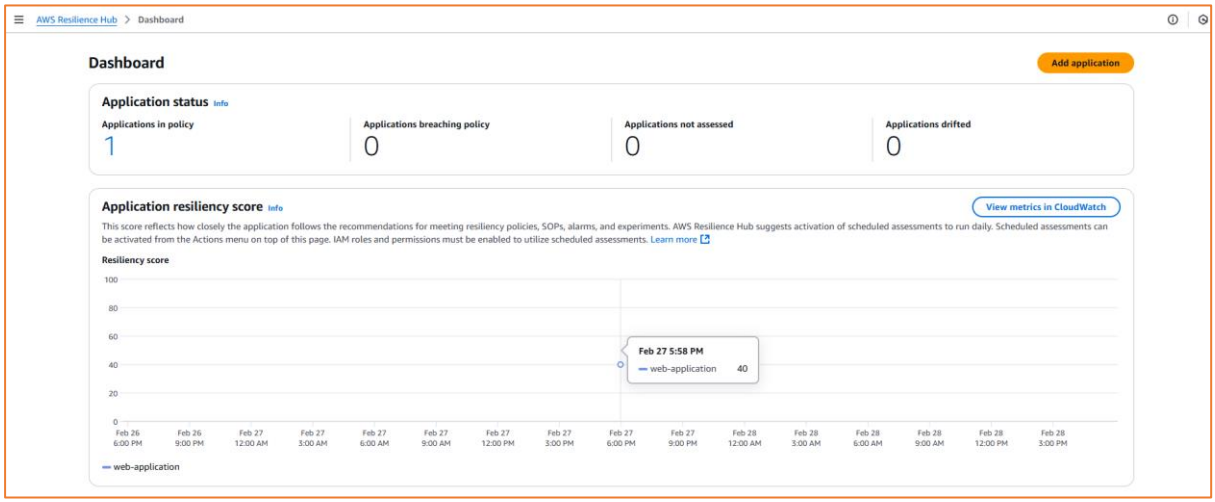
5 Run Resilience Assessment

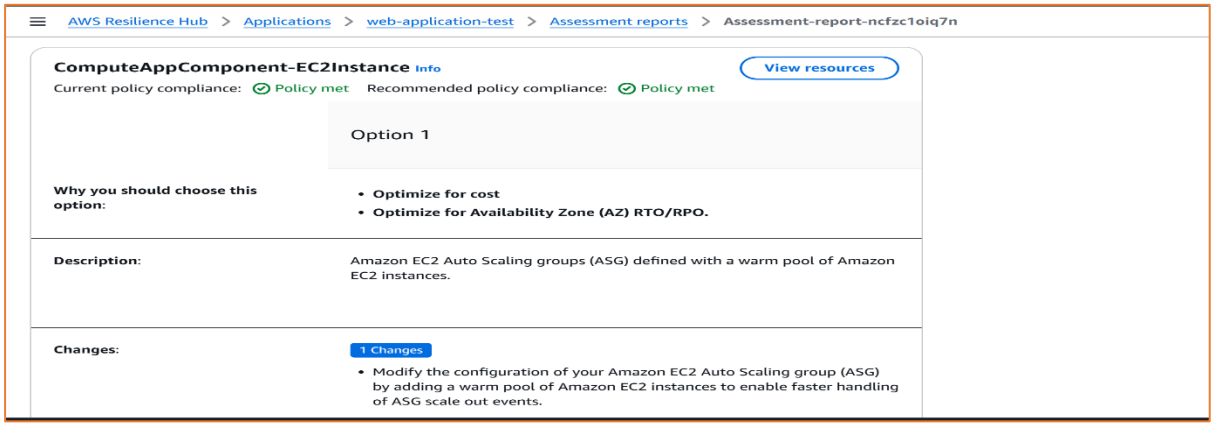
1. Open **AWS Resilience Hub**.
2. Select your **EC2 application**.
3. Click **Run assessment**.

This is web application used for resilience hub.



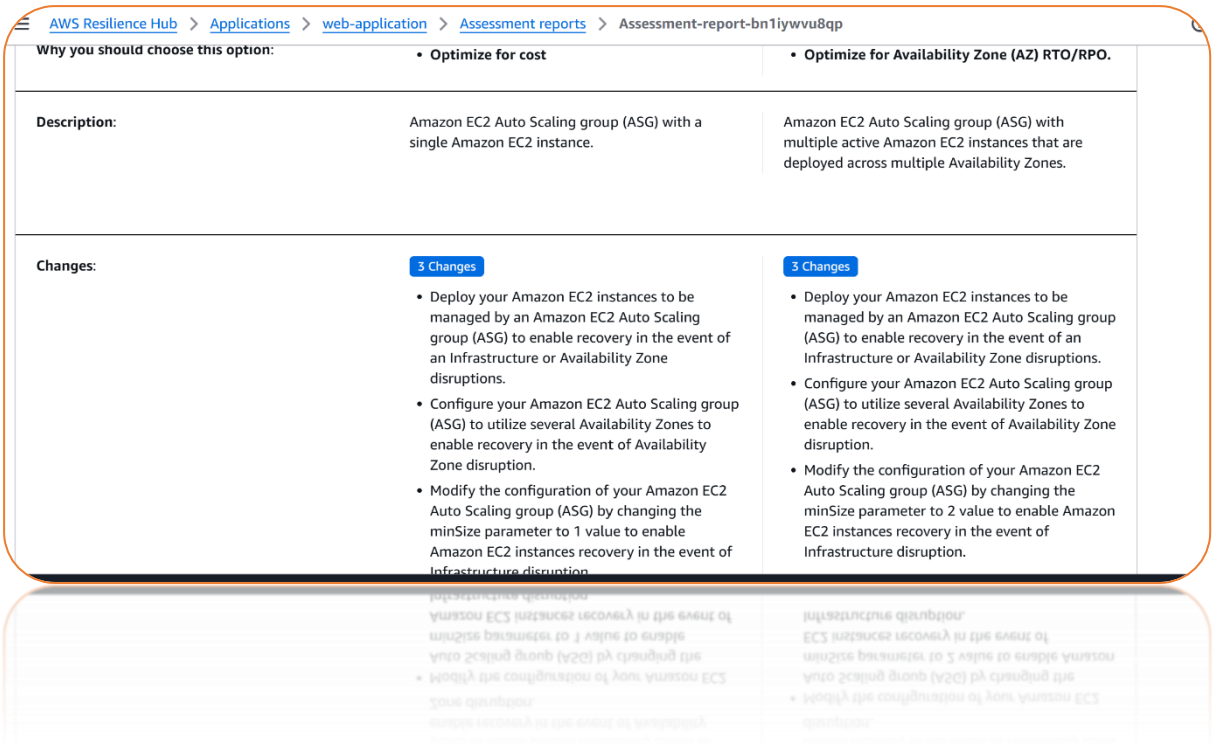
Application resiliency score.





Ec2 instance with autoscaling

Without autoscaling AWS resilience hub is given to implement autoscaling.



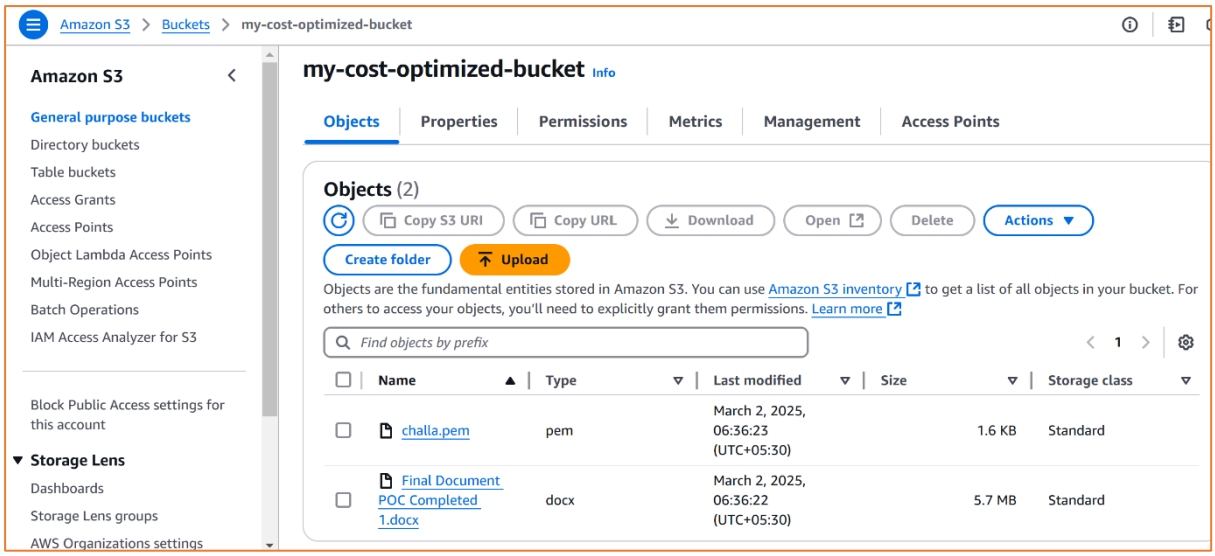
Costs Using Lifecycle Policies and AWS Storage Gateway.

Create an S3 Lifecycle Policy

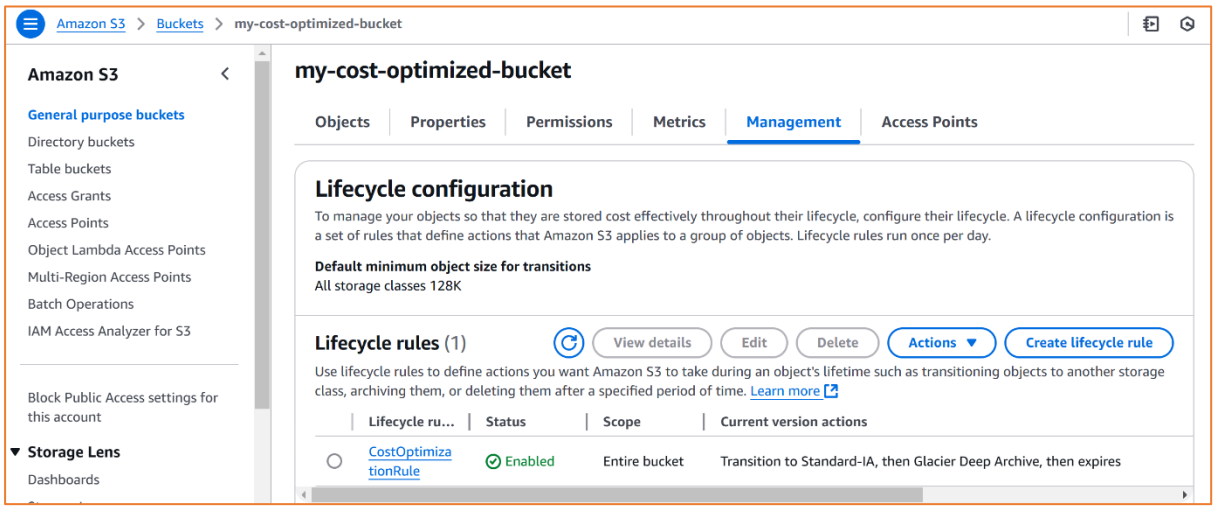
◈ Purpose:- Move old data to cheaper storage (S3 Glacier, Intelligent-Tiering).

1. Open AWS S3 Console.
2. Select the S3 bucket where your data is stored.
3. Click Management → Lifecycle Rules → Create lifecycle rule.
4. Enter a Rule Name (Cost-optimization-bucket).
5. Select Apply to all objects in the bucket.
6. Under Lifecycle rule actions, enable:
    - Move to Standard-IA after 30 days.
    - Move to Glacier after 90 days.
7. Click Create rule.

My bucket to add file with cost optimized. Whenever you add file to store the bucket and storage class like Standard-IA after 30 days you need to store the more days you can move the S3 Glacier there is option after that you can automatically deleted file.



To create Lifecycle Rules.

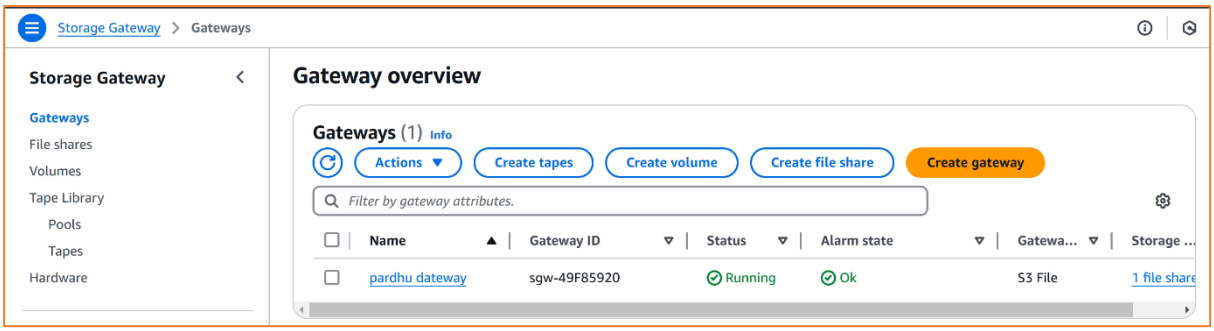


Configure AWS Storage Gateway

reducing expensive local storage costs.

1. Set Up AWS Storage Gateway

1. Open **AWS Storage Gateway Console**.
2. Click **Create gateway**.
3. Choose **File Gateway** (for file storage).
4. Select **Amazon S3 as the storage destination**.
5. Deploy the **Storage Gateway VM** (on **EC2)**
6. Connect the **gateway to AWS** using the activation key provided.



To lunch ec2 instance storage gateway

2. Attach S3 Bucket to Storage Gateway

1. Open **AWS Storage Gateway Console**.
2. Click **Create file share**.
3. Select your **S3 bucket**.
4. Set up **NFS** for access.
5. Click **Create file share**.

One you create file gateway then you need to attach the s3 bucket and given commands.

It should be connected to ec2 server then you need to give commands like mount s3 bucket, and you can create one file then you can replicate the s3 bucket store the file

This is my s3 bucket you can be able to see the file.