

AWS Backup & Disaster Recovery

Table of Contents

1. Objective
2. Project Scope
3. Architecture Diagram
4. Basic Explanation about PoC
 - Why Use AWS Backup & Disaster Recovery?
 - Business Benefits
5. Automating Backups using AWS Backup
 - AWS Backup Setup for RDS
 - AWS Backup Setup for EC2
 - AWS Backup Setup for EFS
6. Cross-Region Replication for Disaster Recovery
 - S3 Replication Configuration
 - RDS Multi-AZ Setup
7. Failover Mechanisms
 - Route 53 Failover Configuration
 - AWS Elastic Disaster Recovery (AWS DRS) Setup
8. Testing Recovery Scenarios
 - AWS Resilience Hub Setup
 - Disaster Recovery Simulation
 - Failover Testing & Validation
9. Optimizing Costs
 - Lifecycle Policies for Backups
 - AWS Storage Gateway Implementation
 - Cost Optimization Strategies
10. POC Daily Work Progress

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.

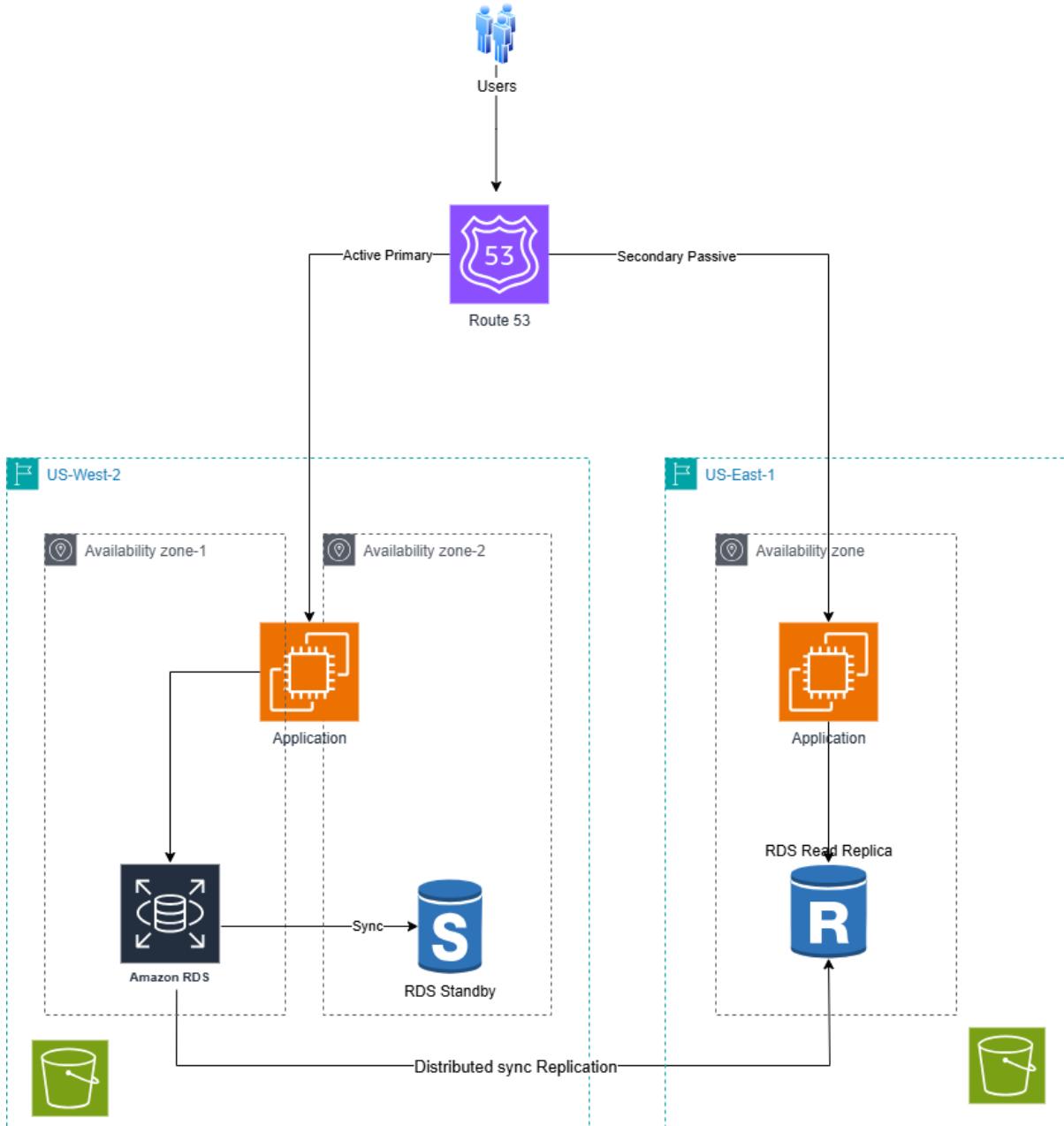
This PoC will serve as a foundation for a fully scalable and production-ready backup and disaster recovery solution.

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

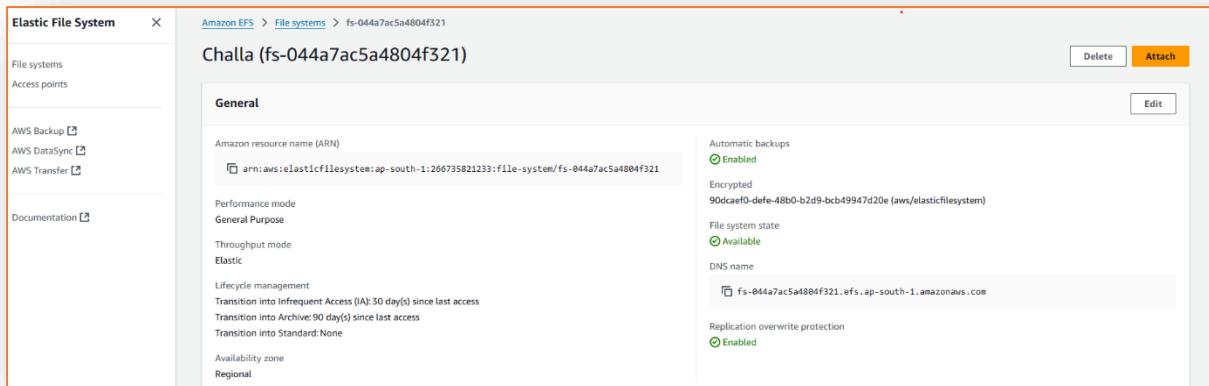
1. Go to the [AWS Management Console](#).
2. Search for EC2 in the search bar and click EC2.

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.

Instances (1/3) Info								
		Last updated less than a minute ago	Connect	Instance state ▾	Actions ▾	Launch instances	▼	
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
	Test	i-07c3f3b8bfd2f8012	Running View details Edit	t3.micro	3/3 checks passed	View alarms +	ap-south-1c	ec2-15-207-105-92.ap...

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

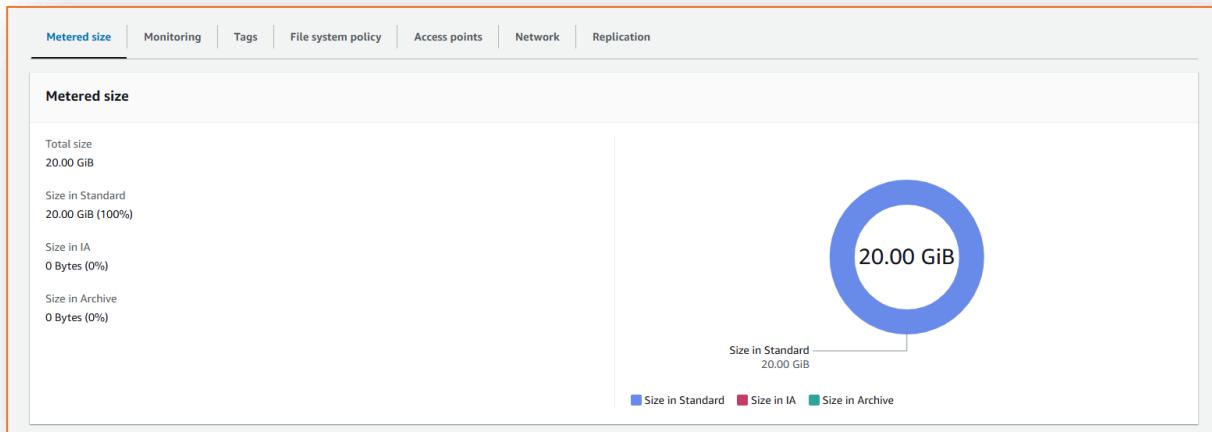
```
[root@ip-172-31-19-116 ~]# df -hT
Filesystem      Type      Size  Used Avail Use% Mounted on
devtmpfs        devtmpfs  455M    0   455M  0% /dev
tmpfs           tmpfs     464M    0   464M  0% /dev/shm
tmpfs           tmpfs     464M  452K  464M  1% /run
tmpfs           tmpfs     464M    0   464M  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.

The screenshot shows the 'Databases' section of the Amazon RDS console. On the left, there's a sidebar with links like 'Databases', 'Query Editor', 'Performance insights', 'Schemas', 'Exports in Amazon S3', 'Automated backups', 'Reserved instances', and 'Proxies'. The main area is titled 'Databases (1)' and contains a table with one row. The row details are: DB identifier: database-2, Status: Available, Role: Primary, Engine: MySQL Community Server, Region: ap-south-1b, and Size: db.t4g.micro. There are also columns for Recommendations, CPU usage (4.20%), and Current activity (1 Connections). At the top right, there are buttons for 'Group resources', 'Modify', 'Actions', 'Restore from S3', and 'Create database'.

With Multi-AZ Database with Read Replication server.

This screenshot shows the same 'Databases' section as the previous one, but it now displays two entries. The first entry is a Primary database named 'challa' with the same details as before: Status: Available, Role: Primary, Engine: MySQL Community Server, Region: us-east-1a, and Size: db.t4g.micro. The second entry is a Replica database with the same details: Status: Available, Role: Replica, Engine: MySQL Community Server, Region: us-east-1a, and Size: db.t4g.micro. The rest of the interface, including the sidebar and top navigation, remains identical to the first screenshot.

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.

The screenshot shows the AWS Backup console interface. The left sidebar navigation includes links for EC2, IAM, RDS, S3, Elastic Container Registry, Elastic Container Service, Elastic Beanstalk, VPC, and Route 53. The main content area is titled "AWS Backup > Backup plans > EC2".

Backup rules (1)
Backup rules specify the backup schedule, backup window, and lifecycle rules.

Name	Backup vault	Destination Backup vault
EC2	Default	Default

Resource assignments (1)
Resource assignments specify which resources will be backed up by this Backup plan.

Name	IAM role ARN	Creation time
ec2	arn:aws:iam::266735821233:role/service-role/AWSBackupDefaultServiceRole	February 25, 2025, 12:56:51 (UTC+05:30)

Backup jobs (2) Info
Records of your scheduled or on-demand backups in the last 30 days.

Backup job ID	Status	Resource name	Message category	Resource ID	Resource type	Creation time	Start by
1B042F7F-D64F-EA1C-2D0F-3BA82465493A	Completed	Test	Success	instance/i-07c5f3b8bfd2f8012	EC2	February 25, 2025, 14:00:00 (UTC+05:30)	February 25, 2025, 14:00:00 (UTC+05:30)
F65DFBF-E701-F210-96E9-28158A90EE1A	Completed	Test	Success	instance/i-07c5f3b8bfd2f8012	EC2	February 25, 2025, 13:00:00 (UTC+05:30)	February 25, 2025, 13:00:00 (UTC+05:30)

EFS Backups

Backup rules (1)

Name	Backup vault	Destination Backup vault
partha-EFS	Default	Default

Resource assignments (1)

Name	IAM role ARN	Creation time
efs	arn:aws:iam::266735821233:role/service-role/AWSBackupDefaultServiceRole	February 25, 2025, 12:55:33 (UTC+05:30)

Backup jobs (2) Info

Backup job ID	Status	Resource name	Message category	Resource ID	Resource type	Creation time	Start by
6CB55834-F489-FD38-7F5-E3185B717D2B	Completed	Challa	Success	file-system/fs-044a7ac5a4804f321	EFS	February 25, 2025, 14:00:00 (UTC+05:30)	February 25
7F04424A-556F-C467-0D80-40E772CEFB45	Completed	Challa	Success	file-system/fs-044a7ac5a4804f321	EFS	February 25, 2025, 13:00:00 (UTC+05:50)	February 25

RDS Backups.

Backup plan name: RDS

Version ID: YwE1ZDNlZjAtZmU0Y00N2NlLW13MzYtMmUwNjlyYz

Last modified: February 27, 2025, 12:27:21 (UTC+05:30)

Last runtime: February 27, 2025, 13:08:50 (UTC+05:30)

Backup rules (1)

Name	Backup vault	Destination Backup vault
RDS	Default	Default

Resource assignments (1)

Name	IAM role ARN	Creation time
RDS	arn:aws:iam::266735821233:role/service-role/AWSBackupDefaultServiceRole	February 27, 2025, 12:27:36 (UTC+05:30)

Backup jobs (1) Info

Backup job ID	Status	Resource name	Message category	Resource ID	Resource type	Creation time	Start by
CF2DCAFE-06D7-9450-73AA-4883E5CA23D6	Completed	database-2	Success	database-2	RDS	February 27, 2025, 13:05:00 (UTC+05:30)	February 27, 2025, 14:05:00 (UTC+05:30)

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.

Amazon Machine Images (AMIs) (2) Info						
Owned by me		Find AMI by attribute or tag			Actions	
<input type="checkbox"/>	Name	<input type="checkbox"/>	AMI name	<input type="checkbox"/>	AMI ID	<input type="checkbox"/>
<input type="checkbox"/>	partha-1	<input type="checkbox"/>	AwsBackup_i-0a7e7b967c2ec1f...	<input type="checkbox"/>	ami-031fdebd93510faf2	<input checked="" type="checkbox"/> 266735821233/AwsBackup_i-0a7e7b967c2ec1f67_C718C232-5...

Ec2 instance restore the within region.

The screenshot shows the AWS EC2 Instances page. There are three instances listed:

- Test**: Instance ID i-07c1f5fb0fc7f9012, Running, t1.micro, 3/3 checks passed, ap-south-1, Public IPv4 DNS ec2-15-203-105-92.ap...
- restore backup**: Instance ID i-0453bef60aaa00bdb, Running, t1.micro, 3/3 checks passed, ap-south-1, Public IPv4 DNS ec2-65-2-116-95.ap...
- rsht**: Instance ID i-006e078e42bd9cf, Stopped, t2.micro, View alarms +, ap-south-1, Public IPv4 DNS -

The 'restore backup' instance is selected, and its details are shown in the bottom panel:

Details		Status and alarms	Monitoring	Security	Networking	Storage	Tags
i-0453bef60aaa00bdb (restore backup)							
Instance summary							
Instance ID	i-0453bef60aaa00bdb	Public IPv4 address	65.2.116.95 open address	Private IPv4 address	172.31.26.119	Public IPv6 DNS	wc2-65-2-116-95.ap-south-1.compute.amazonaws.com open address
IPv6 address	-	Instance state	Running	Private IP DNS name (IPv6 only)	(ip-172-31-26-119.ap-south-1.compute.internal)		
Hostname type	IP name: ip-172-31-26-119.ap-south-1.compute.internal						

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.

The screenshot shows the AWS RDS Databases page. There are two databases listed:

DB identifier	Status	Role	Engine	Region ...	Size	Recommendations	CPU	Current activity	Ma
database-2	Available	Primary	MySQL Co...	ap-south-1b	db.t4g.micro	1	3.55%	1 Connections	no
test	Available	Instance	MySQL Co...	ap-south-1a	db.t4g.micro	1	3.13%	0 Connections	no

AWS EC2 instance with RDS connection

The screenshot shows the AWS CloudWatch Metrics console with a single entry in the 'Connected compute resources' list:

Resource identifier	Resource type	Availability Zone	VPC security group	Compute resource security group
i-0ee9d53fc8f071270	EC2 instance	ap-south-1c	rds-ec2-1	ec2-rds-1

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

```
[root@ip-10-10-2-107 ~]# mysql --version
mysql Ver 15.1 Distrib 5.7.6-MariaDB, for Linux (x86_64) using readline 5.1
[root@ip-10-10-2-107 ~]# mysql -h "mydb.cluster-xys123.ap-south-1.rds.amazonaws.com" -u admin -p
Enter password:
mysqld: unknown option '--port=33060'
Unknown MySQL server host "mydb.cluster-xys123.ap-south-1.rds.amazonaws.com" (2)
[root@ip-10-10-2-107 ~]# mysql -h test.c3gy1s0by1is.ap-south-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 15
Server version: 8.0.40 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> CREATE DATABASE mydatabase;
Query OK, 1 row affected (0.01 sec)

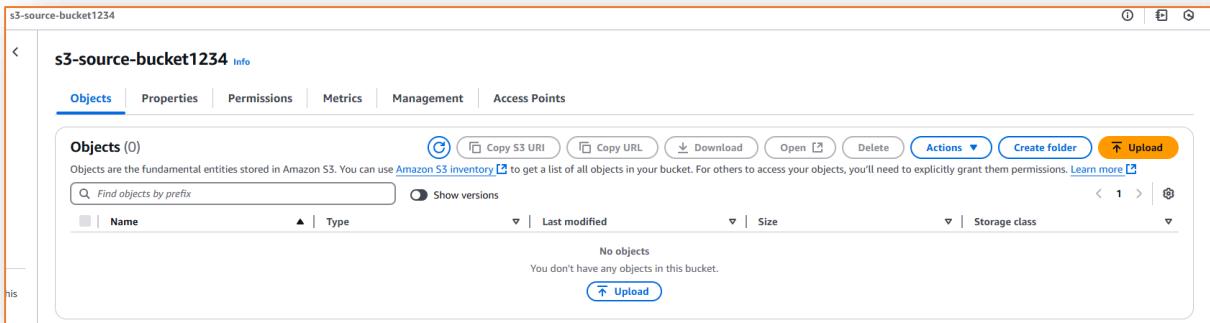
MySQL [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
6 rows in set (0.01 sec)

MySQL [(none)]>
```

S3 Cross-Region Replication (CRR) for DR

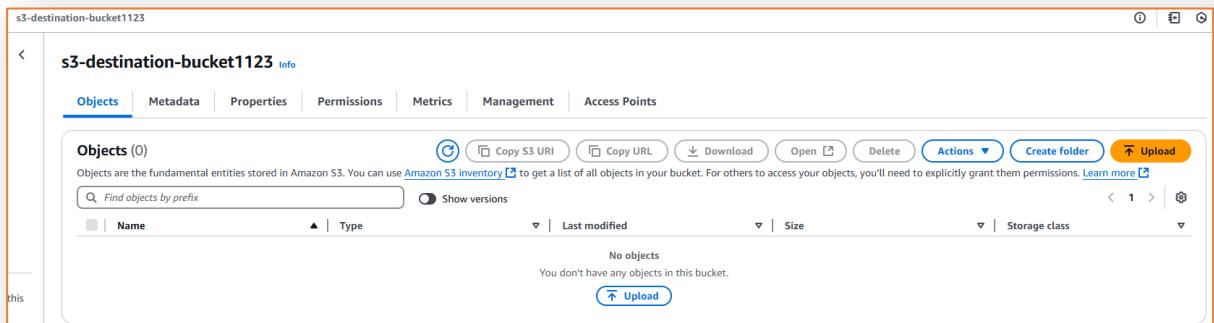
Step 1: Create Source and Destination Buckets

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

Roles (105) Info		
An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.		
<input type="text"/> s3 X 1 match		
Role name	Trusted entities	Last activity
S3bucket	AWS Service: s3	Yesterday

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.

Replication rules (1) View details Edit rule Delete Actions ▾ Create replication rule							
Use replication rules to define options you want Amazon S3 to apply during replication such as server-side encryption, replica ownership, transitioning replicas to another storage class, and more. Learn more							
Replication rule name	Status	Destination bucket	Destination Region	Priority	Scope	Storage class	Replica ownership
pardhu	Enabled	s3://s3-destination-bucket1123	US East (N. Virginia) us-east-1	0	Entire bucket	Same as source	Same as source

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

This source bucket

Amazon S3 > Buckets > s3-source-bucket1234								
s3-source-bucket1234 Info								
Objects (1) 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								
Name	Type	Last modified	Size	Storage class				
MobaTerm_Portable_v22.3/	Folder	-	-	-	-	-	-	-

This is destination bucket.

The screenshot shows the AWS S3 console with the path 'Amazon S3 > Buckets > s3-destination-bucket1123'. The 'Objects' tab is selected. There is one object listed: 'MobaXterm_Portable_v22.3/' which is a folder. Other tabs include 'Metadata', 'Properties', 'Permissions', 'Metrics', 'Management', and 'Access Points'. Action buttons at the top include 'Copy S3 URI', 'Copy URL', 'Download', 'Open', 'Delete', 'Actions', 'Create folder', and 'Upload'. A search bar and filter options for 'Name', 'Type', 'Last modified', 'Size', and 'Storage class' are available.

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

The screenshot shows the AWS S3 console with the path 'Amazon S3 > Buckets > s3-source-bucket1234'. The 'Objects' tab is selected and shows six objects: 'Deploying an EC2 Auto scaling-partha.docx', 'Deploying an EC2 Auto scaling-partha.doc', 'MobaXterm_Portable_v22.3/', 'Shop-Verse-master.zip', and 'Shop-Verse-master.zip'. The 'Properties' tab is also visible. Action buttons like Copy S3 URI, Copy URL, Download, Open, Delete, Actions, Create folder, and Upload are present. A search bar and filter options for Name, Type, Version ID, Last modified, Size, and Storage class are included.

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.

The screenshot shows the AWS Management Console interface for the EC2 service. The left sidebar has sections for Instances, IAM Store, Security Groups, and VPCs. The main area is titled 'Instances (1/2)' and shows a table with two rows. The first row, 'Primary', is selected and highlighted in blue. The second row is 'ip-10-10-2-107.ap-south-1.c...' with instance ID i-097eaac685bd4fccf. The columns include Name, Instance ID, Instance state, Instance type, and Status check. A 'Launch instances' button is visible at the top right of the table area.

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

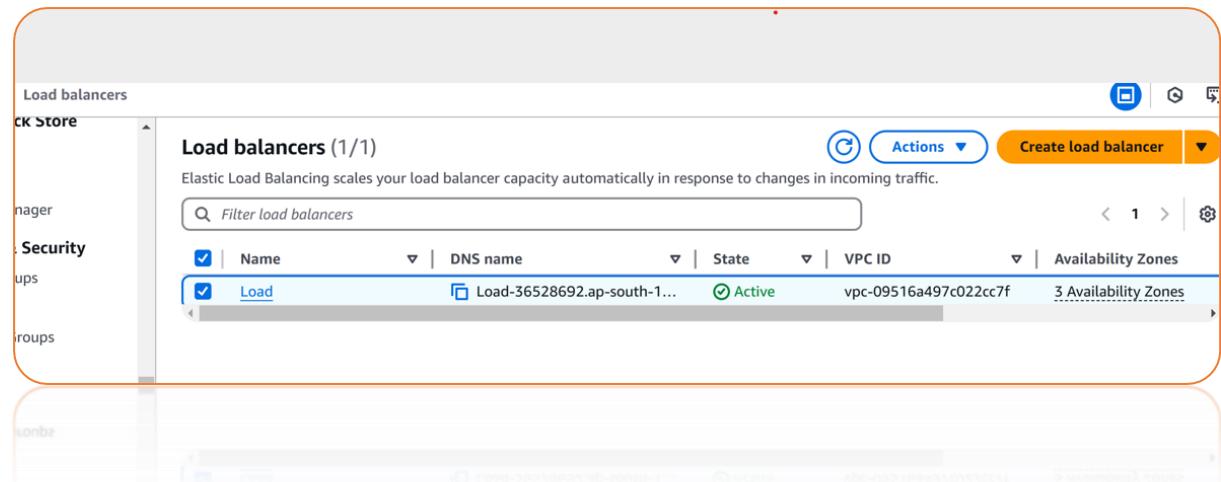
The screenshot shows the AWS Management Console interface for the EC2 service. The left sidebar has sections for Instances, IAM Store, Security Groups, and VPCs. The main area is titled 'Instances (1/1)' and shows a table with one row. The row is 'Se-1' with instance ID i-08fcabf568c396558. The columns include Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. A 'View alarms +' button is visible next to the status check column. The 'Se-1' instance is selected and highlighted in blue.

Set Up Elastic Load Balancers (ALB)

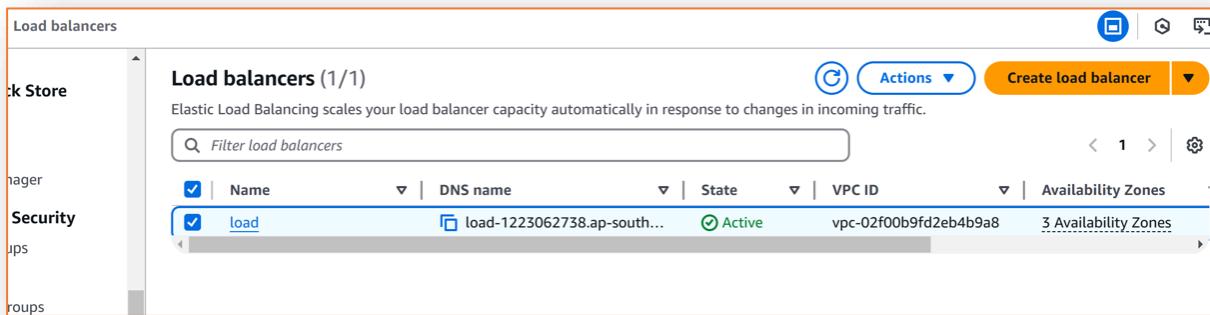
Step 1: Create Load Balancers in Mumbai and Singapore Regions

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.

The screenshot shows the AWS Lambda console with the sidebar navigation bar visible. Under the 'Target groups' section, there is one target group named 'target'. The table displays the following details:

Name	ARN	Port	Protocol	Target type
target	arn:aws:elasticloadbalancing:ap-south-1:.../target	80	HTTP	Instance

Target group Singapore region.

The screenshot shows the AWS Lambda console with the sidebar navigation bar visible. Under the 'Target groups' section, there is one target group named 'target'. The table displays the following details:

Name	ARN	Port	Protocol	Target type
target	arn:aws:elasticloadbalancing:ap-southeast-1:.../target	80	HTTP	Instance

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

The screenshot shows the AWS Lambda console with the sidebar navigation bar visible. Under the 'Health checks' section, there are two health checks listed:

ID	Name	Details	Status	Alarm	State
808e4856-38...	Primary	http://load-36528692.ap-s...	Healthy	1 of 1 in OK	Enabled
f936efc5-a00f...	Se-1	http://load-1223062738.ap...	Healthy	1 of 1 in OK	Enabled

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**.

The screenshot shows the AWS Route 53 Hosted Zone Records page. At the top, there are tabs for 'Records (4)', 'DNSSEC signing', and 'Hosted zone tags (0)'. Below the tabs, there's a search bar labeled 'Filter records by property or value' and several dropdown filters for 'Type', 'Routing p...', and 'Alias'. A large orange button at the top right says 'Create record'. The main table lists four records:

Record name	Type	Alias	Value/Route traffic to
challa.today	NS	Simple	ns-1206.awsdns-22.org. ns-937.awsdns-53.net. ns-209.awsdns-26.com. ns-1799.awsdns-32.co.uk
challa.today	SOA	Simple	ns-1206.awsdns-22.org. a

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.

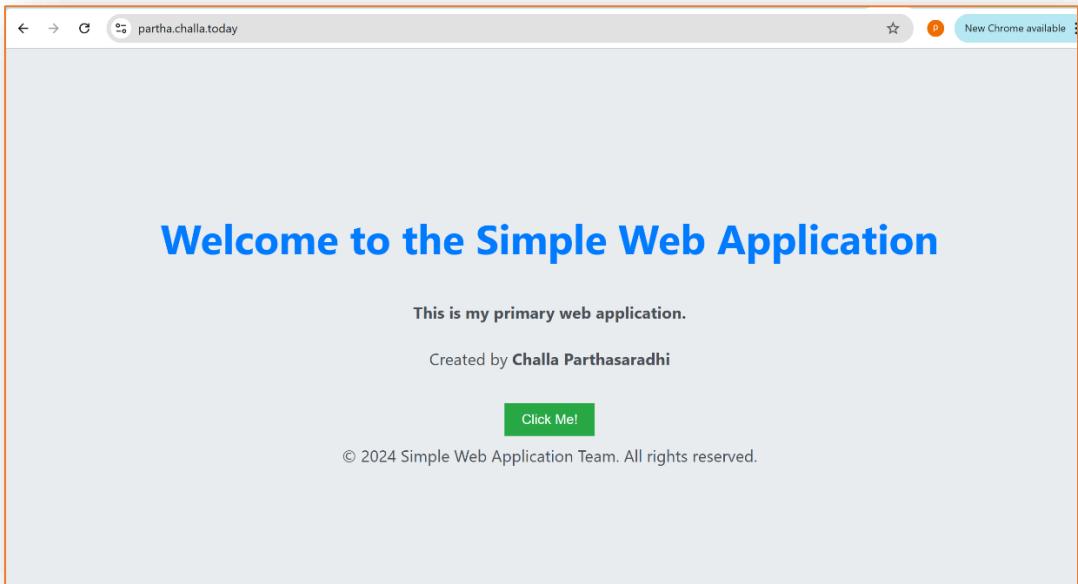
Records (4) DNSSEC signing Hosted zone tags (0)

Records (4) [Info](#) [Delete record](#) [Import zone file](#) [Create record](#)

Automatic mode is the current search behavior optimized for best filter results. [To change modes go to settings.](#)

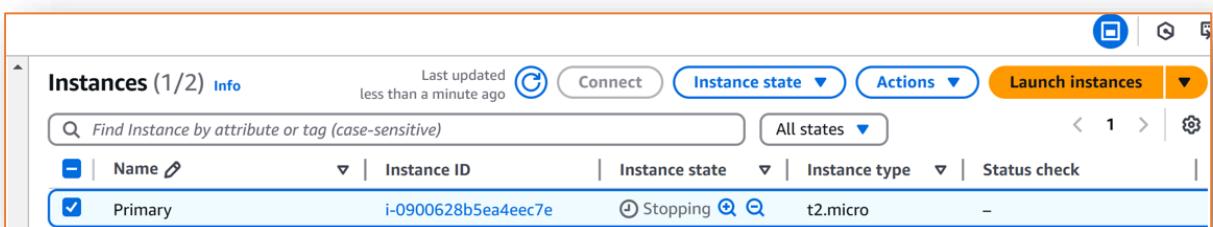
<input type="checkbox"/>	Record name	Type	Routin...	Differ...	Alias	Value/Route traffic to
<input type="checkbox"/>	challa.today	NS	Simple	-	No	ns-1206.awsdns-22.org. ns-937.awsdns-53.net. ns-209.awsdns-26.com. ns-1799.awsdns-32.co.uk.
<input type="checkbox"/>	challa.today	SOA	Simple	-	No	ns-1206.awsdns-22.org. aw
<input type="checkbox"/>	partha.challa.today	A	Failover	Primary	Yes	dualstack.load-36528692.a
<input type="checkbox"/>	partha.challa.today	A	Failover	Secondary	Yes	dualstack.load-1223062738

- 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.

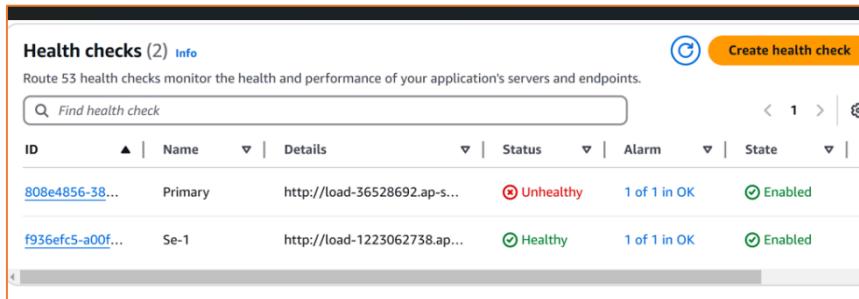


- You can test the traffic redirect or not you can check.
- First You need to be ec2 instance stop the server then you can open my secondary web application

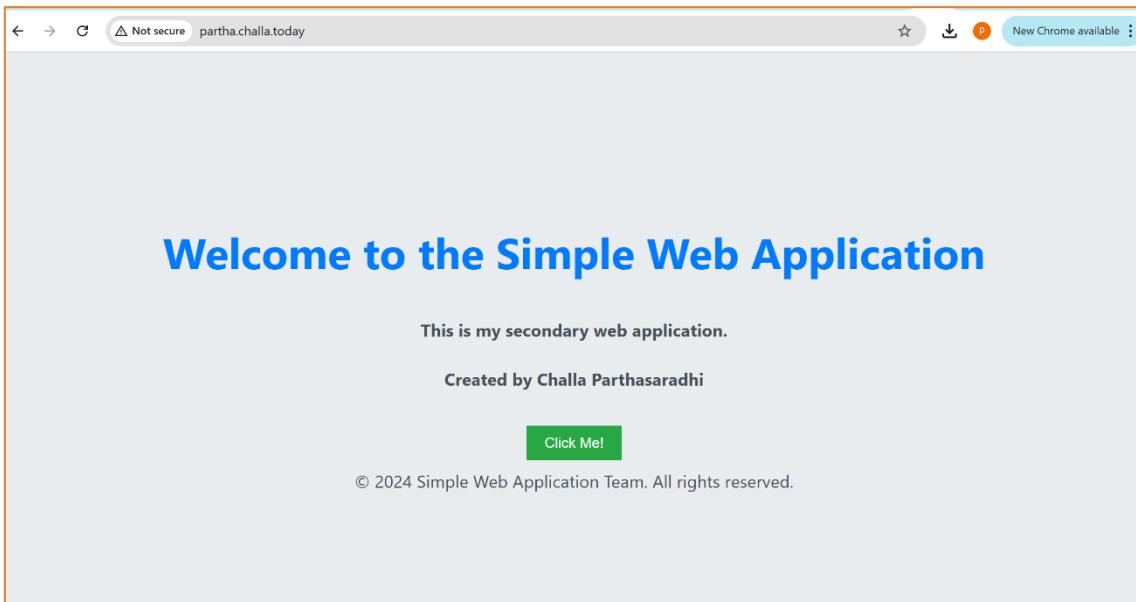
Ec2 instance stop the server.



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

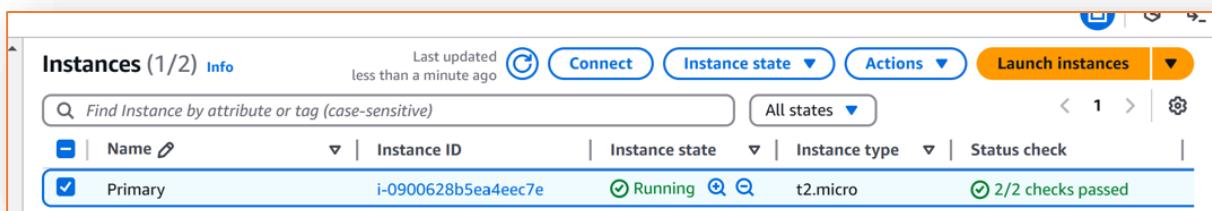


- You can check web site is open secondary web application because of route traffic redirect.

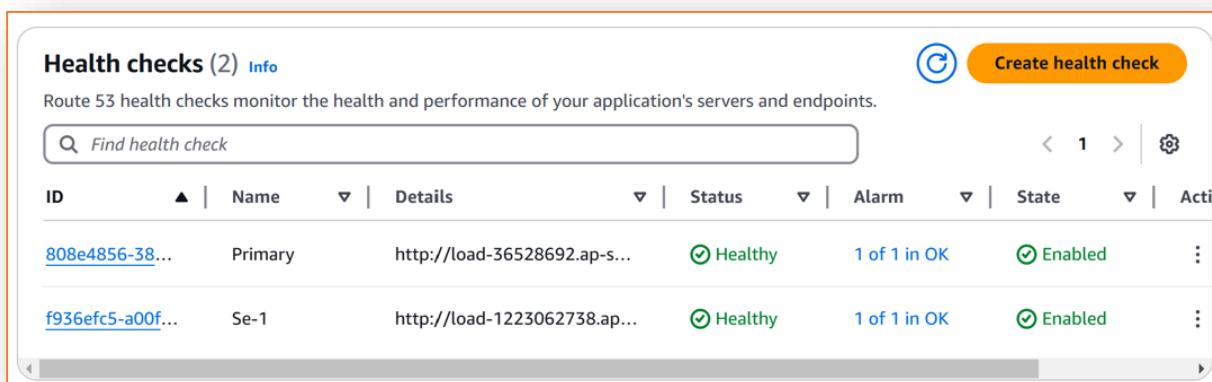


- Once you tested that all then you can up the server then you can access the primary web application.

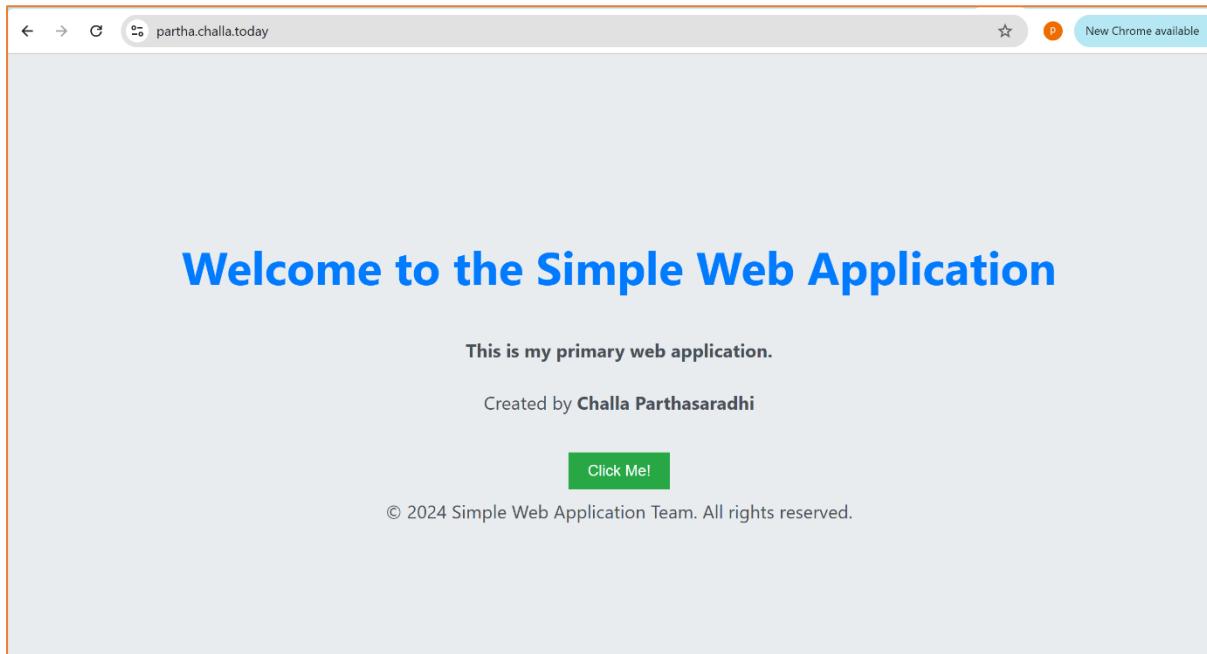
EC2 instance.



- 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

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

2 Create a New IAM Role

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

3 Attach Required Policies

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

- AWSElasticDisasterRecoveryConversionServerPolicy
- AWSElasticDisasterRecoveryEc2InstancePolicy
- AmazonSSMManagedInstanceCore
- [AmazonS3FullAccess](#)
- [AmazonSSMFullAccess](#)
- [AWSElasticDisasterRecoveryAgentPolicy](#)
- [AWSElasticDisasterRecoveryFallbackPolicy](#)
- [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**.

AWS Elastic Disaster Recovery > Settings: Default replication > Edit default replication settings

Edit default replication settings Info

Replication server configuration Info
Replication servers are light weight EC2 instances launched by AWS Elastic Disaster Recovery to facilitate the transfer of blocks of data from your source servers to AWS.

Staging area subnet Info
The staging area subnet is the subnet within which replication servers and conversion servers are launched. By default, AWS Elastic Disaster Recovery will use the default subnet on your AWS Account.

subnet-0362108727e6822b0 vpc-076da4d57335454c3	▼
---	---

Replication server instance type
The replication server instance type is the default EC2 instance type to use for replication servers. The recommended best practice is to not change the replication server instance type unless there is a specific reason for doing so. This feature is not supported on Outposts.

t3.small	▼
----------	---

Volumes

This is my AWS Disaster recovery Source server

AWS Elastic Disaster Recovery > Source servers

Source servers (2) Info

Actions	Replication	Initiate recovery job					
< 1 > ⚙							
Hostname	Ready for recovery	Data replication status	Last recovery result	Pending actions	Replicating from	Replicating to	Failing back
ip-10-10-2-107.ap-south-1.compute.internal (i-097eaac685bd4fcff)	Ready	Healthy	-	-	ap-south-1a	ap-south-1a	-
ip-10-10-10-170.ap-south-1.compute.internal (i-01e97e67d02364c95)	Ready	Healthy	-	-	ap-south-1a	ap-south-1a	-

This is recovery instance.

AWS Elastic Disaster Recovery > Recovery instances

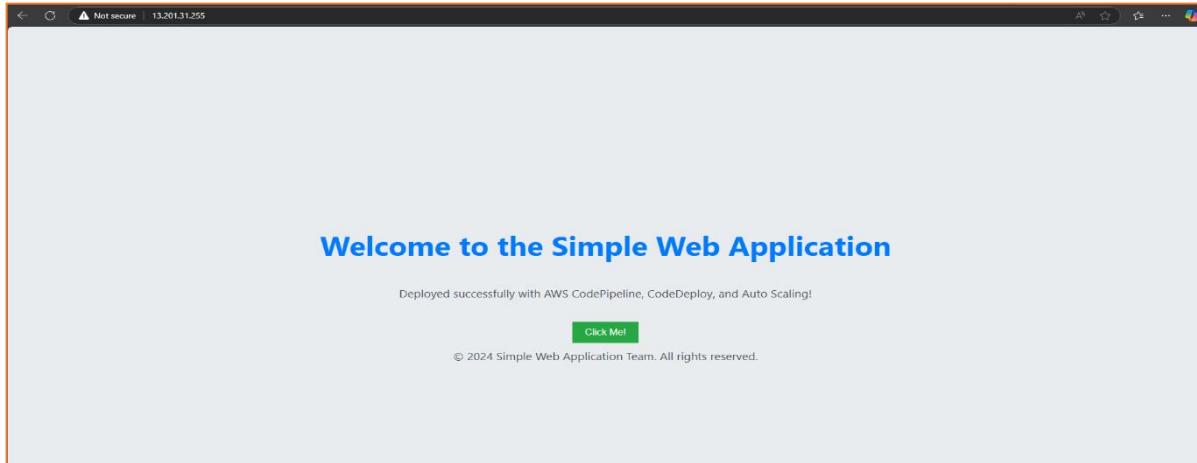
Recovery instances (1) Info

Actions	Start reversed replication	Complete failback						
< 1 > ⚙								
Instance ID	Reversed direction launch state	Data replication status	Pending actions	Replicating to source server	Replicating from	Replicating to	Last launch result	Launched from source server
i-097eaac685bd4fcff	Ready	Healthy	Initiate recovery or drill on source server	ap-south-1: ip-10-10-2-107.ap-south-1.compute.internal (s-e11abb88d388e378f)	ap-south-1a	ap-south-1a	-	ip-10-10-2-107.ap-south-1.compute.internal (s-e11abb88d388e378f)

This is my recovery Ec2 instance for testing.

<input checked="" type="checkbox"/> ip-10-10-2-107.ap-south-1.compute.internal	i-097eaac685bd4fcff	Running	Q Q	c5.large	3/3 checks passed	View alarms +	ap-south-1a	-
<input type="checkbox"/> ip-10-10-10-170.ap-south-1.compute.internal	i-01e97e67d02364c95	Running	Q Q	c5.large	0 Initializing	View alarms +	ap-south-1a	-

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.

The screenshot shows the AWS Elastic Disaster Recovery console. On the left, there's a navigation sidebar with options like "Source servers", "Recovery instances", "Source networks", "Recovery job history" (which is selected), "Settings", and "Documentation". The main area displays a table titled "Job: drsjob-e72894330899f90bc". The table has three columns: "Details", "Status", and "Initiated by". Under "Details", it shows "Type: Recovery", "Start time: 28/2/2025, 3:01:03 pm", and "Completed time: 28/2/2025, 3:13:12 pm". Under "Status", it shows "Status: Completed". Under "Initiated by", it shows "Successfully launched 1/1". Below this, there's a section titled "Job log" with a table of log entries. The log table has columns for "Time", "Event", and "Additional data". The log entries are:

Time	Event	Additional data
28/2/2025, 3:13:12 pm	Job ended	
28/2/2025, 3:13:10 pm	Successfully launched recovery instance	Source server: ip-10-10-2-107.ap-south-1.compute.internal (s-e11abb88d388e378f) Recovery instance ID: i-0360a9e149ce8e057
28/2/2025, 3:08:04 pm	Started launching recovery instance	Source server: ip-10-10-2-107.ap-south-1.compute.internal (s-e11abb88d388e378f)
28/2/2025, 3:08:02 pm	Conversion succeeded	Source server: ip-10-10-2-107.ap-south-1.compute.internal (s-e11abb88d388e378f) Conversion server instance ID: i-0d58eaebe1725c4
28/2/2025, 3:03:11 pm	Conversion started	Source server: ip-10-10-2-107.ap-south-1.compute.internal (s-e11abb88d388e378f)
28/2/2025, 3:03:10 pm	Finished taking snapshot	Source server: ip-10-10-2-107.ap-south-1.compute.internal (s-e11abb88d388e378f)
28/2/2025, 3:03:10 pm	Started taking snapshot	Source server: ip-10-10-2-107.ap-south-1.compute.internal (s-e11abb88d388e378f)

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.

AWS Elastic Disaster Recovery > Settings: Default replication > Edit default replication settings

Edit default replication settings Info

Replication server configuration Info
Replication servers are light weight EC2 instances launched by AWS Elastic Disaster Recovery to facilitate the transfer of blocks of data from the disks on your source servers to AWS.

Staging area subnet Info
The staging area subnet is the subnet within which replication servers and conversion servers are launched. By default, AWS Elastic Disaster Recovery will use the default subnet on your AWS Account.

subnet-0362108727e6822b0 vpc-076da4d57335454c3

Replication server instance type
The replication server instance type is the default EC2 instance type to use for replication servers. The recommended best practice is to not change the replication server instance type unless there is a specific reason for doing so. This feature is not supported on Outposts.

t3.small

Volumes

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

AWS Elastic Disaster Recovery > Source servers

Source servers (1) Info

Filter source servers by property or value

Hostname	Ready for recovery	Data replication status	Last recovery result	Pending actions	Replicating from	Replicating to
ip-172-31-16-198.ap-south-1.compute.internal (i-063dea1c5f583b3ab)	Ready	Healthy	-	Initiate drill	ap-south-1c	ap-southeast-1b

This is recovery instance.

AWS Elastic Disaster Recovery > Recovery instances

Recovery instances (1) Info

Filter recovery instances by property or value

Instance ID	Reversed direction launch state	Data replication status	Pending actions	Replicating to source server	Replicating from	Replicating to
i-006b93c470a5e31f7	Not started	-	Start reversed replication to ap-south-1	-	ap-southeast-1b	-

This is my recovery Ec2 instance for testing.

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table of instances:

Name	Instance ID	Instance state	Instance type	Status
ip-172-31-16-198.ap-south-1.compute.internal	i-0bc79d30a981918eb	Running	c5.large	Init
AWS Elastic Disaster Recovery Replication Server	i-01b975ba690f83fec	Running	t3.small	3/3
AWS Elastic Disaster Recovery Conversion Server	i-0023d835adb1c5d56	Terminated	m5.large	-

Below the table, the instance details for **i-0bc79d30a981918eb (ip-172-31-16-198.ap-south-1.compute.internal)** are shown. The Public IPv4 address is highlighted with a green box and the text "Public IPv4 address copied". The address is 54.255.236.130. A tooltip indicates it can be opened in a browser.

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

The screenshot shows a web browser window displaying a simple web application. The URL bar shows the IP address 54.255.236.130. The page content is as follows:

Welcome to the Simple Web Application

Deployed successfully with AWS CodePipeline, CodeDeploy, and Auto Scaling!

[Click Me!](#)

© 2024 Simple Web Application Team. All rights reserved.

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

Time	Event	Additional data
3/9/2025, 1:44:10 PM	Job ended	
3/9/2025, 1:44:08 PM	Successfully launched recovery instance	Source server: ip-172-31-16-198.ap-south-1.compute.internal (s- Recovery instance ID: i-006b93c470a5e31f7
3/9/2025, 1:37:57 PM	Started launching recovery instance	Source server: ip-172-31-16-198.ap-south-1.compute.internal (s- Conversion server instance ID: i-00ce6859b95fbda4f
3/9/2025, 1:37:54 PM	Conversion succeeded	Source server: ip-172-31-16-198.ap-south-1.compute.internal (s- Conversion server instance ID: i-00ce6859b95fbda4f
3/9/2025, 1:33:24 PM	Conversion started	Source server: ip-172-31-16-198.ap-south-1.compute.internal (s- Conversion server instance ID: i-00ce6859b95fbda4f
3/9/2025, 1:33:23 PM	Finished taking snapshot	Source server: ip-172-31-16-198.ap-south-1.compute.internal (s- Conversion server instance ID: i-00ce6859b95fbda4f
3/9/2025, 1:33:23 PM	Started taking snapshot	Source server: ip-172-31-16-198.ap-south-1.compute.internal (s- Conversion server instance ID: i-00ce6859b95fbda4f

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.

Name	Status	Compliance status	Drift status	App version	Invoker	Start time
Assessment-report-m5uv25dq01d	Success	Policy met	Not drifted	Version_2	User	February 27, 2025 at 18:01 (UTC+5:30)
Assessment-report-vv4wuptOysl	Success	Policy met	Not drifted	Version_1	User	February 27, 2025 at 17:57 (UTC+5:30)

Application resiliency score.

Ec2 instance with autoscaling

Without autoscaling AWS resilience hub is given to implement autoscaling.

The screenshot shows the AWS Resilience Hub interface with a navigation bar at the top: AWS Resilience Hub > Applications > web-application > Assessment reports > Assessment-report-bn1iywvu8qp. Below the navigation is a table comparing two options:

Why you should choose this option:	• Optimize for cost	• Optimize for Availability Zone (AZ) RTO/RPO.
Description:	Amazon EC2 Auto Scaling group (ASG) with a single Amazon EC2 instance.	Amazon EC2 Auto Scaling group (ASG) with multiple active Amazon EC2 instances that are deployed across multiple Availability Zones.
Changes:	3 Changes <ul style="list-style-type: none">Deploy your Amazon EC2 instances to be managed by an Amazon EC2 Auto Scaling group (ASG) to enable recovery in the event of an Infrastructure or Availability Zone disruptions.Configure your Amazon EC2 Auto Scaling group (ASG) to utilize several Availability Zones to enable recovery in the event of Availability Zone disruption.Modify the configuration of your Amazon EC2 Auto Scaling group (ASG) by changing the minSize parameter to 1 value to enable Amazon EC2 instances recovery in the event of Infrastructure disruption.	3 Changes <ul style="list-style-type: none">Deploy your Amazon EC2 instances to be managed by an Amazon EC2 Auto Scaling group (ASG) to enable recovery in the event of an Infrastructure or Availability Zone disruptions.Configure your Amazon EC2 Auto Scaling group (ASG) to utilize several Availability Zones to enable recovery in the event of Availability Zone disruption.Modify the configuration of your Amazon EC2 Auto Scaling group (ASG) by changing the minSize parameter to 2 value to enable Amazon EC2 instances recovery in the event of Infrastructure disruption.

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.

The screenshot shows the 'Objects' tab of the 'my-cost-optimized-bucket' page. There are two objects listed:

- challa.pem**: Type pem, Last modified March 2, 2025, 06:36:23 (UTC+05:30), Size 1.6 KB, Storage class Standard.
- Final Document 1.docx**: Type docx, Last modified March 2, 2025, 06:36:22 (UTC+05:30), Size 5.7 MB, Storage class Standard.

To create Lifecycle Rules.

The screenshot shows the 'Management' tab of the 'my-cost-optimized-bucket' page. Under the 'Lifecycle configuration' section, there is one rule defined:

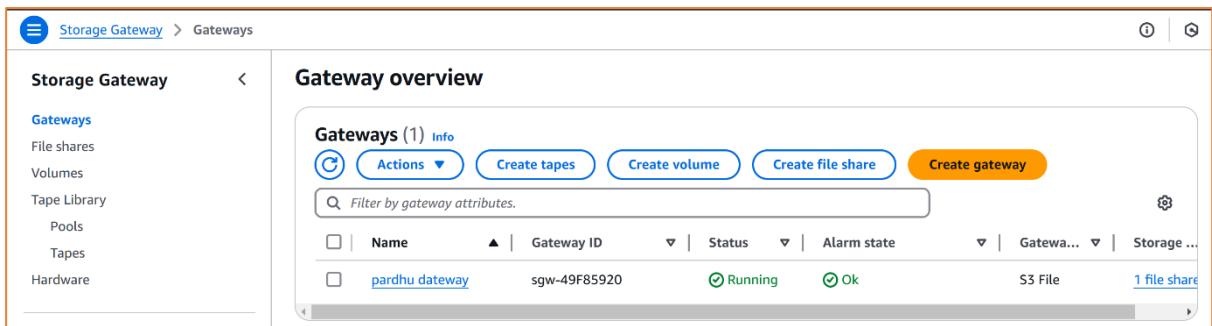
Lifecycle rule...	Status	Scope	Current version actions
CostOptimizationRule	Enabled	Entire bucket	Transition to Standard-IA, then Glacier Deep Archive, then expires

Configure AWS Storage Gateway

reducing expensive local storage costs.

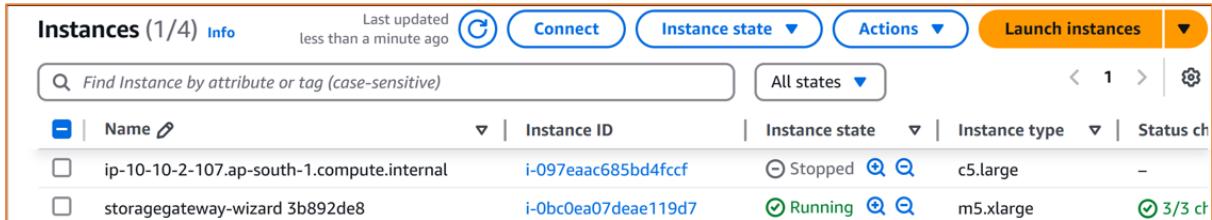
1. Set Up AWS Storage Gateway

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.



The screenshot shows the 'Gateway overview' section of the AWS Storage Gateway console. On the left, there's a sidebar with options like 'File shares', 'Volumes', 'Tape Library', 'Pools', 'Tapes', and 'Hardware'. The main area has a title 'Gateway overview' and a sub-section 'Gateways (1) Info'. It features several buttons: 'Actions', 'Create tapes', 'Create volume', 'Create file share', and 'Create gateway' (which is highlighted). Below these buttons is a search bar with the placeholder 'Filter by gateway attributes.' A table follows, showing the single gateway entry with columns for Name, Gateway ID, Status, Alarm state, and more. The gateway listed is 'pardhu dateway' with ID 'sgw-49F85920', status 'Running', alarm state 'Ok', and associated with an 'S3 File' share.

To lunch ec2 instance storage gateway



The screenshot shows the 'Instances (1/4)' section of the AWS EC2 console. It lists two instances: 'ip-10-10-2-107.ap-south-1.compute.internal' (instance ID i-097eaac685bd4fccf) and 'storagegateway-wizard 3b892de8' (instance ID i-0bc0ea07deae119d7). The first instance is stopped, while the second is running. The table includes columns for Name, Instance ID, Instance state, Instance type, and Status. There are also buttons for 'Connect', 'Actions', and 'Launch instances'.

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.

The screenshot shows the AWS Storage Gateway interface. On the left, there's a sidebar with 'Storage Gateway' selected under 'File shares'. The main area displays 'All NFS clients' settings, including 'Root squash (default)', 'Requester pays', 'Directory permissions', and 'File permissions'. Below this is a section titled 'Example Commands' with three examples:

- On Linux:** sudo mount -t nfs -o noblock,hard 172.31.37.19:/my-cost-optimized-bucket [MountPath] [Copy](#)
- On Microsoft Windows:** mount -o noblock -o mtype=hard 172.31.37.19:/my-cost-optimized-bucket [WindowsDriveLetter]: [Copy](#)
- On macOS:** sudo mount -t nfs -o vers=3,rsize=1048576,wsize=1048576,hard, noblock -v 172.31.37.19:/my-cost-optimized-bucket [MountPath] [Copy](#)

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

Session ID: Pardha_user-
z7lyxeaoorbtj40shdl4n7s884 Instance ID: i-0e1f660cbf7ab60b1

```
sh-4.2$ sudo -i
[root@ip-172-31-34-223 ~]# sudo mkdir -p /mnt/my-bucket
[root@ip-172-31-34-223 ~]# ls
[root@ip-172-31-34-223 ~]# sudo mount -t nfs -o noblock,hard 172.31.37.19:/my-cost-optimized-bucket /mnt/my-bucket
[root@ip-172-31-34-223 ~]# ls
[root@ip-172-31-34-223 ~]# cd /mnt/my-bucket
[root@ip-172-31-34-223 my-bucket]# ls
Final Document POC Completed 1.docx challa.pem
[root@ip-172-31-34-223 my-bucket]# vi text.html
[root@ip-172-31-34-223 my-bucket]# ls
Final Document POC Completed 1.docx challa.pem  text.html
[root@ip-172-31-34-223 my-bucket]#
```

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

The screenshot shows the AWS S3 Buckets page for 'my-cost-optimized-bucket'. The left sidebar has 'General purpose buckets' selected. The main area shows the 'Objects' tab with three items listed:

Name	Type	Last modified	Size	Storage class
challa.pem	pem	March 2, 2025, 06:36:23 (UTC+05:30)	1.6 KB	Standard
Final Document POC Completed 1.docx	docx	March 2, 2025, 06:36:22 (UTC+05:30)	5.7 MB	Standard
text.html	html	March 2, 2025, 09:54:42 (UTC+05:30)	27.0 B	Standard