

NAME: PRANALI SACHIN KAPARE

BATCH: MCA 10 JUNE

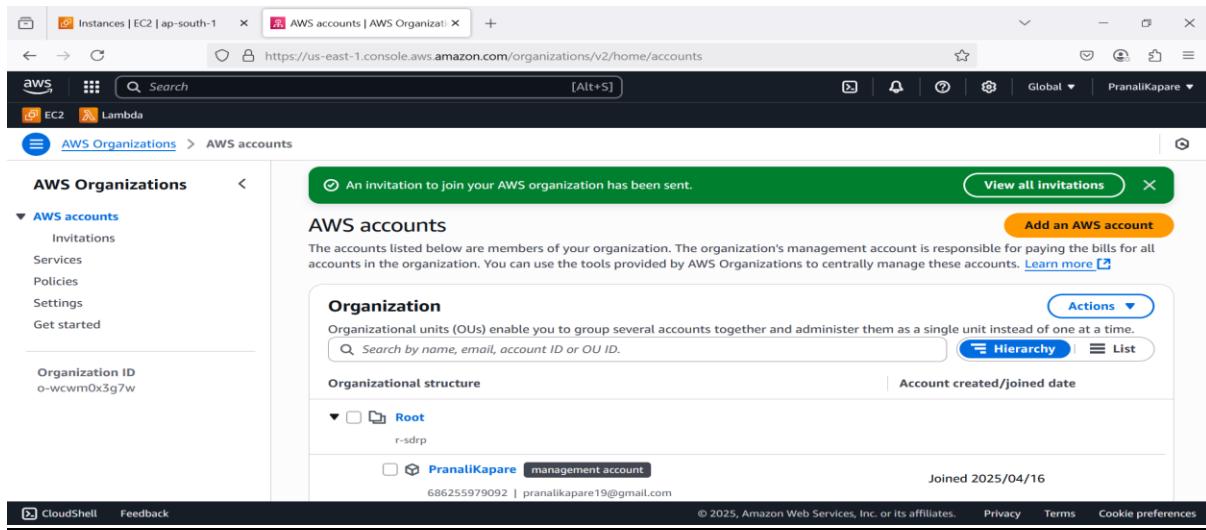
SET B1

Task 1: Set up AWS Organizations for managing multiple AWS accounts.

ANS:

The screenshot shows the AWS Organizations console with the URL <https://us-east-1.console.aws.amazon.com/organizations/v2/home/accounts>. The left sidebar shows the AWS Organizations navigation pane with 'AWS accounts' selected. The main content area is titled 'AWS accounts' and displays a list of organization members. It includes a search bar, a 'Hierarchy' button, and a 'List' button. A single account, 'PranaliKapare', is listed as a 'management account' with the ID '686255979092' and email 'pranalikapare19@gmail.com'. The account was joined on '2025/04/16'. The bottom of the page includes standard AWS footer links like CloudShell, Feedback, Privacy, Terms, and Cookie preferences.

The screenshot shows the 'Add an AWS account' page within the AWS Organizations console. The URL is <https://us-east-1.console.aws.amazon.com/organizations/v2/home/accounts/add/invite>. The left sidebar shows the AWS Organizations navigation pane with 'AWS accounts' selected. The main content area has two options: 'Create an AWS account' (unselected) and 'Invite an existing AWS account' (selected). Below this, there is a section titled 'Invite one or more existing AWS accounts to join your organization' with a text input field containing 'komalrode267@gmail.com' and a 'Add another account' button. At the bottom, there is a message input field with the placeholder 'This is only for Practice....Thank You!!!'. The bottom of the page includes standard AWS footer links like CloudShell, Feedback, Privacy, Terms, and Cookie preferences.



Task 2: What is the purpose of use of AWS-CLI? What are the key requirements to access AWS account using AWS-CLI.

Create a working custom VPC. Launch one server in each subnet & connect private server with bastion host using CLI.

ANS:

Purpose of AWS CLI

The AWS CLI (Command Line Interface) allows you to interact with AWS services from your terminal or shell using simple commands.

Key Requirements to Access AWS Account Using AWS CLI

1. Install AWS CLI

- Download and install the AWS Command Line Interface tool on your system.

2. IAM User with Programmatic Access

- You need an IAM user in your AWS account.
- When creating the user, enable programmatic access (this gives you the Access Key and Secret Key).
- The user should have the appropriate permissions or policies (like AmazonEC2FullAccess, AdministratorAccess, etc.).

3. Access Key and Secret Access Key

- After creating the IAM user, you'll get:
 - AWS_ACCESS_KEY_ID

- **AWS_SECRET_ACCESS_KEY**

4. Configure the CLI

- Run this command in your terminal:

aws configure

- Enter the following when prompted:

Access Key ID

Secret Access Key

Default Region (e.g., ap-south-1 for Mumbai)

Default Output Format (e.g., json)

➤ AWS Configure

```
ec2-user@ip-172-31-12-93: ~ + - x Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Pranali kapare> ssh -i ..\Downloads\newkey.pem ec2-user@3.111.38.77
The authenticity of host '3.111.38.77' (3.111.38.77) can't be established.
ED25519 key fingerprint is SHA256:KNFW6y/FKyIV7NFebqxJwc3XbP3Y0uoXigCRR5vN2IQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.111.38.77' (ED25519) to the list of known hosts.

          #
          _###_      Amazon Linux 2023
~~ \_####\_
~~  \##|
~~   \|#
~~    \#/ ___  https://aws.amazon.com/linux/amazon-linux-2023
~~     \~` ' ->
~~      / \
~~     / \
~~    / \
~~   / \
[ec2-user@ip-172-31-12-93 ~]$ aws configure
AWS Access Key ID [None]: AKIAZ7SALHJKHDX7EUN3
AWS Secret Access Key [None]: 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
Default region name [None]: ap-south-1
Default output format [None]: json
[ec2-user@ip-172-31-12-93 ~]$ aws s3 ls
2025-03-13 16:00:42 lecturesbuck
2025-02-07 07:04:14 myterraformbuc
[ec2-user@ip-172-31-12-93 ~]$
```

➤ Create Custom VPC

```
> [ec2-user@ip-172-31-12-93 ~]$ aws ec2 create-vpc --cidr-block 10.0.0.0/16 --tag-specifications 'ResourceType=vpc,Tags=[{"Key=Name,Value=MyCustomVPC"}]'
{
    "Vpc": {
        "OwnerId": "686255979092",
        "InstanceTenancy": "default",
        "Ipv6CidrBlockAssociationSet": [],
        "CidrBlockAssociationSet": [
            {
                "AssociationId": "vpc-cidr-assoc-02dee1443ff867cfc",
                "CidrBlock": "10.0.0.0/16",
                "CidrBlockState": {
                    "State": "associated"
                }
            }
        ],
        "IsDefault": false,
        "Tags": [
            {
                "Key": "Name",
                "Value": "MyCustomVPC"
            }
        ],
        "VpcId": "vpc-04ede421a1a810adb",
        "State": "pending",
        "CidrBlock": "10.0.0.0/16",
        "DhcpOptionsId": "dopt-0fef4c16be8bf8d46"
    }
}
[ec2-user@ip-172-31-12-93 ~]$
```

➤ Create Public Subnet

```
[ec2-user@ip-172-31-12-93-] $ aws ec2 create-subnet --vpc-id vpc-04ede421ala810adb --cidr-block 10.0.1.0/20 --availability-zone ap-south-1a --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=PublicSubnet}]'
{
    "Subnet": {
        "AvailabilityZoneId": "aps1-az1",
        "OwnerId": "686255979092",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [],
        "Tags": [
            {
                "Key": "Name",
                "Value": "PublicSubnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:ap-south-1:686255979092:subnet/subnet-083f21bbb490cd2bd",
        "EnableDns64": false,
        "Ipv6Native": false,
        "PrivateDnsNameOptionsOnLaunch": {
            "HostnameType": "ip-name",
            "EnableResourceNameDnsARecord": false,
            "EnableResourceNameDnsAAAARecord": false
        },
        "SubnetId": "subnet-083f21bbb490cd2bd",
        "State": "available",
        "VpcId": "vpc-04ede421ala810adb",
        "CidrBlock": "10.0.1.0/20",
        "AvailableIpAddressCount": 4091,
        "AvailabilityZone": "ap-south-1a",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false
    }
}
```

➤ Create Private Subnet

```
[ec2-user@ip-172-31-12-93-] $ aws ec2 create-subnet --vpc-id vpc-04ede421ala810adb --cidr-block 10.0.16.0/20 --availability-zone ap-south-1b --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=PrivateSubnet}]'
{
    "Subnet": {
        "AvailabilityZoneId": "aps1-az3",
        "OwnerId": "686255979092",
        "AssignIpv6AddressOnCreation": false,
        "Ipv6CidrBlockAssociationSet": [],
        "Tags": [
            {
                "Key": "Name",
                "Value": "PrivateSubnet"
            }
        ],
        "SubnetArn": "arn:aws:ec2:ap-south-1:686255979092:subnet/subnet-0b6660420b6592986",
        "EnableDns64": false,
        "Ipv6Native": false,
        "PrivateDnsNameOptionsOnLaunch": {
            "HostnameType": "ip-name",
            "EnableResourceNameDnsARecord": false,
            "EnableResourceNameDnsAAAARecord": false
        },
        "SubnetId": "subnet-0b6660420b6592986",
        "State": "available",
        "VpcId": "vpc-04ede421ala810adb",
        "CidrBlock": "10.0.16.0/20",
        "AvailableIpAddressCount": 4091,
        "AvailabilityZone": "ap-south-1b",
        "DefaultForAz": false,
        "MapPublicIpOnLaunch": false
    }
}
```

➤ Create BastionHost

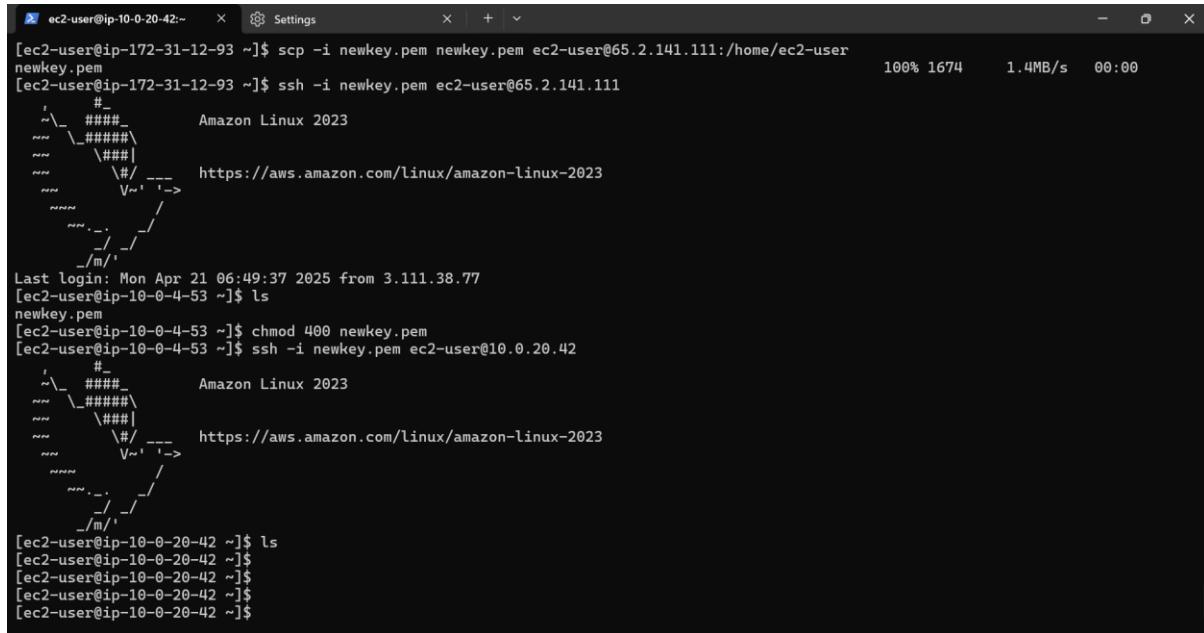
```
[ec2-user@ip-172-31-12-93-] $ aws ec2 run-instances --image-id ami-0fdcc636b69a6438 --instance-type t2.micro --key-name newkey --subnet-id subnet-083f21bbb490cd2bd --associate-public-ip-address --security-group-ids sg-04ad0614aeb85626b --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=BastionHost}]'
{
    "ReservationId": "r-08ce4162d8a7cbf14",
    "OwnerId": "686255979092",
    "Groups": [],
    "Instances": [
        {
            "Architecture": "x86_64",
            "BlockDeviceMappings": [],
            "ClientToken": "e09bab0c-585a-4e12-9f3b-73d0d946869d",
            "EbsOptimized": false,
            "EnaSupport": true,
            "Hypervisor": "xen",
            "NetworkInterfaces": [
                {
                    "Attachment": {
                        "AttachmentTime": "2025-04-21T06:34:43+00:00",
                        "AttachmentId": "eni-attach-00038fcdfc6f0a44",
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "Status": "attaching",
                        "NetworkCardIndex": 0
                    },
                    "Description": "",
                    "Groups": [
                        {
                            "GroupId": "sg-04ad0614aeb85626b",
                            "GroupName": "BastionSG"
                        }
                    ]
                }
            ]
        }
    ]
}
```

➤ Create Private Server



```
[ec2-user@ip-172-31-12-93 ~]$ aws ec2 run-instances --image-id ami-0f1dcc636b69a6438 --instance-type t2.micro --key-name newkey --subnet-id subnet-0b6660420b6592986 --security-group-ids sg-0ac0a5ffa0e04eaf9 --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=PrivateServer}]'
{
  "ReservationId": "r-0705e84fc423d89f",
  "OwnerId": "686255979092",
  "Groups": [],
  "Instances": [
    {
      "Architecture": "x86_64",
      "BlockDeviceMappings": [],
      "ClientToken": "f84f469d-bf03-45e8-a139-4d25fc20cf4",
      "EbsOptimized": false,
      "EnaSupport": true,
      "Hypervisor": "xen",
      "NetworkInterfaces": [
        {
          "Attachment": {
            "AttachTime": "2025-04-21T06:38:11+00:00",
            "AttachmentId": "eni-attach-06167dac17d9c56d1",
            "DeleteOnTermination": true,
            "DeviceIndex": 0,
            "Status": "attaching",
            "NetworkCardIndex": 0
          },
          "Description": "",
          "Groups": [
            {
              "GroupId": "sg-0ac0a5ffa0e04eaf9",
              "GroupName": "PrivateSG"
            }
          ],
          "Ipv6Addresses": []
        }
      ]
    }
  ]
}
```

➤ Connect Private Server with Bastion Host



```
[ec2-user@ip-172-31-12-93 ~]$ scp -i newkey.pem newkey.pem ec2-user@65.2.141.111:/home/ec2-user
newkey.pem                                          100% 1674     1.4MB/s   00:00
[ec2-user@ip-172-31-12-93 ~]$ ssh -i newkey.pem ec2-user@65.2.141.111
#_
# \_ #####_           Amazon Linux 2023
#~ \_ #####\_
#~ \###_
#~ \#/ _-- https://aws.amazon.com/linux/amazon-linux-2023
#~ V~' '-->
#~ .-' /
#~ /_/
#~ /m/
Last login: Mon Apr 21 06:49:37 2025 from 3.111.38.77
[ec2-user@ip-10-0-4-53 ~]$ ls
newkey.pem
[ec2-user@ip-10-0-4-53 ~]$ chmod 400 newkey.pem
[ec2-user@ip-10-0-4-53 ~]$ ssh -i newkey.pem ec2-user@10.0.20.42
#_
# \_ #####_           Amazon Linux 2023
#~ \_ #####\_
#~ \###_
#~ \#/ _-- https://aws.amazon.com/linux/amazon-linux-2023
#~ V~' '-->
#~ .-' /
#~ /_/
#~ /m/
[ec2-user@ip-10-0-20-42 ~]$ ls
[ec2-user@ip-10-0-20-42 ~]$
```

Task 3: Write a short note on S3. What is difference between Static website and Dynamic website?

Create a static website using S3 with minimum 3 pages having images and videos.

ANS:

Amazon S3 (Simple Storage Service)

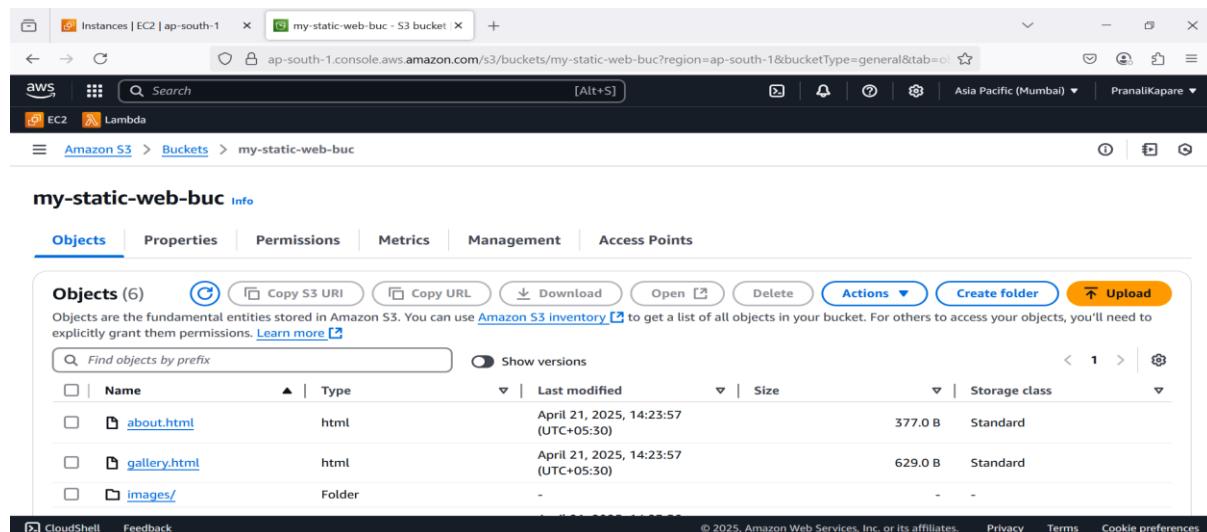
Amazon S3 is a scalable object storage service provided by AWS. It is used to store and retrieve any amount of data at any time, from anywhere on the web.

◆ Key Features:

- Stores objects (files) in buckets
- Supports versioning, encryption, and lifecycle policies
- Common use cases: backups, website hosting, media storage, big data analytics
- Offers high durability (99.99999999%) and availability

Difference Between Static Website and Dynamic Website

Feature	Static Website	Dynamic Website
Content	Same for all users	Changes based on user input or server logic
Technologies	HTML, CSS, JavaScript	PHP, Python, Node.js + Database (MySQL, MongoDB, etc.)
Hosting	Simple hosting (e.g., AWS S3)	Needs server (e.g., EC2, web hosting with backend)
Interactivity	Limited or none	High — based on user interactions
Use Cases	<td>E-commerce sites, dashboards, login-based applications</td>	E-commerce sites, dashboards, login-based applications
Speed	Very fast, no server processing needed	Slower, depends on server performance



Name	Type	Last modified	Size	Storage class
about.html	html	April 21, 2025, 14:23:57 (UTC+05:30)	377.0 B	Standard
gallery.html	html	April 21, 2025, 14:23:57 (UTC+05:30)	629.0 B	Standard
images/	Folder	-	-	-

The screenshot shows the AWS S3 console interface. At the top, there are tabs for 'Instances | EC2 | ap-south-1' and 'my-static-web-buc - S3 bucket'. The main content area is titled 'Amazon S3 > Buckets > my-static-web-buc'. Below this, a table lists the objects in the bucket:

Name	Type	Last modified	Size	Storage class
gallery.html	html	April 21, 2025, 14:23:57 (UTC+05:30)	629.0 B	Standard
images/	Folder	-	-	-
index.html	html	April 21, 2025, 14:23:56 (UTC+05:30)	390.0 B	Standard
style.css	css	April 21, 2025, 14:23:56 (UTC+05:30)	258.0 B	Standard
videos/	Folder	-	-	-

At the bottom of the page, there are links for 'CloudShell', 'Feedback', and copyright information: '© 2025, Amazon Web Services, Inc. or its affiliates.' followed by 'Privacy', 'Terms', and 'Cookie preferences'.

The screenshot shows the static website hosting configuration for the 'my-static-web-buc' bucket. The top navigation bar includes tabs for 'Instances | EC2 | ap-south-1' and 'my-static-web-buc - S3 bucket'. The main content area is titled 'Gallery | My Static Site'.

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

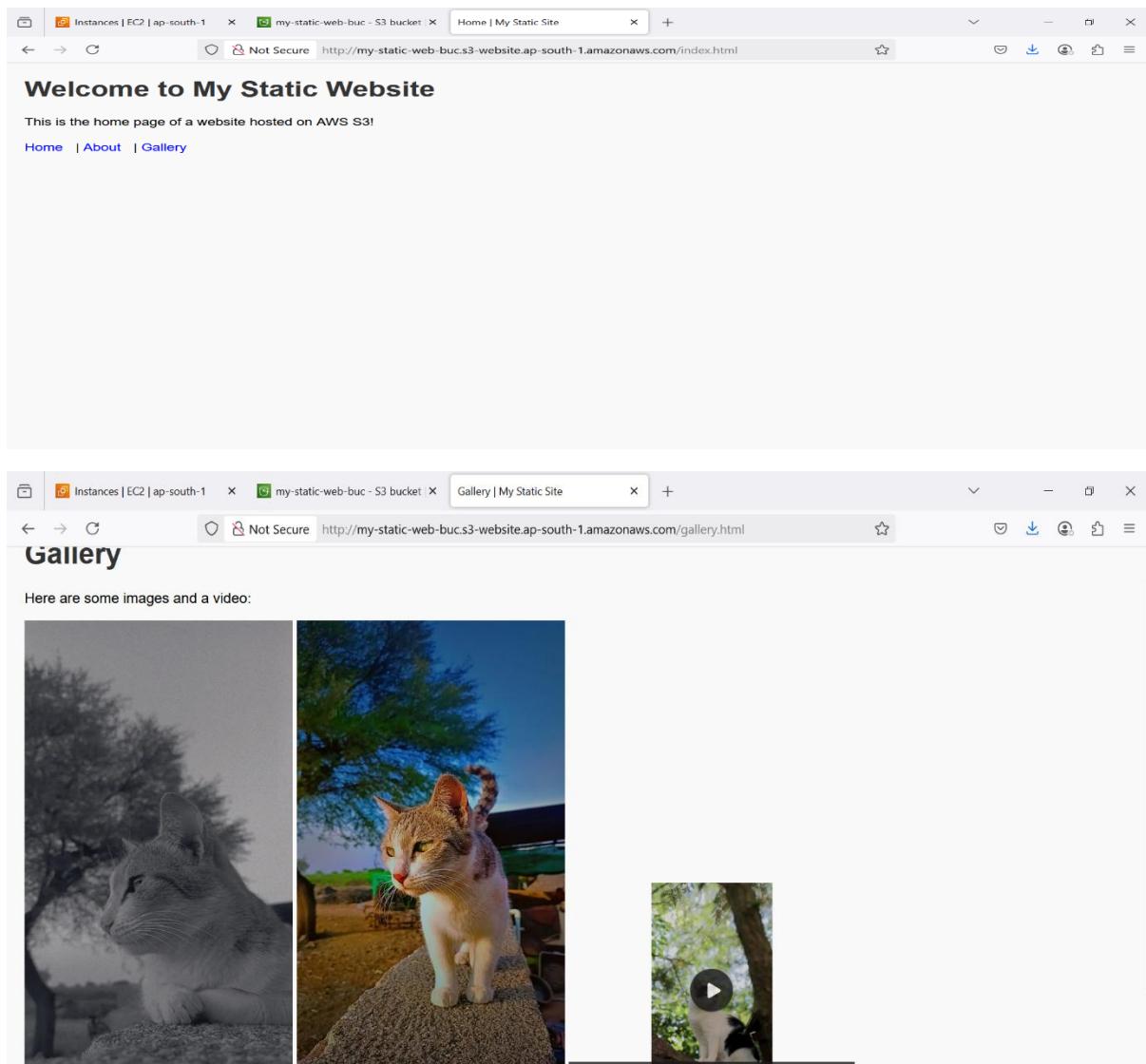
We recommend using AWS Amplify Hosting for static website hosting
Deploy a fast, secure, and reliable website quickly with AWS Amplify Hosting. Learn more about [Amplify Hosting](#) or [View your existing Amplify apps](#).

S3 static website hosting
Enabled

Hosting type
Bucket hosting

Bucket website endpoint
When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. [Learn more](#)
<http://my-static-web-buc.s3-website.ap-south-1.amazonaws.com>

At the bottom of the page, there are links for 'CloudShell', 'Feedback', and copyright information: '© 2025, Amazon Web Services, Inc. or its affiliates.' followed by 'Privacy', 'Terms', and 'Cookie preferences'.



Task 4: What is Instance Refresh? Demonstrate the use of Instance Refresh.

ANS:

Instance Refresh

Instance Refresh is a feature of **Amazon EC2 Auto Scaling Groups (ASG)** that enables you to **safely update the group's instances** without replacing or recreating the ASG itself.

It helps apply changes such as:

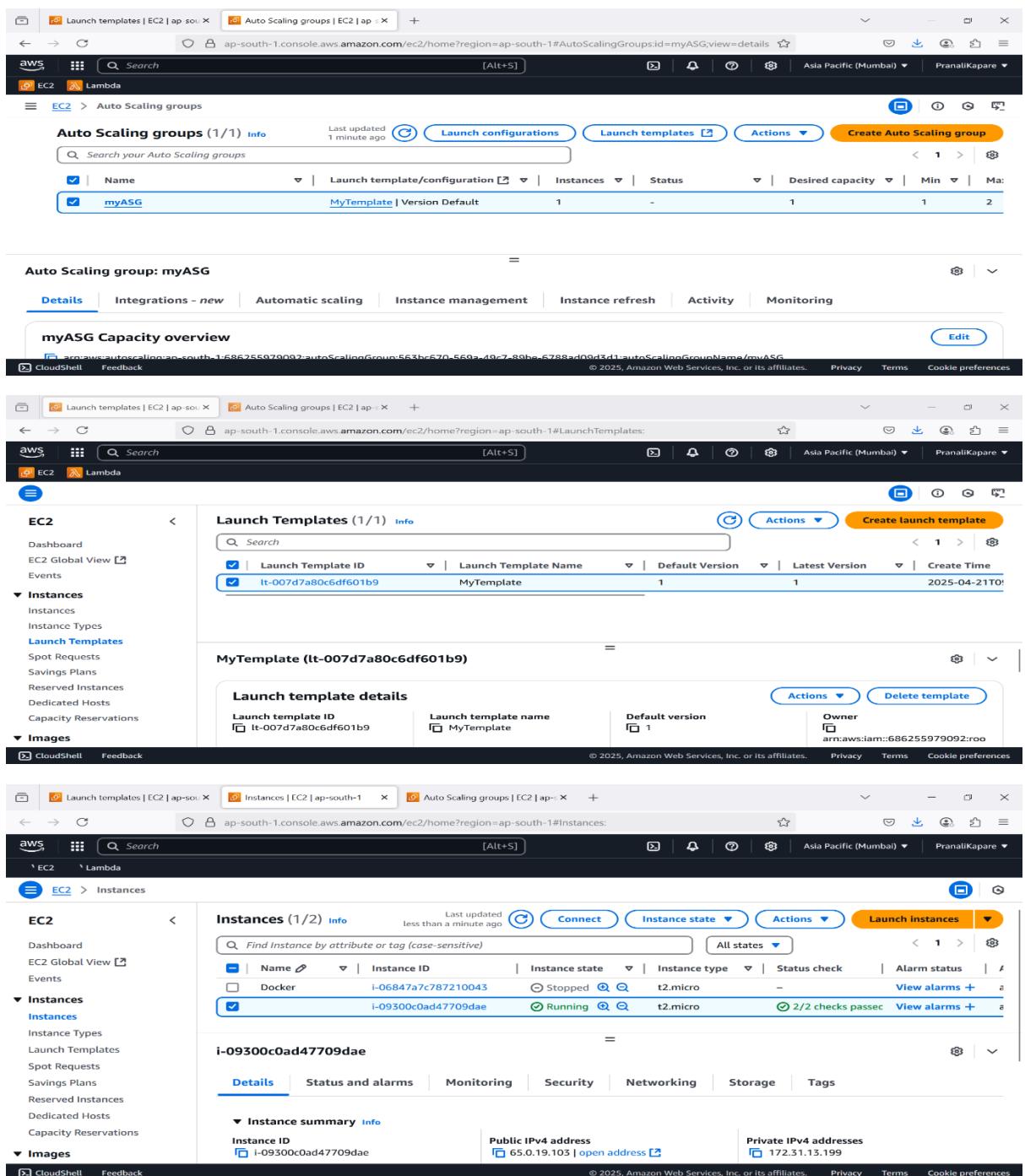
- New Amazon Machine Image (AMI)
- Updated launch template or launch configuration
- Configuration changes (e.g., instance type, user data)

The refresh happens in a **rolling update manner**, ensuring **high availability** and **minimal disruption**.

Demonstration: Using Instance Refresh via AWS Console

Make sure you already have:

- An **Auto Scaling Group (ASG)** created
- A **Launch Template** attached to it
- At least one running EC2 instance managed by the ASG



The image consists of three vertically stacked screenshots from the AWS EC2 console.

Screenshot 1: Auto Scaling groups

This screenshot shows the "Auto Scaling groups" page with one entry named "myASG". The "Launch template/configuration" column shows "MyTemplate | Version Default". The "Instances" column shows 1 instance.

Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max
myASG	MyTemplate Version Default	1	-	1	1	2

Screenshot 2: myASG Capacity overview

This screenshot shows the "myASG Capacity overview" page. It includes tabs for Details, Integrations - new, Automatic scaling, Instance management, Instance refresh, Activity, and Monitoring. The "Instance refresh" tab is active.

Screenshot 3: Launch Templates

This screenshot shows the "Launch Templates" page with one entry named "MyTemplate (lt-007d7a80c6df601b9)". The "Launch template details" section shows the launch template ID and name.

Launch template ID	Launch template name	Default version	Latest version	Create Time
lt-007d7a80c6df601b9	MyTemplate	1	1	2025-04-21T01:54:22Z

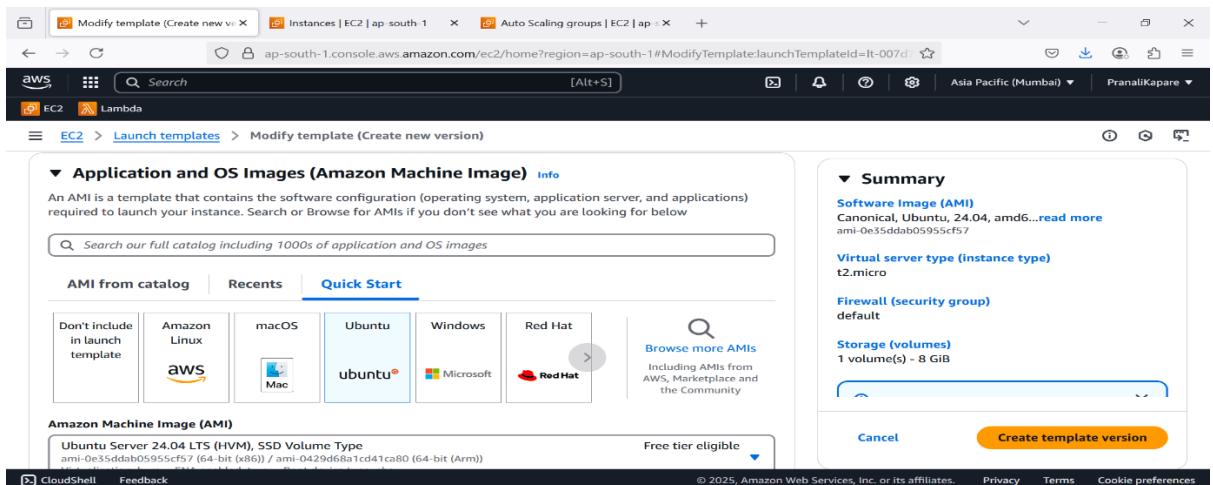
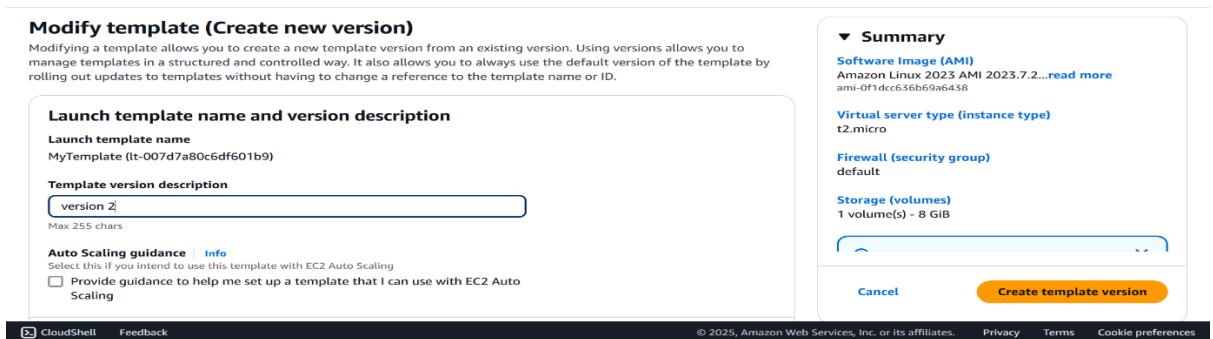
Screenshot 4: Instances

This screenshot shows the "Instances" page with two instances listed: "Docker" (Stopped) and "i-09300c0ad47709dae" (Running). The "Details" tab is active.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
Docker	i-06847a7c787210043	Stopped	t2.micro	-	View alarms
	i-09300c0ad47709dae	Running	t2.micro	2/2 checks passed	View alarms

Step 1: Update Launch Template

1. Go to **EC2 Dashboard → Launch Templates**
2. Select your existing Launch Template
3. Click on **Actions → Create new version**
4. Update configuration (e.g., use a new AMI)
5. Click **Create launch template version**



Step 2: Attach Updated Launch Template to ASG

1. Go to **EC2 Dashboard → Auto Scaling Groups**
2. Click on your Auto Scaling Group
3. Under the **Details** tab, click **Edit**
4. Select the new **Launch Template version** (e.g., Version 2)
5. Click **Update**

The screenshot shows the AWS EC2 Auto Scaling Groups page. In the top navigation bar, there are tabs for Launch templates, Instances, and Edit Auto Scaling group. The main content area is titled 'Edit myASG'. A sub-section titled 'Launch template' is highlighted with a blue border. Inside this section, it says 'For accounts created after May 31, 2023, the EC2 console only supports creating Auto Scaling groups with launch templates. Creating Auto Scaling groups with launch configurations is not recommended but still available via the CLI and API until December 31, 2023.' Below this message, there is a dropdown menu set to 'MyTemplate', a link to 'Create a launch template', and a dropdown for 'Version' set to '2'. To the right of the dropdown, there is a 'Launch template' card with the name 'MyTemplate' and ID 'lt-007d7a80c6df601b9'. Further right is an 'Instance type' card showing 't2.micro'. At the bottom of the page, there are links for CloudShell, Feedback, and a footer with copyright information and links for Privacy, Terms, and Cookie preferences.

✓ Step 3: Start Instance Refresh

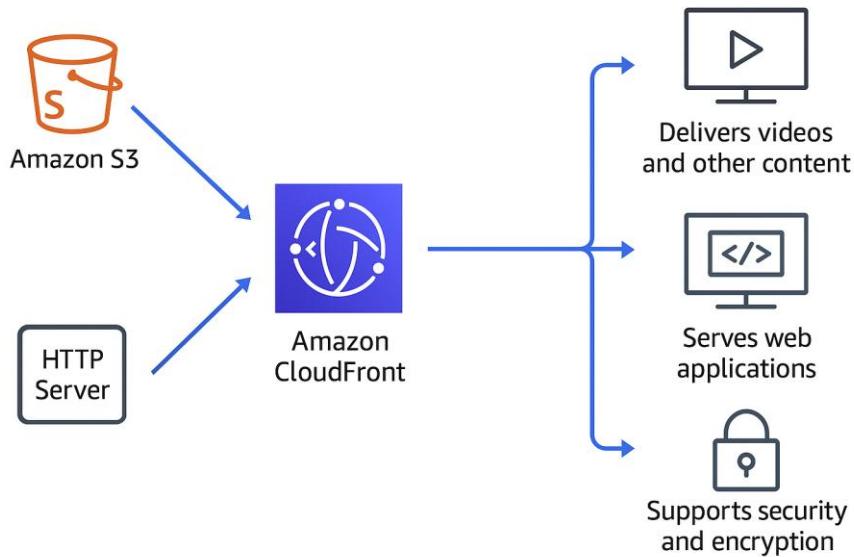
1. Inside your Auto Scaling Group page
2. Navigate to the **Instance Refresh** tab
3. Click **Start instance refresh**
4. You can configure:
 - **Minimum healthy percentage** (e.g., 90%)
 - **Instance warm-up time** (e.g., 20 seconds)
5. Click **Start**

The screenshot shows the AWS EC2 Auto Scaling Groups page. The top navigation bar has tabs for Launch templates, Instances, and Auto Scaling group details. The main content area shows a green banner at the top stating 'Instance refresh started successfully'. Below this, there are tabs for Details, Integrations - new, Automatic scaling, Instance management, **Instance refresh**, Activity, and Monitoring. The 'Instance refresh' tab is selected. Underneath, there is a section titled 'Active instance refresh' with a sub-section 'Info'. This section contains several configuration options with their current values:

Setting	Value	Setting	Value
Instance refresh ID	3e12c844-f828-4ec5-844a-7f70b152c422	Minimum healthy percentage	90%
Instance refresh status	Pending	Instance warmup	20 seconds
Start time	-	Bake time	-
CloudWatch alarm	-	Checkpoints	-
Skip matching	Enabled	Desired configuration	View
Scale-in protected instances	Ignore	Auto rollback	Disabled
Standby instances	Ignore		

At the bottom of the page, there are links for CloudShell, Feedback, and a footer with copyright information and links for Privacy, Terms, and Cookie preferences.

Task 5: Create a architectural diagram of uses of cloud front.



SET D

1. How to install Prometheus and Grafana on you cluster using Terraform and helm.

ANS:

- Install Kubernetes

On Master Node:

```
ubuntu@ip-172-31-47-6:~$ kubectl get nodes
NAME           STATUS    ROLES      AGE     VERSION
ip-172-31-46-213  NotReady <none>    15s    v1.29.15
ip-172-31-47-6   Ready     control-plane 3m18s   v1.29.15
ubuntu@ip-172-31-47-6:~$ kubectl get nodes
NAME           STATUS    ROLES      AGE     VERSION
ip-172-31-46-213  NotReady <none>    32s    v1.29.15
ip-172-31-47-6   Ready     control-plane 3m35s   v1.29.15
ubuntu@ip-172-31-47-6:~$ kubectl get nodes
NAME           STATUS    ROLES      AGE     VERSION
ip-172-31-46-213  Ready    <none>    3m22s   v1.29.15
ip-172-31-47-6   Ready     control-plane 6m25s   v1.29.15
ubuntu@ip-172-31-47-6:~$
```

On Worker Node:

```
ubuntu@ip-172-31-47-6:~ x  ubuntu@ip-172-31-46-213:~ x + v
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.

kubebundle set on hold.
kubeadm set on hold.
kubectl set on hold.

Kubernetes setup completed.

ubuntu@ip-172-31-46-213:~$ sudo kubeadm join 172.31.47.6:6443 --token v8bteu.9zjxos522jn2bulq --discovery-token-ca-cert-hash sha256:2
be5ed0289282b66ded87f9d80aec321e03f5e745bd767f764f583a41c19dbdf
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

ubuntu@ip-172-31-46-213:~$
```

➤ Terraform and Helm Installation:

```
ubuntu@ip-172-31-47-6:~ x  ubuntu@ip-172-31-46-213:~ x + v
Selecting previously unselected package terraform.
(Reading database ... 70637 files and directories currently installed.)
Preparing to unpack .../terraform_1.11.4-1_amd64.deb ...
Unpacking terraform (1.11.4-1) ...
Setting up terraform (1.11.4-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.

ubuntu@ip-172-31-47-6:~$ terraform --version
Terraform v1.11.4
on linux_amd64
ubuntu@ip-172-31-47-6:~$ curl https://get.helm.sh/helm-v3.9.0-linux-amd64.tar.gz -o helm-v3.9.0-linux-amd64.tar.gz
% Total    % Received % Xferd  Average Speed   Time   Time  Current
          Dload  Upload   Total   Spent    Left  Speed
100 13.3M  100 13.3M    0      0  12.2M     0:00:01  0:00:01  --:--:-- 12.2M
ubuntu@ip-172-31-47-6:~$ tar -zxfv helm-v3.9.0-linux-amd64.tar.gz
linux-amd64/
linux-amd64/helm
linux-amd64/LICENSE
linux-amd64/README.md
ubuntu@ip-172-31-47-6:~$ sudo mv linux-amd64/helm /usr/local/bin/helm
ubuntu@ip-172-31-47-6:~$ helm version
version.BuildInfo{Version:"v3.9.0", GitCommit:"7ceeda6c585217a19a1131663d8cd1f7d641b2a7", GitTreeState:"clean", GoVersion:"go1.17.5"}
ubuntu@ip-172-31-47-6:~$
```



```
ubuntu@ip-172-31-47-6:~/pr x + v
ubuntu@ip-172-31-47-6:~$ mkdir promGraf
ubuntu@ip-172-31-47-6:~$ cd promGraf/
ubuntu@ip-172-31-47-6:~/promGraf$ nano main.tf
ubuntu@ip-172-31-47-6:~/promGraf$ terraform init
Terraform has no command named "iniy". Did you mean "init"?
To see all of Terraform's top-level commands, run:
  terraform -help

ubuntu@ip-172-31-47-6:~/promGraf$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/helm versions matching "> 2.7"...
- Finding hashicorp/kubernetes versions matching "> 2.0"...
- Installing hashicorp/helm v2.7.0...
- Installed hashicorp/helm v2.17.1 (signed by HashiCorp)
- Installing hashicorp/kubernetes v2.37.1...
- Installed hashicorp/kubernetes v2.37.1 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider selections it made above. Include this file in your version control repository so that Terraform can guarantee to make the same selections by default when you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.
ubuntu@ip-172-31-47-6:~/promGraf$
```

Main.tf

```
terraform {
  required_providers {
    kubernetes = {
      source  = "hashicorp/kubernetes"
      version = "~> 2.0"
    }
    helm = {
      source  = "hashicorp/helm"
      version = "~> 2.7"
    }
  }
}

provider "kubernetes" {
  config_path = "~/.kube/config"
}

provider "helm" {
  kubernetes {
    config_path = "~/.kube/config"
  }
}

resource "helm_repository" "prometheus" {
  name = "prometheus-community"
  url = "https://prometheus-community.github.io/helm-charts"
}
resource "helm_repository" "grafana" {
  name = "grafana"
  url = "https://grafana.github.io/helm-charts"
}

resource "kubernetes_namespace" "monitoring" {
  metadata {
    name = "monitoring"
  }
}
```

```
resource "helm_release" "prometheus_stack" {
    name      = "kube-prometheus-stack"
    repository = helm_repository.prometheus.url
    chart      = "kube-prometheus-stack"
    namespace   = kubernetes_namespace.monitoring.metadata[0].name
    create_namespace = false

    values = [
        file("prometheus-values.yaml") # optional, remove if you don't have this file
    ]

    depends_on = [kubernetes_namespace.monitoring]
}

resource "helm_release" "grafana" {
    name      = "grafana"
    repository = helm_repository.grafana.url
    chart      = "grafana"
    namespace   = kubernetes_namespace.monitoring.metadata[0].name
    create_namespace = false

    values = [
        file("grafana-values.yaml")
    ]

    depends_on = [kubernetes_namespace.monitoring]
}
```

➤ Prometheus Port Forwarding

```

ubuntu@ip-172-31-47-6:~ x  ubuntu@ip-172-31-47-6:~ x + | v
Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows
PS C:\Users\Pranali kapare> ssh -i ..\Downloads\newkey.pem -L 9090:localhost:9090 ubuntu@52.66.204.141
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1028-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Fri May 23 09:51:01 UTC 2025
System load: 0.7          Processes:           168
Usage of /: 30.8% of 18.33GB   Users logged in: 1
Memory usage: 30%          IPv4 address for enX0: 172.31.47.6
Swap usage: 0%
* Ubuntu Pro delivers the most comprehensive open source security and
  compliance features.
  https://ubuntu.com/aws/pro

Expanded Security Maintenance for Applications is not enabled.

55 updates can be applied immediately.
12 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

Last login: Fri May 23 09:20:58 2025 from 152.58.30.178
ubuntu@ip-172-31-47-6:~$
```

Access Prometheus UI on Browser

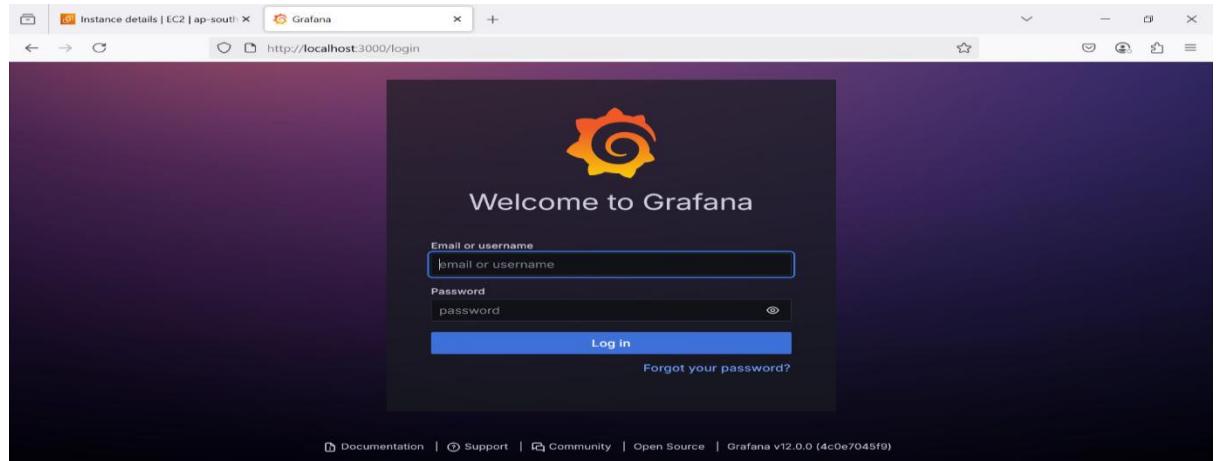
The screenshot shows a web browser window with the URL `localhost:9090/query`. The page is titled "Prometheus Time Series Collector". At the top, there are tabs for "Instance details | EC2 | ap-south" and a search bar. Below the search bar are buttons for "query", "Alerts", and "Status". A dropdown menu labeled "Evaluation time" is open. At the bottom of the page, a message reads "No data queried yet".

➤ Grafana Port Forwarding

```

ubuntu@ip-172-31-47-6:~ x  ubuntu@ip-172-31-47-6:~ x + | v
name: kube-prometheus-stack-grafana
namespace: monitoring
resourceVersion: "282075"
uid: 5601abfb-2391-4231-98bb-f1b8c48414ed
spec:
  clusterIP: 10.100.94.51
  clusterIPs:
  - 10.100.94.51
  internalTrafficPolicy: Cluster
  ipFamilies:
  - IPv4
  ipFamilyPolicy: SingleStack
  ports:
  - name: http-web
    port: 80
    protocol: TCP
    targetPort: 3000
  selector:
    app.kubernetes.io/instance: kube-prometheus-stack
    app.kubernetes.io/name: grafana
  sessionAffinity: None
  type: ClusterIP
status:
  loadBalancer: {}
ubuntu@ip-172-31-47-6:~$ kubectl get pods -n monitoring | grep grafana
kube-prometheus-stack-grafana-598f4fff786-2f74g      3/3     Running   0          17m
ubuntu@ip-172-31-47-6:~$ kubectl port-forward kube-prometheus-stack-grafana-598f4fff786-2f74g 3000:3000 -n monitoring
Forwarding from 127.0.0.1:3000 -> 3000
Forwarding from [::]:3000 -> 3000
Handling connection for 3000
Handling connection for 3000
Handling connection for 3000
```

Access Grafana UI on Browser



2. Using Ansible to deploy a static web application onto a remote server.

ANS:

➤ Install Ansible

```
ubuntu@ip-172-31-42-3:~ x + - Setting up python3-kerberos (1.1.14-3.1build9) ... Setting up ansible-core (2.18.5-1ppa-noble) ... Setting up sshpass (1.09-1) ... Setting up python3-xmldictodict (0.13.0-1) ... Setting up ansible (11.4.0-1ppa-noble) ... Setting up python3-nacl (1.5.0-4build1) ... Setting up python3-requests_ntlm (1.1.0-3) ... Setting up python3-winrm (0.4.3-2) ... Setting up python3-paramiko (2.12.0-2ubuntu4.1) ... Processing triggers for man-db (2.12.0-4build2) ... Scanning processes... Scanning linux images... Running kernel seems to be up-to-date. No services need to be restarted. No containers need to be restarted. No user sessions are running outdated binaries. No VM guests are running outdated hypervisor (qemu) binaries on this host. ubuntu@ip-172-31-42-3:~$ ansible --version ansible [core 2.18.5] config file = /etc/ansible/ansible.cfg configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules'] ansible python module location = /usr/lib/python3/dist-packages/ansible ansible collection location = /home/ubuntu/.ansible/collections:/usr/share/ansible/collections executable location = /usr/bin/ansible python version = 3.12.3 (main, Feb 4 2025, 14:48:35) [GCC 13.3.0] (/usr/bin/python3) jinja version = 3.1.2 libyaml = True ubuntu@ip-172-31-42-3:~$
```

➤ Myhost.ini

```
ubuntu@ip-172-31-42-3: ~$ nano /etc/ansible/hosts  
[remote]  
3.198.223.240 ansible_ssh_user='ubuntu' ansible_private_key_file="newkey.pem"  
localhost ansible_connection=local  
myhost.ini
```

➤ Playbook.yml

```
GNU nano 7.2                                         playbook.yml
- name: Deploy Static Web App on Local and Remote
  hosts: all
  become: yes

  tasks:
    - name: Install Nginx
      apt:
        name: nginx
        state: present
        update_cache: yes

    - name: Copy website files
      copy:
        src: ./webpage/index.html
        dest: /var/www/html/index.html
        owner: www-data
        group: www-data
        mode: '0644'

    - name: Ensure Nginx is running
      service:
        name: nginx
        state: started
        enabled: yes
```

The terminal window shows the 'playbook.yml' file in the nano text editor. The file contains an Ansible playbook with three tasks: installing Nginx, copying a static website file, and ensuring Nginx is running. The terminal has a standard nano keymap at the bottom.

➤ Index.html

```
GNU nano 7.2                                         index.html
<!DOCTYPE html>
<html>
<head><title>Ansible Demo</title></head>
<body>
  <h1>Hello from Pranali's Ansible Deploy!</h1>
</body>
</html>
```

The terminal window shows the 'index.html' file in the nano text editor. It contains a simple HTML document with a single heading. The terminal has a standard nano keymap at the bottom.

➤ Execute the Ansible Playbook

```
ubuntu@ip-172-31-42-3:~/static-web$ nano playbook.yml
ubuntu@ip-172-31-42-3:~/static-web$ ansible-playbook -i myhost.ini playbook.yml
PLAY [Deploy Static Web App on Local and Remote] ****
TASK [Gathering Facts] ****
[WARNING]: Platform Linux on host localhost is using the discovered Python interpreter at /usr/bin/python3.12, but future
installation of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.18/reference_appendices/interpreter_discovery.html for more information.
ok: [localhost]
[WARNING]: Platform Linux on host 3.108.223.240 is using the discovered Python interpreter at /usr/bin/python3.12, but future
installation of another Python interpreter could change the meaning of that path. See https://docs.ansible.com/ansible-
core/2.18/reference_appendices/interpreter_discovery.html for more information.
ok: [3.108.223.240]

TASK [Install Nginx] ****
ok: [localhost]
ok: [3.108.223.240]

TASK [Copy website files] ****
changed: [localhost]
changed: [3.108.223.240]

TASK [Ensure Nginx is running] ****
ok: [localhost]
ok: [3.108.223.240]

PLAY RECAP ****
3.108.223.240 : ok=4   changed=1   unreachable=0   failed=0    skipped=0   rescued=0   ignored=0
localhost     : ok=4   changed=1   unreachable=0   failed=0    skipped=0   rescued=0   ignored=0
```

The terminal window shows the execution of the playbook. It gathers facts, installs Nginx, copies the website file, and ensures Nginx is running on both the local host and the remote host (IP 3.108.223.240). The output shows the tasks being run and their results.

➤ Output on Browser



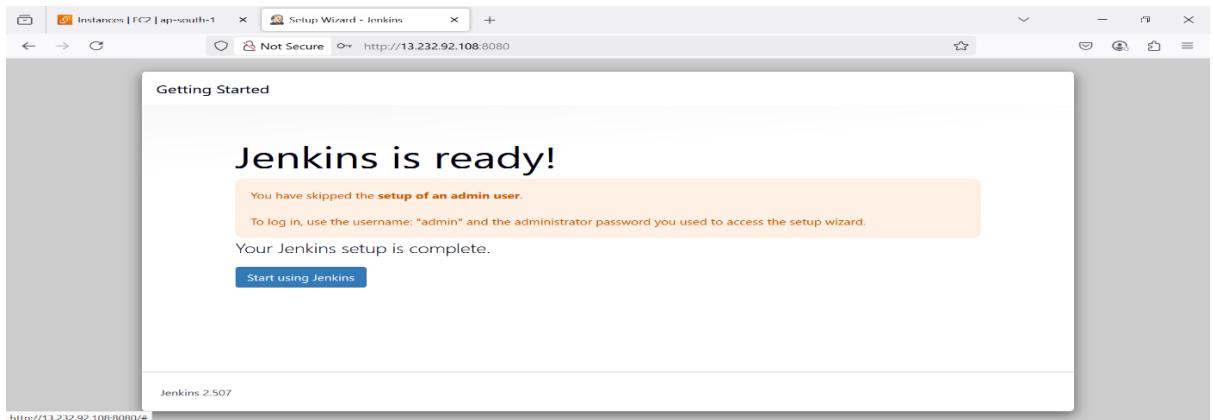
3. Jenkins

- Install Jenkins from scratch on an ec2 instance with default plulgins.

➤ Install Jenkins

```
ubuntu@ip-172-31-5-104:~ $ sudo systemctl start jenkins
ubuntu@ip-172-31-5-104:~ $ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
  Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
  Active: active (running) since Thu 2025-04-24 06:39:08 UTC; 3s ago
    Main PID: 4614 (java)
       Tasks: 39 (limit: 1129)
      Memory: 331.0M (peak: 337.2M)
        CPU: 17.873s
       CGroup: /system.slice/jenkins.service
               └─4614 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Apr 24 06:38:58 ip-172-31-5-104 jenkins[4614]: f64a1d0915fb468eb6dbede73090f9f9
Apr 24 06:38:58 ip-172-31-5-104 jenkins[4614]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Apr 24 06:38:58 ip-172-31-5-104 jenkins[4614]: ****
Apr 24 06:39:08 ip-172-31-5-104 jenkins[4614]: 2025-04-24 06:39:08.037+0000 [id=32]      INFO      jenkins.InitReactorRunner$1#>
Apr 24 06:39:08 ip-172-31-5-104 jenkins[4614]: 2025-04-24 06:39:08.066+0000 [id=24]      INFO      hudson.lifecycle.Lifecycle#on>
Apr 24 06:39:08 ip-172-31-5-104 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Apr 24 06:39:09 ip-172-31-5-104 jenkins[4614]: 2025-04-24 06:39:09.673+0000 [id=47]      INFO      h.m.DownloadService$Download&gt;
Apr 24 06:39:09 ip-172-31-5-104 jenkins[4614]: 2025-04-24 06:39:09.674+0000 [id=47]      INFO      hudson.util.Retrier#start: Pe>
ubuntu@ip-172-31-5-104:~ $ jenkins --version
2.492.3
ubuntu@ip-172-31-5-104:~ $
ubuntu@ip-172-31-5-104:~ $
```



b. Create 3 users in Alice/Bob/Mary with different roles.

➤ Install Plugin

A screenshot of the Jenkins 'Available plugins' page. The title bar says 'Instances | EC2 | ap-south-1' and 'Available plugins - Plugins - Jenkins'. The main content area shows a sidebar with 'Updates', 'Available plugins' (selected), 'Installed plugins', 'Advanced settings', and 'Download progress'. A search bar at the top right contains the text 'role'. Below is a table listing two plugins:

Install	Name	Released
<input checked="" type="checkbox"/>	Role-based Authorization Strategy 756.v978cb_392eb_d3 Security Authentication and User Management Enables user authorization using a Role-Based strategy. Roles can be defined globally or for particular jobs or nodes selected by regular expressions.	2 mo 26 days ago
<input type="checkbox"/>	AWS Credentials 245.v8a_1b_7c11a_94d aws Allows storing Amazon IAM credentials within the Jenkins Credentials API. Store Amazon IAM access keys (AWSAccessKeyId and AWSSecretKey) within the Jenkins Credentials API. Also support IAM Roles and IAM MFA Token.	1 mo 14 days ago

At the bottom right are 'REST API' and 'Jenkins 2.507' links.

➤ Set Authorization

A screenshot of the Jenkins 'Security' configuration page. The title bar says 'Instances | EC2 | ap-south-1' and 'Security - Jenkins'. The main content area shows a 'Security Realm' section with 'Jenkins' own user database selected. Below is an 'Authorization' section with 'Role-Based Strategy' selected. Further down is a 'Markup Formatter' section with 'Plain text' selected. At the bottom are 'Save' and 'Apply' buttons.

➤ Manage Roles

The screenshot shows the Jenkins Manage Roles interface. At the top, there are links for 'Assign Roles', 'Permission Templates', and 'Role Strategy Macros'. Below this is a 'Global roles' section containing a large grid table. The columns represent various Jenkins management tasks: Overall, Credentials, Agent, Job, Run, and View. The rows represent user roles: Administer, Read, Create, Delete, Provision, Connect, Configure, Build, Discover, Create, Delete, Configure, Cancel, and View. The 'admin' role has checked boxes in almost all columns, while 'developer' and 'viewer' have checked boxes in fewer columns. A 'Role to add' input field and 'Save' and 'Apply' buttons are at the bottom.

➤ Create Users

The screenshot shows the Jenkins Users page. It displays a table of four users: admin, Alice, bob, and marry. The columns are 'User ID' and 'Name'. Each user has a profile icon, a settings gear icon, and a trash can icon. A 'Create User' button is located at the top right. A note at the bottom states: "These users can log into Jenkins. This is a sub set of [this list](#), which also contains auto-created users who really just made some commits on some projects and have no direct Jenkins access."

Jenkins 2.507

➤ Assign Roles to Users

The screenshot shows the Jenkins Assign Roles page. It features a 'User/Group' column on the left with categories for Anonymous, Authenticated Users, and specific users: admin, Alice, bob, and marry. To the right is a grid where each user/group row has a column for each role listed in the 'Manage Roles' table. The 'admin' user has checked boxes in every column. The 'viewer' role is present in the first three columns. The 'Save' and 'Apply' buttons are at the bottom.

c. create free style project and execute various Linux commands in Build > Execute shell

The image consists of three vertically stacked screenshots of the Jenkins web interface.

Screenshot 1: New Item - Jenkins

This screenshot shows the "New Item" creation page. A search bar at the top contains the text "shell_commands". Below it, a section titled "Select an item type" lists three options:

- Freestyle project**: Described as a "Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications."
- Pipeline**: Described as "Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type."
- Multi-configuration project**: Described as "Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc."

A blue "OK" button is at the bottom right.

Screenshot 2: Configuration - shell_commands - Jenkins

This screenshot shows the configuration page for the "shell_commands" job. The left sidebar has "Build Steps" selected. The main area is titled "Execute shell" and contains a command block with the following content:

```
ls -la
echo "Hello"
echo "$(pwd)"
echo "$(whoami)"
```

Buttons for "Save" and "Apply" are at the bottom.

Screenshot 3: shell_commands - Jenkins

This screenshot shows the main project page for "shell_commands". The left sidebar includes "Status", "Changes", "Workspace", "Build Now", "Configure", "Delete Project", and "Rename". The main area displays the "Permalinks" section with a bulleted list of recent builds and a "Builds" summary table with one entry:

Build	Timestamp
#1	8:02 AM

At the bottom right, there are links for "REST API" and "Jenkins 2.507".

The screenshot shows the Jenkins interface with the job 'shell_commands' running. The 'Console Output' tab is selected. The output window displays the following terminal session:

```

Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/shell_commands
[shell_commands] $ /bin/sh -xe /tmp/jenkins7823474016499936576.sh
+ ls -la
total 8
drwxr-xr-x 2 jenkins jenkins 4096 Apr 24 08:02 .
drwxr-xr-x 3 jenkins jenkins 4096 Apr 24 08:02 ..
+ echo Hello
Hello
+ pwd
+ echo /var/lib/jenkins/workspace/shell_commands
/var/lib/jenkins/workspace/shell_commands
+ whoami
+ echo Jenkins
Jenkins
Finished: SUCCESS

```

- d. install a git plugin and configure repo [LuisJoseSanchez/hello-world-java: Hello world with Java \(github.com\)](#) into it.

The screenshot shows the Jenkins 'Manage Jenkins' section under 'Plugins'. The 'Installed plugins' tab is selected. A search bar at the top contains 'git'. The results show the 'Git plugin' is installed and enabled (blue switch).

Plugin	Status	Action
Git plugin 5.7.0	<input checked="" type="checkbox"/>	Uninstall
GitHub API Plugin 1.321-488.v9b_c0da_9533f8	<input checked="" type="checkbox"/>	Uninstall
GitHub Branch Source Plugin 1815.v9152b_2ff7a_1b_	<input checked="" type="checkbox"/>	Uninstall
GitHub plugin 1.43.0	<input checked="" type="checkbox"/>	Uninstall
Pipeline: GitHub Groovy Libraries 65.v203688e7727e	<input checked="" type="checkbox"/>	Uninstall

The screenshot shows the Jenkins configuration page for the 'shell_commands' job. The 'Source Code Management' section is active. Under 'General', the 'Source Code Management' tab is selected. The 'Git' option is chosen. In the 'Repositories' section, the 'Repository URL' is set to 'https://github.com/LuisJoseSanchez/hello-world-java.git' and the 'Credentials' dropdown is set to '- none -'. At the bottom, there are 'Save' and 'Apply' buttons.

- e. Execute the commands given in the repo description and the output should be printed into jenkins console.

The screenshots show the Jenkins interface for a job named "shell_commands".

- Screenshot 1:** Shows the Jenkins dashboard for the "shell_commands" job. It displays the build history with three entries: #3 (Success at 8:21 AM), #2 (Failure at 8:15 AM), and #1 (Success at 8:02 AM). The sidebar shows options like Status, Changes, Workspace, Build Now, Configure, Delete Project, and Rename.
- Screenshot 2:** Shows the "Console Output" for build #3. The log output is as follows:

```

Started by user admin
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/shell_commands
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/shell_commands/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/LuisJoseSanchez/hello-world-java.git # timeout=10
Fetching upstream changes from https://github.com/LuisJoseSanchez/hello-world-java.git
> git --version # timeout=10
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/LuisJoseSanchez/hello-world-java.git +refs/heads/
*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 4947e1e9e32f20606f186e5f7267286b848beb3e (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 4947e1e9e32f20606f186e5f7267286b848beb3e # timeout=10
Commit message: "Updating README.md"
> git rev-list --no-walk 4947e1e9e32f20606f186e5f7267286b848beb3e # timeout=10

```

- Screenshot 3:** Shows the "Console Output" for build #3. The log output is as follows:

```

The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/shell_commands/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/LuisJoseSanchez/hello-world-java.git # timeout=10
Fetching upstream changes from https://github.com/LuisJoseSanchez/hello-world-java.git
> git --version # timeout=10
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/LuisJoseSanchez/hello-world-java.git +refs/heads/
*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 4947e1e9e32f20606f186e5f7267286b848beb3e (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 4947e1e9e32f20606f186e5f7267286b848beb3e # timeout=10
Commit message: "Updating README.md"
> git rev-list --no-walk 4947e1e9e32f20606f186e5f7267286b848beb3e # timeout=10
[shell_commands] $ /bin/sh -xe /tmp/jenkins3541632691695666737.sh
+ javac HelloWorld.java
+ java HelloWorld
Hello world!
Finished: SUCCESS

```

- f. Fork the repo, changes the description in print statement into code and rerun the Jenkins job the updated output should be shown into jenkins console log.

The image consists of three vertically stacked screenshots from a web browser showing GitHub operations:

- Screenshot 1: Forking a Repository**
A screenshot of a browser window with multiple tabs. The active tab shows the URL github.com/LuisJoseSanchez/hello-world-java/fork. The page title is "Create a new fork". It asks for the owner ("PranaliKapare") and repository name ("hello-world-java"). A note says "hello-world-java is available". Below, there's a "Description (optional)" field containing "Hello world with Java" and a checked checkbox for "Copy the master branch only".
- Screenshot 2: Repository Details**
A screenshot of a browser window showing the forked repository at github.com/PranaliKapare/hello-world-java/blob/master/HelloWorld.java. The code editor shows the Java file `HelloWorld.java` with the following content:

```
1 public class HelloWorld {
2     public static void main(String[] args) {
3         System.out.println("Hello from my forked repo!");
4     }
5 }
```
- Screenshot 3: Repository Overview**
A screenshot of a browser window showing the forked repository at github.com/PranaliKapare/hello-world-java. The repository summary shows it is a fork of [LuisJoseSanchez/hello-world-java](https://github.com/LuisJoseSanchez/hello-world-java). It has 1 commit ahead of the master branch. The repository details include the README, .gitignore, and HelloWorld.java file. The right sidebar shows the repository's activity, including 6 commits by PranaliKapare, 0 stars, 0 forks, and 0 releases.

Instances | EC2 | ap-south-1 × **shell_commands Config Jenkins** × **PranaliKapare/hello-world-java: x** +

Not Secure http://13.232.92.108:8080/job/shell_commands/configure

Jenkins / shell_commands / Configuration

Configure

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

None

Git ?

Repositories ?

Repository URL ?
https://github.com/PranaliKapare/hello-world-java

Credentials ?
- none -

+ Add

Save **Apply**

Instances | EC2 | ap-south-1 × **shell_commands - Jenkins** × **PranaliKapare/hello-world-java: x** +

Not Secure http://13.232.92.108:8080/job/shell_commands/

Jenkins / shell_commands

Status

</> Changes

Workspace

Build Now

Configure

Delete Project

Rename

shell_commands

Permalinks

- Last build (#4), 1 min 23 sec ago
- Last stable build (#4), 1 min 23 sec ago
- Last successful build (#4), 1 min 23 sec ago
- Last failed build (#2), 22 min ago
- Last unsuccessful build (#2), 22 min ago
- Last completed build (#4), 1 min 23 sec ago

Builds

Filter

Today

- #4 8:36AM
- #3 8:21AM
- #2 8:15AM

Instances | EC2 | ap-south-1 × **shell_commands #4 Console - Jenkins** × **PranaliKapare/hello-world-java: x** +

Not Secure http://13.232.92.108:8080/job/shell_commands/4/console

Jenkins / shell_commands / #4 / Console Output

Edit Build Information

Delete build '#4'

Timings

Git Build Data

Previous Build

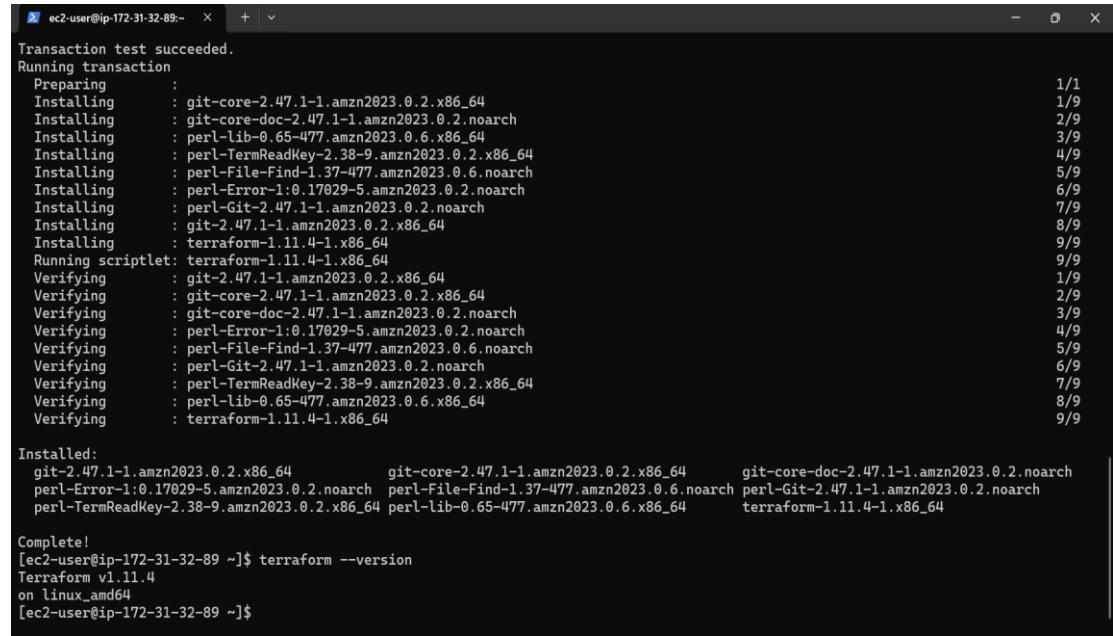
```

Building in workspace /var/lib/jenkins/workspace/shell_commands
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/shell_commands/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/PranaliKapare/hello-world-java # timeout=10
Fetching upstream changes from https://github.com/PranaliKapare/hello-world-java
> git --version # timeout=10
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/PranaliKapare/hello-world-java +refs/heads/*
:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision ad9a66c903ca043f9b982095c05f917ad6e3f889 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f ad9a66c903ca043f9b982095c05f917ad6e3f889 # timeout=10
Commit message: "Update HelloWorld.java"
First time build. Skipping changelog.
[shell_commands] $ /bin/sh -xe /tmp/jenkins9953300781550558217.sh
+ javac HelloWorld.java
+ java HelloWorld
Hello from my forked repo!
Finished: SUCCESS

```

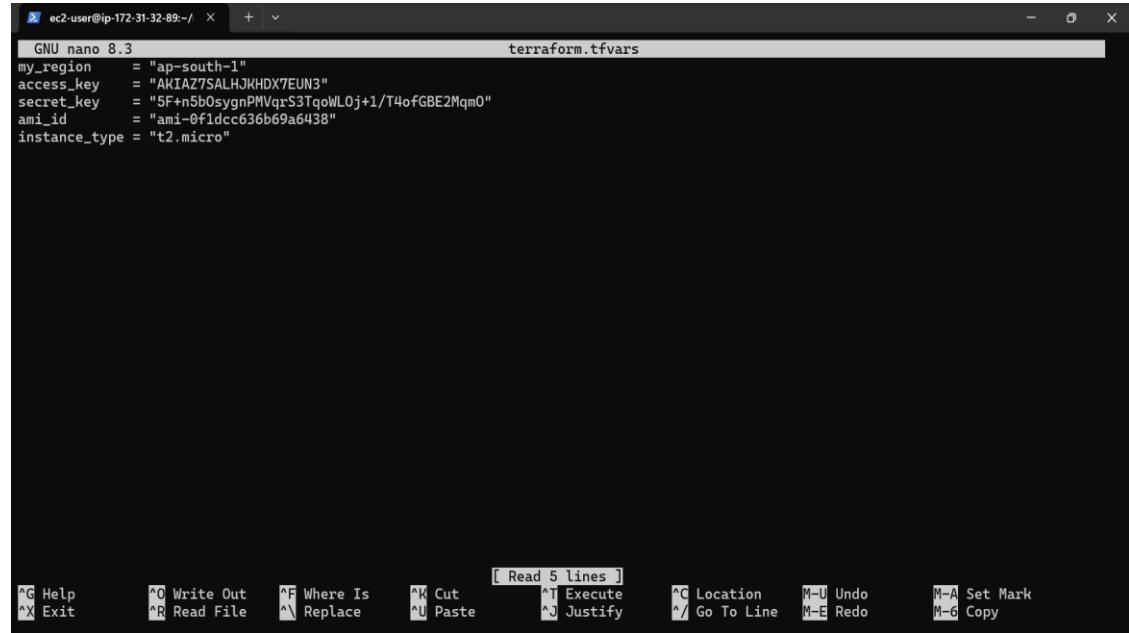
4. Terraform Installation and Setup

- Install Terraform on your local machine or a designated server.



```
ec2-user@ip-172-31-32-89: ~ + | v
Transaction test succeeded.
Running transaction
Preparing : 1/1
Installing : git-core-2.47.1-1.amzn2023.0.2.x86_64 1/9
Installing : git-core-doc-2.47.1-1.amzn2023.0.2.noarch 2/9
Installing : perl-lib-0.65-477.amzn2023.0.6.x86_64 3/9
Installing : perl-TermReadKey-2.38-9.amzn2023.0.2.x86_64 4/9
Installing : perl-File-Find-1.37-477.amzn2023.0.6.noarch 5/9
Installing : perl-Error-1:0.17029-5.amzn2023.0.2.noarch 6/9
Installing : perl-Git-2.47.1-1.amzn2023.0.2.noarch 7/9
Installing : git-2.47.1-1.amzn2023.0.2.x86_64 8/9
Installing : terraform-1.11.4-1.x86_64 9/9
Running scriptlet: terraform=1.11.4-1.x86_64 9/9
Verifying : git-2.47.1-1.amzn2023.0.2.x86_64 1/9
Verifying : git-core-2.47.1-1.amzn2023.0.2.x86_64 2/9
Verifying : git-core-doc-2.47.1-1.amzn2023.0.2.noarch 3/9
Verifying : perl-Error-1:0.17029-5.amzn2023.0.2.noarch 4/9
Verifying : perl-File-Find-1.37-477.amzn2023.0.6.noarch 5/9
Verifying : perl-Git-2.47.1-1.amzn2023.0.2.noarch 6/9
Verifying : perl-TermReadKey-2.38-9.amzn2023.0.2.x86_64 7/9
Verifying : perl-lib-0.65-477.amzn2023.0.6.x86_64 8/9
Verifying : terraform-1.11.4-1.x86_64 9/9
Installed:
git-2.47.1-1.amzn2023.0.2.x86_64          git-core-2.47.1-1.amzn2023.0.2.x86_64      git-core-doc-2.47.1-1.amzn2023.0.2.noarch
perl-Error-1:0.17029-5.amzn2023.0.2.noarch perl-File-Find-1.37-477.amzn2023.0.6.noarch perl-Git-2.47.1-1.amzn2023.0.2.noarch
perl-TermReadKey-2.38-9.amzn2023.0.2.x86_64 perl-lib-0.65-477.amzn2023.0.6.x86_64      terraform-1.11.4-1.x86_64
Complete!
[ec2-user@ip-172-31-32-89 ~]$ terraform --version
Terraform v1.11.4
on linux_amd64
[ec2-user@ip-172-31-32-89 ~]$
```

- Configure your cloud provider's credentials for Terraform.



```
GNU nano 8.3                                     terraform.tfvars
my_region      = "ap-south-1"
access_key     = "AKIAZ7SALHJKHDX7EUN3"
secret_key     = "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"
ami_id         = "ami-0fdcc636b69a6438"
instance_type  = "t2.micro"

[ Read 5 lines ]
^G Help      ^O Write Out   ^F Where Is    ^K Cut        ^T Execute   ^C Location   M-U Undo
^X Exit      ^R Read File   ^\ Replace     ^U Paste      ^J Justify   ^I Go To Line M-E Redo
                                         M-A Set Mark M-6 Copy
```

Define Infrastructure Resources

- Define Terraform configuration files (.tf) to create infrastructure resources using Terraform's HashiCorp Configuration Language (HCL).

- Create resources such as virtual machines, storage, networking components, and security groups. Use a cloud AWS.
- Use variables and data sources to parameterize your configuration.

The image shows two terminal windows side-by-side, both displaying code in a nano editor.

Top Terminal (variables.tf):

```
GNU nano 8.3
variable "my_region" {
  type = string
}

variable "access_key" {
  type = string
}

variable "secret_key" {
  type = string
}

variable "ami_id" {
  type = string
}

variable "instance_type" {
  type = string
}
```

Bottom Terminal (main.tf):

```
GNU nano 8.3
provider "aws" {
  region = var.my_region
  access_key = var.access_key
  secret_key = var.secret_key
}

resource "aws_vpc" "myVPC" {
  cidr_block = "10.0.0.0/16"
}

resource "aws_subnet" "mySubnet" {
  vpc_id      = aws_vpc.myVPC.id
  cidr_block  = "10.0.1.0/24"
  availability_zone = "ap-south.1b"
}

resource "aws_security_group" "mySG" {
  name    = "terraformSG"
  vpc_id = aws_vpc.myVPC.id

  ingress {
    from_port  = 22
    to_port    = 22
    protocol   = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

resource "aws_instance" "myEC2" {
```

Provision and Apply Infrastructure:

- Use the terraform init, terraform plan, and terraform apply commands to initialize and provision your infrastructure.

```
[ec2-user@ip-172-31-32-89: ~]$ ls
myTerraform
[ec2-user@ip-172-31-32-89 ~]$ cd myTerraform/
[ec2-user@ip-172-31-32-89 myTerraform]$ ls
main.tf  terraform.tfvars  variables.tf
[ec2-user@ip-172-31-32-89 myTerraform]$ nano variables.tf
[ec2-user@ip-172-31-32-89 myTerraform]$ nano main.tf
[ec2-user@ip-172-31-32-89 myTerraform]$ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.95.0...
- Installed hashicorp/aws v5.95.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
[ec2-user@ip-172-31-32-89 myTerraform]$
```

```
[ec2-user@ip-172-31-32-89: ~]$ terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following
symbols:
+ create

Terraform will perform the following actions:

# aws_instance.myEC2 will be created
+ resource "aws_instance" "myEC2" {
    + ami                                = "ami-0f1dccc636b69a6438"
    + arn                                = (known after apply)
    + associate_public_ip_address        = (known after apply)
    + availability_zone                  = (known after apply)
    + cpu_core_count                     = (known after apply)
    + cpu_threads_per_core              = (known after apply)
    + disable_api_stop                  = (known after apply)
    + disable_api_termination           = (known after apply)
    + ebs_optimized                      = (known after apply)
    + enable_primary_ipv6               = (known after apply)
    + get_password_data                 = false
    + host_id                            = (known after apply)
    + host_resource_group_arn            = (known after apply)
    + iam_instance_profile              = (known after apply)
    + id                                 = (known after apply)
    + instance_initiated_shutdown_behavior = (known after apply)
    + instance_lifecycle                = (known after apply)
    + instance_state                    = (known after apply)
    + instance_type                     = "t2.micro"
    + ipv6_address_count                = (known after apply)
    + ipv6_addresses                   = (known after apply)
    + key_name                           = (known after apply)
    + monitoring                         = (known after apply)
```

```
[ec2-user@ip-172-31-32-89:~]$ terraform apply --auto-approve

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

# aws_instance.myEC2 will be created
+ resource "aws_instance" "myEC2" {
    ami                               = "ami-0f1dcc636b69a6438"
    arn                             = (known after apply)
    associate_public_ip_address      = (known after apply)
    availability_zone                = (known after apply)
    cpu_core_count                   = (known after apply)
    cpu_threads_per_core            = (known after apply)
    disable_api_stop                 = (known after apply)
    disable_api_termination          = (known after apply)
    ebs_optimized                    = (known after apply)
    enable_primary_ipv6              = (known after apply)
    get_password_data                = false
    host_id                          = (known after apply)
    host_resource_group_arn          = (known after apply)
    iam_instance_profile             = (known after apply)
    id                                = (known after apply)
    instance_initiated_shutdown_behavior = (known after apply)
    instance.lifecycle               = (known after apply)
    instance.state                   = (known after apply)
    instance_type                    = "t2.micro"
    ipv6_address_count               = (known after apply)
    ipv6_addresses                   = (known after apply)
    key_name                         = (known after apply)
    monitoring                       = (known after apply)
}
```

- Make sure that Terraform successfully provisions the defined resources and saves their state.

```
[ec2-user@ip-172-31-32-89:~]$ terraform apply --auto-approve

+ cidr_block                           = "10.0.0.0/16"
+ default_network_acl_id               = (known after apply)
+ default_route_table_id               = (known after apply)
+ default_security_group_id            = (known after apply)
+ dhcp_options_id                     = (known after apply)
+ enable_dns_hostnames                = (known after apply)
+ enable_dns_support                  = true
+ enable_network_address_usage_metrics = (known after apply)
+ id                                  = (known after apply)
+ instance_tenancy                    = "default"
+ ipv6_association_id                 = (known after apply)
+ ipv6_cidr_block                     = (known after apply)
+ ipv6_cidr_block_network_border_group = (known after apply)
+ main_route_table_id                 = (known after apply)
+ owner_id                            = (known after apply)
+ tags_all                            = (known after apply)
}

Plan: 4 to add, 0 to change, 0 to destroy.
aws_vpc.myVPC: Creating...
aws_vpc.myVPC: Creation complete after 1s [id=vpc-0feffc0d21654b61e]
aws_subnet.mySubnet: Creating...
aws_security_group.mySG: Creating...
aws_subnet.mySubnet: Creation complete after 1s [id=subnet-05057db984848617e]
aws_security_group.mySG: Creation complete after 2s [id=sg-0e8669f9b49c342f9]
aws_instance.myEC2: Creating...
aws_instance.myEC2: Still creating... [10s elapsed]
aws_instance.myEC2: Creation complete after 12s [id=i-001b2c66cc8a39879]

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.
[ec2-user@ip-172-31-32-89 myTerraform]$ ls
main.tf  terraform  terraform.tfstate  terraform.state.backup  terraform.tfvars  variables.tf
[ec2-user@ip-172-31-32-89 myTerraform]$
```

Scaling and Variable Expansion :

- Extend your Terraform configuration to allow for horizontal scaling of resources (e.g., autoscaling groups or instance counts).
- Implement variable expansion and dynamic resource creation based on the number of instances required.

The screenshot shows two terminal windows side-by-side.

The left terminal window displays the Terraform configuration file `main.tf` in the `GNU nano 8.3` editor. The code defines resources for a VPC, Security Group, and EC2 instances:

```

resource "aws_vpc" "myVPC" {
  vpc_id      = aws_vpc.myVPC.id
  cidr_block  = "10.0.1.0/24"
  availability_zone = "ap-south-1b"
}

resource "aws_security_group" "mySG" {
  name     = "terraformSG"
  vpc_id   = aws_vpc.myVPC.id

  ingress {
    from_port  = 22
    to_port    = 22
    protocol   = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

resource "aws_instance" "myEC2" {
  count          = var.instance_count
  ami            = var.ami_id
  instance_type = var.instance_type
  subnet_id     = aws_subnet.mySubnet.id
  vpc_security_group_ids = [aws_security_group.mySG.id]

  tags = {
    Name = "WebInstance-${count.index + 1}"
  }
}

```

The right terminal window shows the output of the `terraform apply` command. It lists the resources being created and their status:

```

+ id                               = (known after apply)
+ instance_tenancy                 = "default"
+ ipv6_association_id              = (known after apply)
+ ipv6_cidr_block                  = (known after apply)
+ ipv6_cidr_block_network_border_group = (known after apply)
+ main_route_table_id              = (known after apply)
+ owner_id                         = (known after apply)
+ tags_all                         = (known after apply)

Plan: 6 to add, 0 to change, 0 to destroy.

aws_vpc.myVPC: Creating...
aws_vpc.myVPC: Creation complete after 1s [id=vpc-0ef7acb50aefeca1]
aws_subnet.mySubnet: Creating...
aws_security_group.mySG: Creating...
aws_subnet.mySubnet: Creation complete after 0s [id=subnet-08c4a111b9f1d6d2]
aws_security_group.mySG: Creation complete after 1s [id=sg-05f7c3fa479539f0e]
aws_instance.myEC2[1]: Creating...
aws_instance.myEC2[0]: Creating...
aws_instance.myEC2[2]: Creating...
aws_instance.myEC2[1]: Still creating... [10s elapsed]
aws_instance.myEC2[0]: Still creating... [10s elapsed]
aws_instance.myEC2[2]: Still creating... [10s elapsed]
aws_instance.myEC2[1]: Still creating... [20s elapsed]
aws_instance.myEC2[0]: Still creating... [20s elapsed]
aws_instance.myEC2[2]: Still creating... [20s elapsed]
aws_instance.myEC2[0]: Creation complete after 22s [id=i-08bfea9a947ef723d]
aws_instance.myEC2[2]: Creation complete after 22s [id=i-0c75e601009200554]
aws_instance.myEC2[1]: Still creating... [30s elapsed]
aws_instance.myEC2[1]: Creation complete after 32s [id=i-09c2e8327c586f0de]

Apply complete! Resources: 6 added, 0 changed, 0 destroyed.
[ec2-user@ip-172-31-32-89 myTerraform]$ ls

```

Remote State Management :

- Set up remote state management using a remote backend, such as an S3 bucket for AWS or a similar solution for your cloud provider.
- Ensure that Terraform state files are stored remotely and securely.

```

[ec2-user@ip-172-31-32-89: ~] ls
backend.tf  main.tf  terraform.tfstate  terraform.tfstate.backup  terraform.tfvars  variables.tf
[ec2-user@ip-172-31-32-89: ~] terraform init
Initializing the backend...
Do you want to copy existing state to the new backend?
Pre-existing state was found while migrating the previous "local" backend to the
newly configured "s3" backend. No existing state was found in the newly
configured "s3" backend. Do you want to copy this state to the new "s3"
backend? Enter "yes" to copy and "no" to start with an empty state.

Enter a value: yes

Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.95.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
[ec2-user@ip-172-31-32-89: ~] terraform
[ec2-user@ip-172-31-32-89: ~]

```

The screenshot shows the AWS S3 buckets page. A success message at the top states: "Successfully created bucket 'my-terraform-state-bucket-19'. To upload files and folders, or to configure additional bucket settings, choose View details." Below this, there are two tabs: "General purpose buckets" (selected) and "Directory buckets". A table lists three buckets:

Name	AWS Region	IAM Access Analyzer	Creation date
lecturesbuck	Asia Pacific (Mumbai) ap-south-1	View analyzer for ap-south-1	March 13, 2025, 21:28:16 (UTC+05:30)
my-terraform-state-bucket-19	Asia Pacific (Mumbai) ap-south-1	View analyzer for ap-south-1	April 24, 2025, 16:27:41 (UTC+05:30)
myterraformbuc	Asia Pacific (Mumbai) ap-south-1	View analyzer for ap-south-1	February 7, 2025, 12:34:12 (UTC+05:30)

At the bottom of the page, there are links for CloudShell, Feedback, and various legal notices.

The screenshot shows the AWS S3 console interface. At the top, there are tabs for 'Instances | EC2 | ap-south-1' and 'TerraformRemoteStateRole | IAI'. Below the tabs, the URL is 'ap-south-1.console.aws.amazon.com/s3/buckets/my-terraform-state-bucket-19?prefix=env%2Fdev%2F®ion=ap-south-1'. The navigation bar includes 'Search [Alt+S]', 'EC2', 'Lambda', 'Amazon S3 > Buckets > my-terraform-state-bucket-19 > env/ > dev/'. The main content area shows a table with one object:

Name	Type	Last modified	Size	Storage class
terraform.tfstate	tfstate	April 24, 2025, 16:36:05 (UTC+05:30)	18.3 KB	Standard

Infrastructure Updates and Changes :

- Modify your Terraform configuration to simulate changes to the infrastructure, such as adding or modifying resources.
- Apply these changes using the terraform apply command and verify that Terraform updates the infrastructure accordingly.

```

ec2-user@ip-172-31-32-89:~ % terraform plan
  ~ tags_all
    ~ "Name" = "WebInstance-2" -> "updatedInstance-2"
  }
  # (37 unchanged attributes hidden)

  }
  # (8 unchanged blocks hidden)

# aws_instance.myEC2[2] will be updated in-place
~ resource "aws_instance" "myEC2" {
  id                         = "i-01561e355d213c126"
  ~ instance_type              = "t3.micro" -> "t2.medium"
  ~ tags
    ~ "Name" = "WebInstance-3" -> "updatedInstance-3"
  }
  ~ tags_all
    ~ "Name" = "WebInstance-3" -> "updatedInstance-3"
  }
  # (37 unchanged attributes hidden)

  }
  # (8 unchanged blocks hidden)

}

Plan: 0 to add, 3 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
[ec2-user@ip-172-31-32-89 myTerraform]$
```

```
ec2-user@ip-172-31-32-89: ~ % terraform destroy
}
}

Plan: 0 to add, 0 to change, 6 to destroy.

aws_instance.myEC2[1]: Destroying... [id=i-0d21534b2038fec13]
aws_instance.myEC2[2]: Destroying... [id=i-01561e355d213c126]
aws_instance.myEC2[0]: Destroying... [id=i-064b5e0eb98546c3b]
aws_instance.myEC2[1]: Still destroying... [id=i-0d21534b2038fec13, 10s elapsed]
aws_instance.myEC2[2]: Still destroying... [id=i-01561e355d213c126, 10s elapsed]
aws_instance.myEC2[0]: Still destroying... [id=i-064b5e0eb98546c3b, 10s elapsed]
aws_instance.myEC2[1]: Still destroying... [id=i-0d21534b2038fec13, 20s elapsed]
aws_instance.myEC2[2]: Still destroying... [id=i-01561e355d213c126, 20s elapsed]
aws_instance.myEC2[0]: Still destroying... [id=i-064b5e0eb98546c3b, 20s elapsed]
aws_instance.myEC2[1]: Destruction complete after 30s
aws_instance.myEC2[2]: Still destroying... [id=i-01561e355d213c126, 30s elapsed]
aws_instance.myEC2[0]: Still destroying... [id=i-064b5e0eb98546c3b, 30s elapsed]
aws_instance.myEC2[1]: Destruction complete after 40s
aws_instance.myEC2[2]: Still destroying... [id=i-01561e355d213c126, 40s elapsed]
aws_instance.myEC2[0]: Still destroying... [id=i-01561e355d213c126, 50s elapsed]
aws_instance.myEC2[2]: Still destroying... [id=i-01561e355d213c126, 1m0s elapsed]
aws_instance.myEC2[1]: Still destroying... [id=i-01561e355d213c126, 1m10s elapsed]
aws_instance.myEC2[2]: Still destroying... [id=i-01561e355d213c126, 1m20s elapsed]
aws_instance.myEC2[0]: Still destroying... [id=i-01561e355d213c126, 1m30s elapsed]
aws_instance.myEC2[2]: Still destroying... [id=i-01561e355d213c126, 1m40s elapsed]
aws_instance.myEC2[1]: Destruction complete after 1m41s
aws_security_group.mySG: Destroying... [id=sg-05f7c3fa479539f0e]
aws_subnet.mySubnet: Destroying... [id=subnet-08c4a111b9fld6d2]
aws_subnet.mySubnet: Destruction complete after 0s
aws_security_group.mySG: Destruction complete after 0s
aws_vpc.myVPC: Destroying... [id=vpc-0ef7acb50afeacal]
aws_vpc.myVPC: Destruction complete after 1s

Destroy complete! Resources: 6 destroyed.
[ec2-user@ip-172-31-32-89 myTerraform]$
```