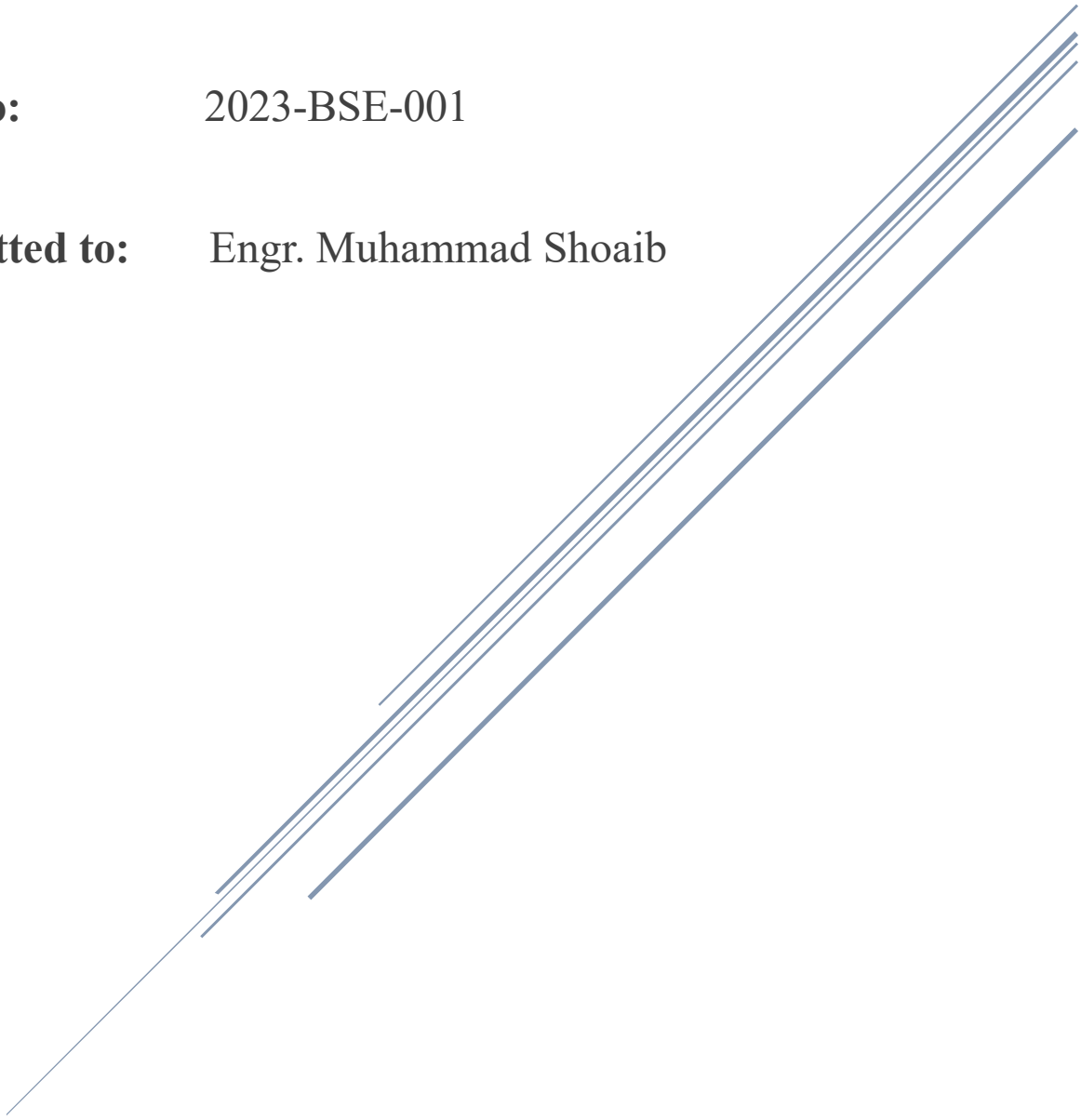
CLOUD COMPUTING

Lab 10

Name: Abiha Nadeem

Roll no: 2023-BSE-001

Submitted to: Engr. Muhammad Shoaib

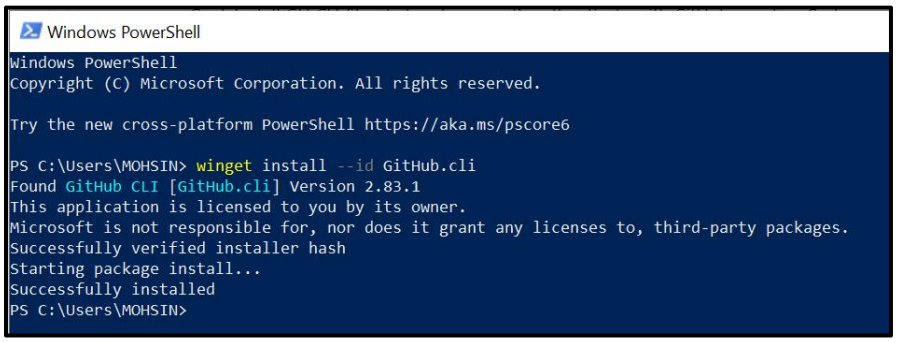


Task 1 — GitHub CLI Codespace Setup & Authentication

Goal: Install GH CLI (if not present), authenticate for Codespace use, and connect into the Codespace shell **(all steps inside Codespace unless noted)**.

1. Install GitHub CLI:

- winget install --id GitHub.cli

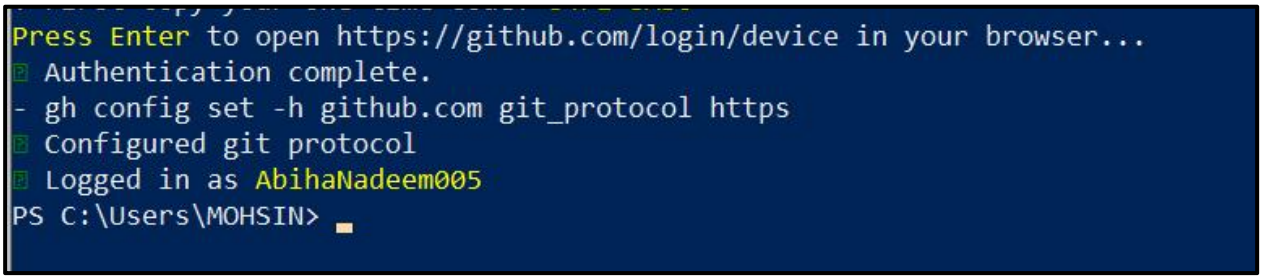


Save screenshot as: task1_gh_install.png — terminal showing the winget install command output.

2. Authenticate GH CLI for Codespaces:

- gh auth login -s codespace
- When prompted, generate a **GitHub access token (classic)**. Set scopes to: admin:org, codespace, repo.

Save screenshot as: task1_gh_auth_login.png — screenshot of gh auth login confirmation (redact token).



3. List available codespaces:

- gh codespace list

Save screenshot as: task1_codespace_list.png — output showing the codespace name.

```
PS C:\Users\MOHSIN> gh codespace list
no codespaces found
PS C:\Users\MOHSIN>
```

4. Connect to a codespace via SSH:

- gh codespace ssh -c <name_of_codespace>

Save screenshot as: task1_codespace_ssh_connected.png — terminal inside the Codespace shell after ssh.

```
PS C:\Users\MOHSIN> gh codespace create --repo AbihaNadeem005/Lab10Repo --branch main --machine basicLinux32gb
  ⓘ Codespaces usage for this repository is paid for by AbihaNadeem005
redesigned-space-cod-g4vqpr59w9r4hv95p
PS C:\Users\MOHSIN> gh codespace ssh -c redesigned-space-cod-g4vqpr59w9r4hv95p
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.8.0-1030-azure x86_64)

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

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

@AbihaNadeem005 ⓘ /workspaces/Lab10Repo (main) $
```

Task 2 — Install AWS CLI, Terraform CLI, Provider Setup

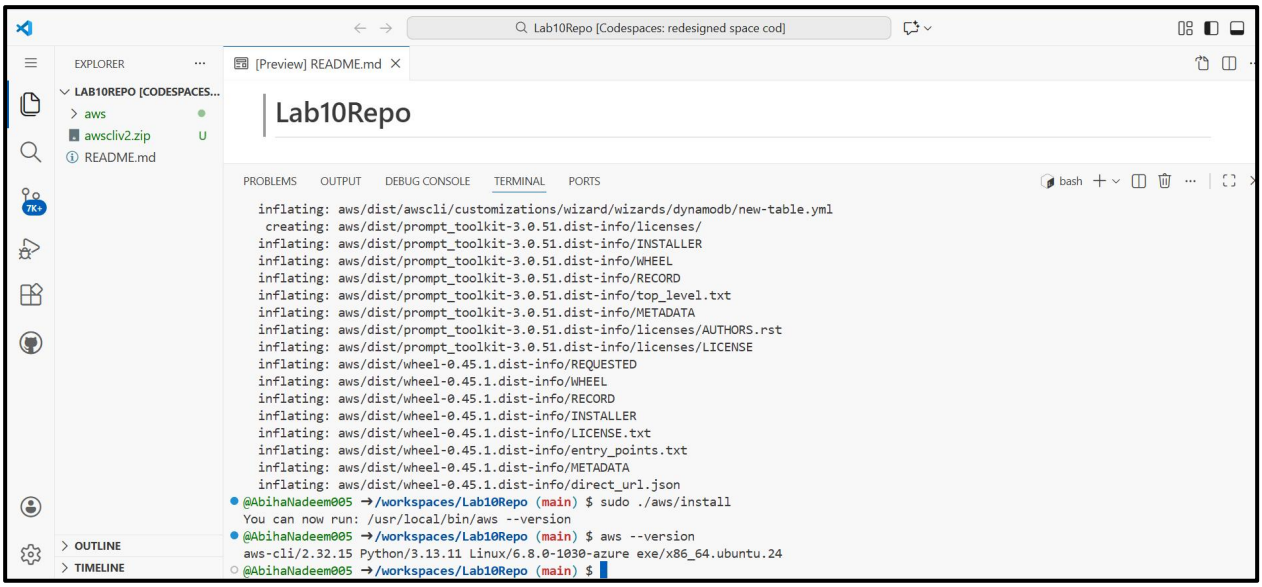
Goal: Install AWS CLI (if not present) & Terraform CLI inside Codespace; configure initial Terraform provider.

A. Install AWS CLI (Skip if already installed)

1. In Codespace shell, install AWS CLI:

- curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
- unzip awscliv2.zip
- sudo ./aws/install
- aws --version

Save screenshot as: task2_aws_install_and_version.png — show AWS CLI installation and version output.



2. Configure AWS CLI:

- aws configure

Save screenshot as: task2_aws_configure_and_files.png — show aws configure prompt (redact secret values).

- Then check config files:

- cat ~/.aws/credentials
- cat ~/.aws/config

Save screenshot as: task2_aws_configure_and_files.png — output of cat commands (redact secret keys if visible).

3. Verify AWS CLI connectivity:

- aws sts get-caller-identity

Save screenshot as: task2_aws_get_caller_identity.png — output of aws sts get-caller-identity.

B. Install Terraform CLI

1. Install Terraform CLI:

- wget -O - https://apt.releases.hashicorp.com/gpg | sudo gpg --dearmor -o /usr/share/keyrings/hashicorp-archive-keyring.gpg
- echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(grep -oP '(?<=UBUNTU_CODENAME=).*' /etc/os-release || lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
- sudo apt update
- sudo apt install terraform
- which terraform
- terraform --version

Save screenshot as: task2_terraform_install_and_version.png — terraform installation and version check output.

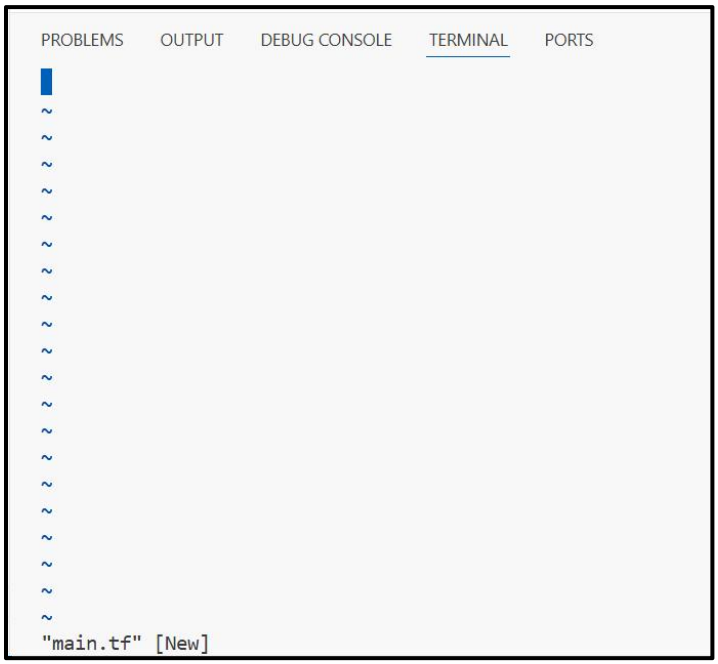
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $ sudo apt install terraform
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  terraform
0 upgraded, 1 newly installed, 0 to remove and 30 not upgraded.
Need to get 30.6 MB of archives.
After this operation, 101 MB of additional disk space will be used.
Get:1 https://apt.releases.hashicorp.com noble/main amd64 terraform amd64 1.14.2-1 [30.6 MB]
Fetched 30.6 MB in 0s (115 MB/s)
Selecting previously unselected package terraform.
(Reading database ... 58629 files and directories currently installed.)
Preparing to unpack .../terraform_1.14.2-1_amd64.deb ...
Unpacking terraform (1.14.2-1) ...
Setting up terraform (1.14.2-1) ...
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $ which terraform
/usr/bin/terraform
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $ terraform --version
Terraform v1.14.2
on linux_amd64
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $
```

C. Provider Configuration (main.tf)

1. Create main.tf using Vim:

- vim main.tf

Save screenshot as: task2_provider_file_creation.png — creation/editing of main.tf.

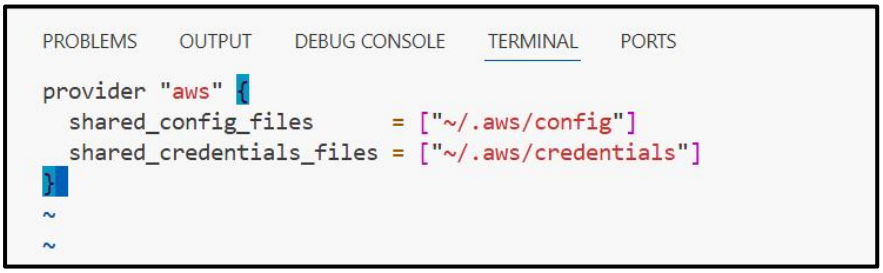


2. Inside main.tf, write:

- provider "aws" {
- shared_config_files = ["~/.aws/config"]
- shared_credentials_files = ["~/.aws/credentials"]

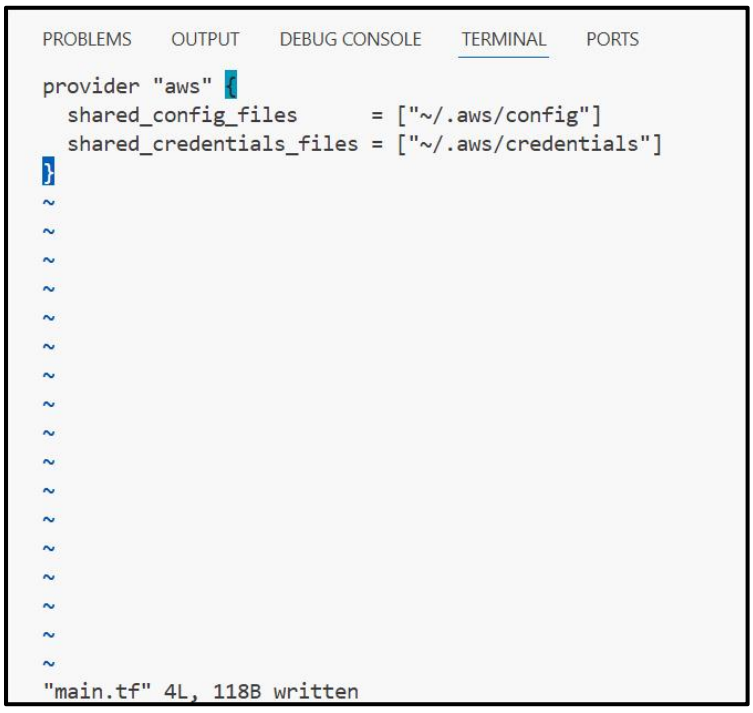
}

Save screenshot as: task2_provider_block.png — final content saved in main.tf.



3. Save and quit: Press ESC then :wq

Save screenshot as: task2_vim_save_main_tf.png — Vim save confirmation (optional).



4. Initialize Terraform:

- terraform init

Save screenshot as: task2_terraform_init_output.png — terraform init output.

```
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $ terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v6.26.0...
- Installed hashicorp/aws v6.26.0 (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.
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $
```

5. Show contents of .terraform.lock.hcl:

cat .terraform.lock.hcl

Save screenshot as: task2_terraform_lock_hcl.png — output of .terraform.lock.hcl.

```
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $ cat .terraform.lock.hcl
# This file is maintained automatically by "terraform init".
# Manual edits may be lost in future updates.

provider "registry.terraform.io/hashicorp/aws" {
  version = "6.26.0"
  hashes = [
    "h1:B7X8EU6aZ9KzIvP0VBDhhgadgjXIrUgMXt/pJ6EUXvo=",
    "zh:038fd943de79acd9f9f73106fa0eba588c6a0d4e0993e146f51f3aa043728c5f",
    "zh:06fa0177d33d3d3f9cb7e205fbeb1c4c3095ba637e2b4d292429401ec5612e81",
    "zh:212714fc8b6ee57e26d11d0fdf2ecfe23b37a6eac1008b399c1d790528c3f072",
    "zh:3197725d772f360e9e466b68a5ba67363e9f6786809c9adefc50f7f7e525bf42",
    "zh:33385539f3e3fafb96c6036421fd72b05c76505eeefaaff8a089c3eeba25db65",
    "zh:4ce065e0d3c384d11c1b59fe92582d331aae27ff6e019ace07b8cedef5653aae",
    "zh:67863d6ff5517db2c0b8097443708dca548f1922d2e08ad76a98d493ff480cb1",
    "zh:771ccf61fc107013b437b0a05cdb342823a99200653bfe9b892702b9fd8997fe",
    "zh:80adcf83bef9d683606c48bbe53cbb2b5a04878641674e957939b5e8f124ada0",
    "zh:9675c7f209db8e64ba2d5197acc8ba0073bd73b48c3dd61a1961a44844bc8a81",
    "zh:9b12af85486a96aedd8d7984b0ff811a4b42e3d88dad1a3fb4c0b580d04fa425",
    "zh:b47d0f5eff91c94c5d5677815b9361e64dfbe2ee2d59ba2867e2d0f5fa7181e4",
    "zh:b4933663b8d6cc1cfb51aa47bd8f26c06012ee2e278e57663faffdc722dd5baa",
    "zh:d53a94ecdb6b68a8dc19ec6e16ba2d4c2acde575af254d1b8b80143e57c76abf",
    "zh:e7cb8c1770c6f87c5ce1d3e28b838380bb8e5296dd03034b796168de8be1c7ec",
  ]
}
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $
```

6. Show contents of .terraform/ directory:

- ls .terraform/

Save screenshot as: task2_terraform_dir_ls.png — output of ls .terraform/.

```
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $ ls .terraform/
providers
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $
```

Task 3 — VPC/Subnet Creation, Initialization, Verification

Goal: Use Terraform to create VPC & Subnet resources, verify via CLI.

1. Edit main.tf to add:

- resource "aws_vpc" "development_vpc" {

cidr_block = "10.0.0.0/16"

}

- resource "aws_subnet" "dev_subnet_1" {

vpc_id = aws_vpc.development_vpc.id

cidr_block = "10.0.10.0/24"

availability_zone = "me-central-1a"}

Use vim main.tf, edit and insert code.

Save screenshot as: task3_main_tf_resource_add.png — new resources added and visible in main.tf.

2. Run:

- terraform apply

Type yes when prompted.

Save screenshot as: task3_terraform_apply_vpc_subnet.png — output showing successful apply and resources creation.

```
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $ terraform apply
      + 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)
      + region                               = "me-central-1"
      + tags_all                             = (known after apply)
    }

Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_vpc.development_vpc: Creating...
aws_vpc.development_vpc: Creation complete after 4s [id=vpc-0fca8f9661b00d1f4]
aws_subnet.dev_subnet_1: Creating...
aws_subnet.dev_subnet_1: Creation complete after 1s [id=subnet-0eaed215dd4d1d0e4]

Apply complete! Resources: 2 added, 0 changed, 0 destroyed.
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $
```

3. Verify resources using AWS CLI:
    - Run:

- aws ec2 describe-subnets --filter "Name=subnet-id,Values=<subnet-id>"

```
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $ aws ec2 describe-subnets --filter "Name=subnet-id,Values=subnet-0eaed215dd4d1d0e4"
{
    "Subnets": [
        {
            "AvailabilityZoneId": "mec1-az1",
            "MapCustomerOwnedIpOnLaunch": false,
            "OwnerId": "709867665899",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": [],
            "SubnetArn": "arn:aws:ec2:me-central-1:709867665899:subnet/subnet-0eaed215dd4d1d0e4",
            "EnableDns64": false,
            "Ipv6Native": false,
            "PrivateDnsNameOptionsOnLaunch": {
                "HostnameType": "ip-name",
                "EnableResourceNameDnsARecord": false,
                "EnableResourceNameDnsAAAARecord": false
            },
            "BlockPublicAccessStates": {
                "InternetGatewayBlockMode": "off"
            },
            "SubnetId": "subnet-0eaed215dd4d1d0e4",
            "State": "available",
            "VpcId": "vpc-0fca8f9661b00d1f4",
            "CidrBlock": "10.0.10.0/24",
            "AvailableIpAddressCount": 251,
            "AvailabilityZone": "me-central-1a",
            "DefaultForAz": false,
            "MapPublicIpOnLaunch": false
```

Save screenshot as: task3_aws_cli_verify_subnet.png — output of describe-subnets command.

- Run:

- aws ec2 describe-vpcs --filter "Name=vpc-id,Values=<vpc-id>"

Save screenshot as: task3_aws_cli_verify_vpc.png — output of describe-vpcs command.

Task 4 — Data Source, Targeted Destroy, Tags

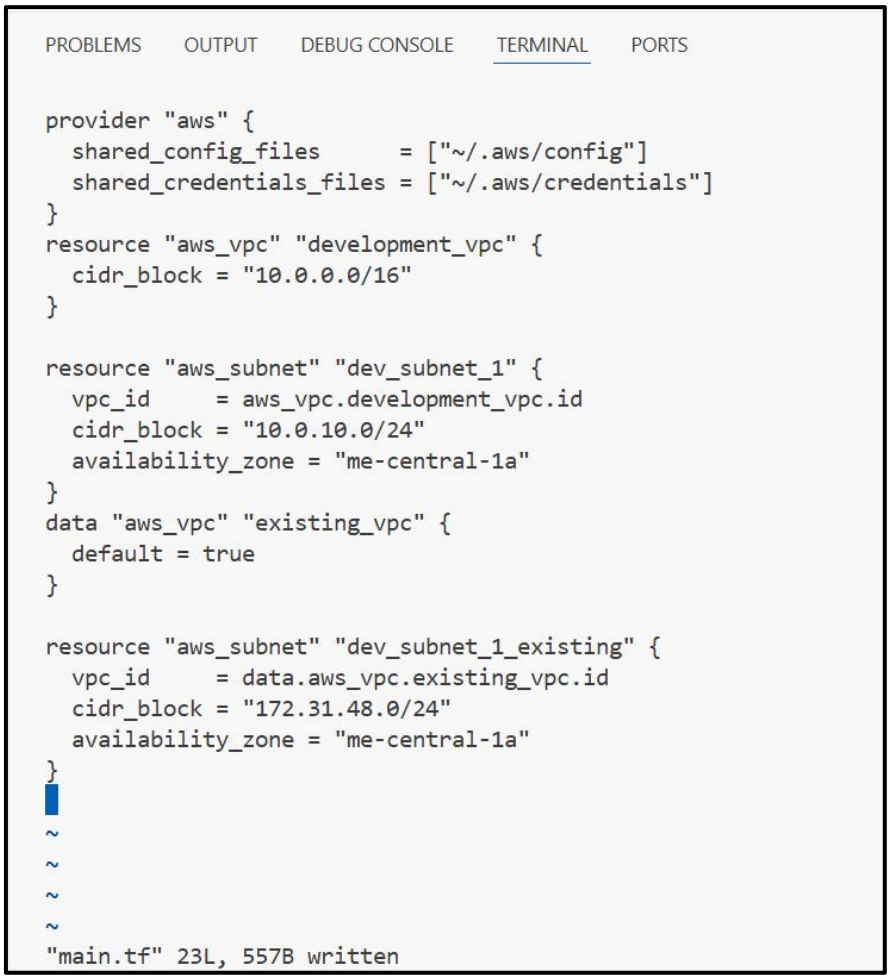
A. Data Source & Resource Creation

1. Add to main.tf:

```
data "aws_vpc" "existing_vpc" {
  default = true
}
resource "aws_subnet" "dev_subnet_1_existing" {
  vpc_id     = data.aws_vpc.existing_vpc.id
  cidr_block = "172.31.48.0/24"
  availability_zone = "me-central-1a"
}
```

Use vim to edit and insert code.

Save screenshot as: task4_main_tf_datasource_resource_add.png — main.tf with datasource and resource added.



2. Apply configuration:

- terraform apply

Confirm only the new subnet is created.

Save screenshot as: task4_terraform_apply_datasource_resource.png — terraform output of resource creation.

```
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $ terraform apply
      + enable_dns64                                 = false
      + enable_resource_name_dns_a_record_on_launch  = false
      + enable_resource_name_dns_aaaa_record_on_launch = false
      + id                                           = (known after apply)
      + ipv6_cidr_block_association_id               = (known after apply)
      + ipv6_native                                  = false
      + map_public_ip_on_launch                      = false
      + owner_id                                     = (known after apply)
      + private_dns_hostname_type_on_launch          = (known after apply)
      + region                                       = "me-central-1"
      + tags_all                                     = (known after apply)
      + vpc_id                                       = "vpc-00805ac6ee2802e51"
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_subnet.dev_subnet_1_existing: Creating...
aws_subnet.dev_subnet_1_existing: Creation complete after 2s [id=subnet-04a3d65ba4bcc5d46]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $
```

B. Targeted Destroy & Refresh

1. Destroy only one resource:

terraform destroy -target=aws_subnet.dev_subnet_1_existing

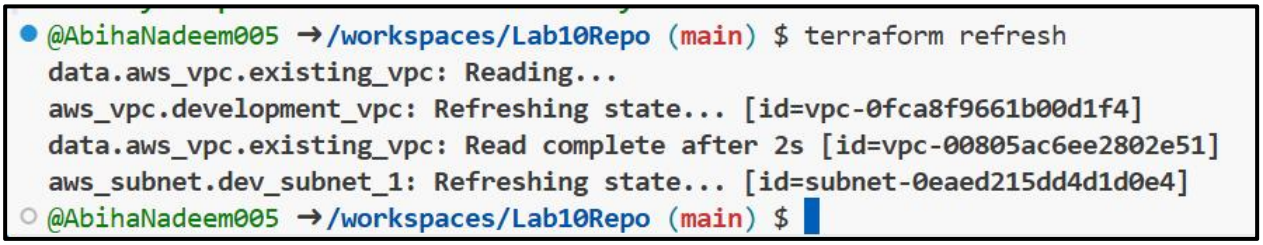
- **Save screenshot as:** task4_terraform_destroy_targeted.png — targeted destroy output.

```
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $ terraform destroy -target=aws_subnet.dev_subnet_1_existing
  by the current configuration.

  The -target option is not for routine use, and is provided only for exceptional situations such as recovering from errors or mistakes, or
  when Terraform specifically suggests to use it as part of an error message.


Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_subnet.dev_subnet_1_existing: Destroying... [id=subnet-04a3d65ba4bcc5d46]
aws_subnet.dev_subnet_1_existing: Destruction complete after 2s

  Warning: Applied changes may be incomplete

  The plan was created with the -target option in effect, so some changes requested in the configuration may have been ignored and the
  output values may not be fully updated. Run the following command to verify that no other changes are pending:
      terraform plan

  Note that the -target option is not suitable for routine use, and is provided only for exceptional situations such as recovering from
  errors or mistakes, or when Terraform specifically suggests to use it as part of an error message.


Destroy complete! Resources: 1 destroyed.
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $
```

2. Refresh state:

terraform refresh

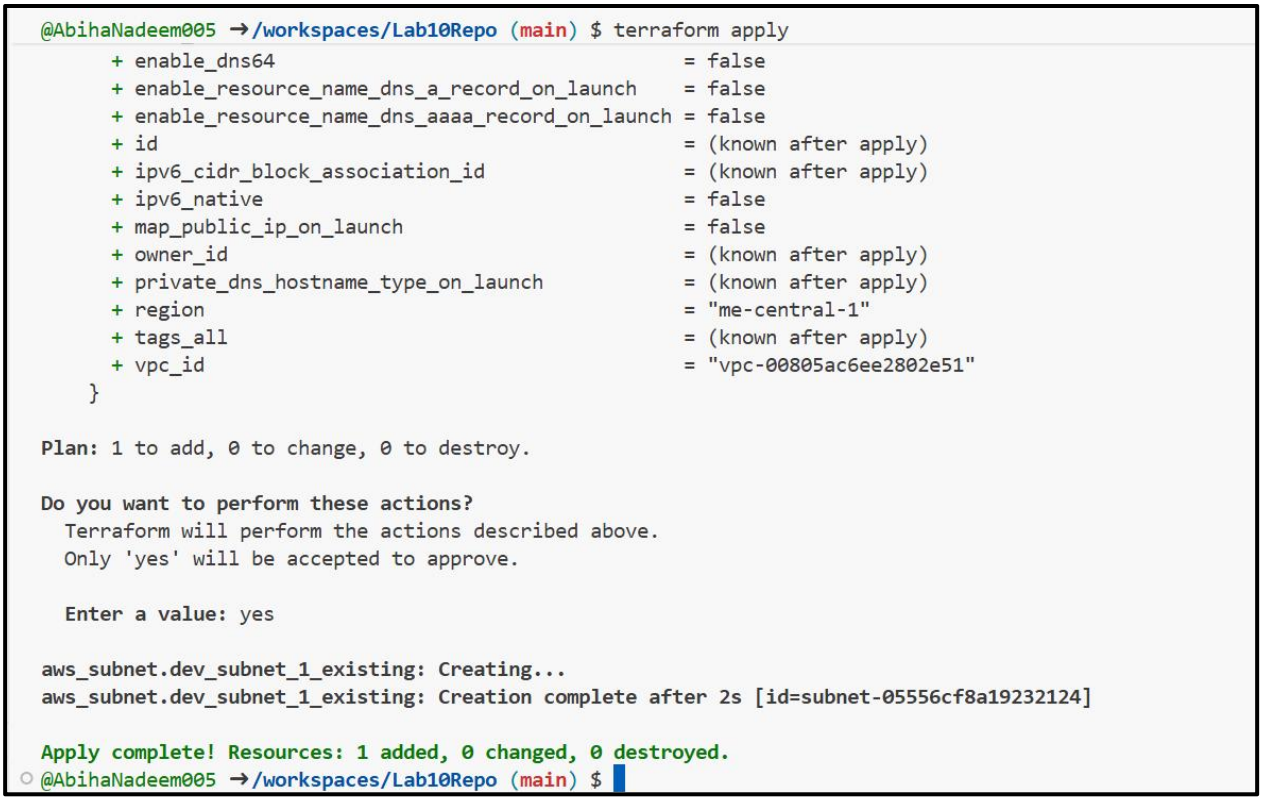
- **Save screenshot as:** task4_terraform_refresh_state.png — terraform refresh output.



3. Re-apply configuration:

terraform apply

- **Save screenshot as:** task4_terraform_apply_after_refresh.png — terraform apply after refresh.



4. Destroy all resources:

terraform destroy

- **Save screenshot as:** task4_terraform_destroy_all.png — full destroy output.

```
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $ terraform destroy
      - instance_tenancy                     = "default" -> null
      - ipv6_netmask_length                  = 0 -> null
      - main_route_table_id                  = "rtb-0249eaafa18abfb03" -> null
      - owner_id                             = "709867665899" -> null
      - region                               = "me-central-1" -> null
      - tags                                 = {} -> null
      - tags_all                             = {} -> null
        # (4 unchanged attributes hidden)
    }

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

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_subnet.dev_subnet_1: Destroying... [id=subnet-0eaed215dd4d1d0e4]
aws_subnet.dev_subnet_1_existing: Destroying... [id=subnet-05556cf8a19232124]
aws_subnet.dev_subnet_1_existing: Destruction complete after 2s
aws_subnet.dev_subnet_1: Destruction complete after 2s
aws_vpc.development_vpc: Destroying... [id=vpc-0fca8f9661b00d1f4]
aws_vpc.development_vpc: Destruction complete after 1s

Destroy complete! Resources: 3 destroyed.
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $
```

5. Plan configuration:

terraform plan

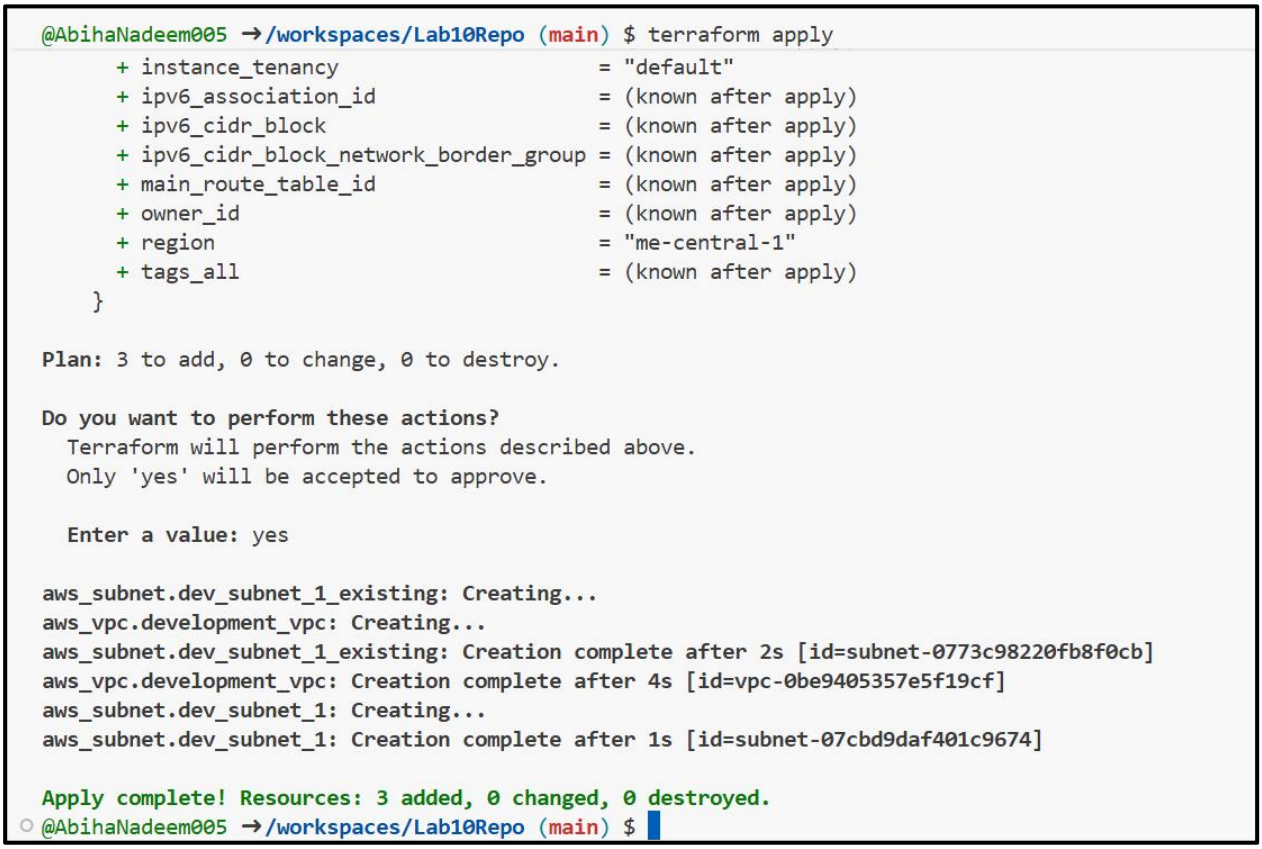
- **Save screenshot as:** task4_terraform_plan_output.png — terraform plan showing next changes.

```
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $ terraform plan
      + arn                                  = (known after apply)
      + 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)
      + region                               = "me-central-1"
      + tags_all                             = (known after apply)
    }

Plan: 3 to add, 0 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.
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $
```

6. Apply again:

terraform apply

- **Save screenshot as:** task4_terraform_apply_after_destroy.png — output showing resources recreated.



C. Tagging Resources

1. Modify main.tf to add tags:

```
resource "aws_vpc" "development_vpc" {
  cidr_block = "10.0.0.0/16"
  tags = {
    Name: "development"
    vpc_env = "dev"
  }
}
```

```
  resource "aws_subnet" "dev_subnet_1" {
    vpc_id     = aws_vpc.development_vpc.id
    cidr_block = "10.0.10.0/24"
    availability_zone = "me-central-1a"
    tags = {
      Name: "subnet-1-dev"
    }
  }

  resource "aws_subnet" "dev_subnet_1_existing" {
    vpc_id     = data.aws_vpc.existing_vpc.id
    cidr_block = "172.31.48.0/24"
    availability_zone = "me-central-1a"
    tags = {
      Name: "subnet-1-default"
    }
}
```

- Use vim to update tags.
- **Save screenshot as:** task4_main_tf_tagging.png — main.tf with tags added.

```
resource "aws_vpc" "development_vpc" {
  cidr_block = "10.0.0.0/16"
  tags = {
     Name: "development"
     vpc_env = "dev"
  }
}

resource "aws_subnet" "dev_subnet_1" {
  vpc_id     = aws_vpc.development_vpc.id
  cidr_block = "10.0.10.0/24"
  availability_zone = "me-central-1a"
  tags = {
     Name: "subnet-1-dev"
  }
}

resource "aws_subnet" "dev_subnet_1_existing" {
  vpc_id     = data.aws_vpc.existing_vpc.id
  cidr_block = "172.31.48.0/24"
  availability_zone = "me-central-1a"
  tags = {
     Name: "subnet-1-default"
  }
}
"main.tf" 47L, 1090B written
```

2. Run:

terraform refresh

terraform apply -auto-approve

- **Save screenshot as:** task4_terraform_apply_tagging.png — output showing tags applied.

```
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $ terraform apply -auto-approve
        # (20 unchanged attributes hidden)
    }

  # aws_vpc.development_vpc will be updated in-place
  ~ resource "aws_vpc" "development_vpc" {
        id                                   = "vpc-0be9405357e5f19cf"
      ~ tags                                 = {
          + "Name"    = "development"
          + "vpc_env" = "dev"
        }
      ~ tags_all                             = {
          + "Name"    = "development"
          + "vpc_env" = "dev"
        }
        # (19 unchanged attributes hidden)
    }

Plan: 0 to add, 3 to change, 0 to destroy.
aws_subnet.dev_subnet_1_existing: Modifying... [id=subnet-0773c98220fb8f0cb]
aws_vpc.development_vpc: Modifying... [id=vpc-0be9405357e5f19cf]
aws_subnet.dev_subnet_1_existing: Modifications complete after 2s [id=subnet-0773c98220fb8f0cb]
aws_vpc.development_vpc: Modifications complete after 3s [id=vpc-0be9405357e5f19cf]
aws_subnet.dev_subnet_1: Modifying... [id=subnet-07cbd9daf401c9674]
aws_subnet.dev_subnet_1: Modifications complete after 1s [id=subnet-07cbd9daf401c9674]

Apply complete! Resources: 0 added, 3 changed, 0 destroyed.
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $
```

3. Remove vpc_env = "dev" tag from development_vpc resource, re-plan/apply:
    - **Save screenshot as:** task4_terraform_plan_remove_tag.png — plan output showing tag removal.

```
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $ terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  ~ update in-place

Terraform will perform the following actions:

  # aws_vpc.development_vpc will be updated in-place
  ~ resource "aws_vpc" "development_vpc" {
        id                                   = "vpc-0be9405357e5f19cf"
      ~ tags                                 = {
            "Name"    = "development"
          - "vpc_env" = "dev" -> null
        }
      ~ tags_all                             = {
          - "vpc_env" = "dev" -> null
            # (1 unchanged element hidden)
        }
        # (19 unchanged attributes hidden)
    }

Plan: 0 to add, 1 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.
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $
```

    - **Save screenshot as:** task4_terraform_apply_remove_tag.png — apply output showing tag deleted.

```
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $ terraform apply
  # aws_vpc.development_vpc will be updated in-place
  ~ resource "aws_vpc" "development_vpc" {
        id                                   = "vpc-0be9405357e5f19cf"
      ~ tags                                 = {
            "Name"    = "development"
          - "vpc_env" = "dev" -> null
        }
      ~ tags_all                             = {
          - "vpc_env" = "dev" -> null
            # (1 unchanged element hidden)
        }
        # (19 unchanged attributes hidden)
    }

Plan: 0 to add, 1 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_vpc.development_vpc: Modifying... [id=vpc-0be9405357e5f19cf]
aws_vpc.development_vpc: Modifications complete after 3s [id=vpc-0be9405357e5f19cf]

Apply complete! Resources: 0 added, 1 changed, 0 destroyed.
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $
```

Task 5 — State File Inspection & Terraform State Commands

1. Destroy all resources:

terraform destroy

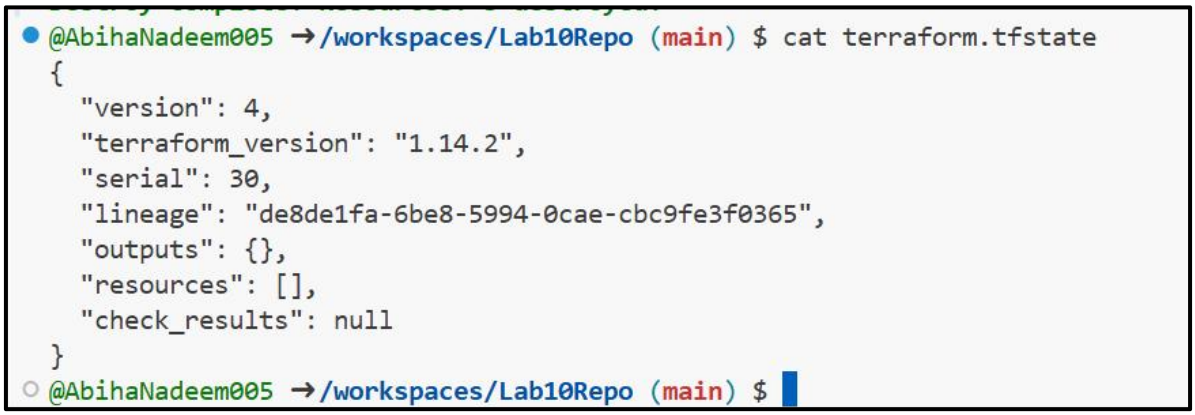
- **Save screenshot as:** task5_terraform_destroy.png — destroy output.

```
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $ terraform destroy
      - region                               = "me-central-1" -> null
      - tags                                 = {
          - "Name" = "development"
        } -> null
      - tags_all                             = {
          - "Name" = "development"
        } -> null
        # (4 unchanged attributes hidden)
    }

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

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_subnet.dev_subnet_1_existing: Destroying... [id=subnet-0773c98220fb8f0cb]
aws_subnet.dev_subnet_1: Destroying... [id=subnet-07cbd9daf401c9674]
aws_subnet.dev_subnet_1_existing: Destruction complete after 2s
aws_subnet.dev_subnet_1: Destruction complete after 2s
aws_vpc.development_vpc: Destroying... [id=vpc-0be9405357e5f19cf]
aws_vpc.development_vpc: Destruction complete after 1s

Destroy complete! Resources: 3 destroyed.
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $
```

2. Inspect state files after destroy:
    - Run:

cat terraform.tfstate

- **Save screenshot as:** task5_terraform_state_file_empty.png — output showing terraform.tfstate is empty after destroy.



- Run:

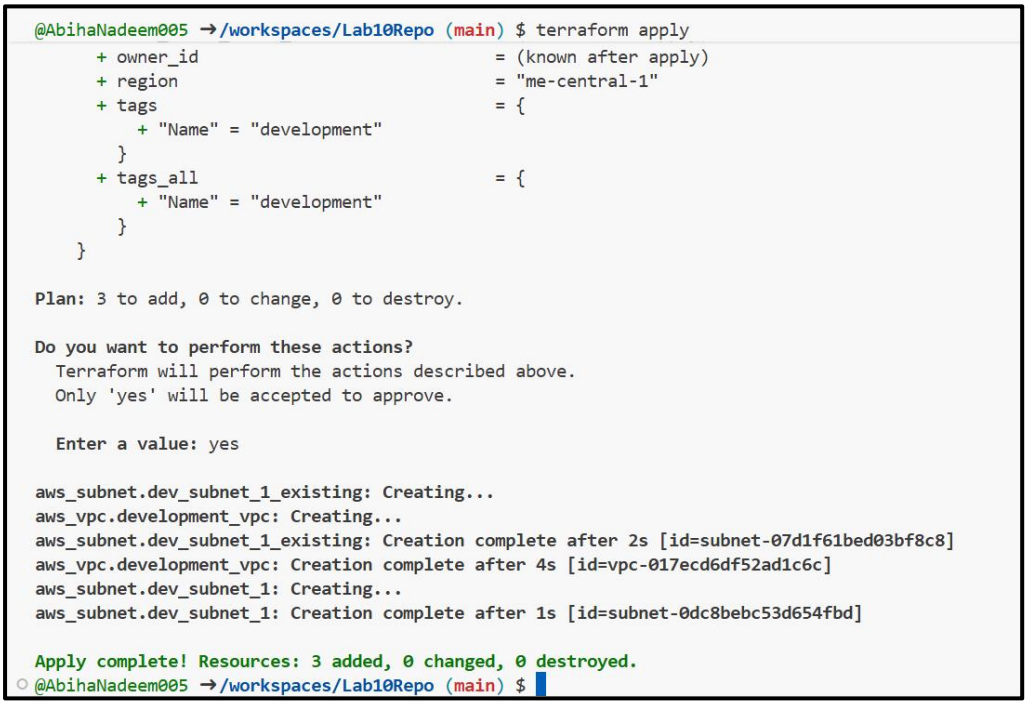
cat terraform.tfstate.backup

- **Save screenshot as:** task5_terraform_state_backup_prev.png — output showing backup file with previous resources.

3. Recreate resources:

terraform apply

- **Save screenshot as:** task5_terraform_apply_recreated.png — apply output, resources recreated.



4. View state files:

- Run:

cat terraform.tfstate

- **Save screenshot as:** task5_terraform_state_file_populated.png — output showing populated terraform.tfstate after restore.

```
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $ cat terraform.tfstate
            "ipv6_ipam_pool_id": "",
            "ipv6_netmask_length": 0,
            "main_route_table_id": "rtb-0aac206e52aa4a043",
            "owner_id": "709867665899",
            "region": "me-central-1",
            "tags": {
              "Name": "development"
            },
            "tags_all": {
              "Name": "development"
            }
          },
          "sensitive_attributes": [],
          "identity_schema_version": 0,
          "identity": {
            "account_id": "709867665899",
            "id": "vpc-017ecd6df52ad1c6c",
            "region": "me-central-1"
          },
          "private": "eyJzY2hlbWFfdmVyc2lvbiI6IjEifQ=="
        }
      ]
    }
  ],
  "check_results": null
}
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $
```

- Run:

cat terraform.tfstate.backup

- **Save screenshot as:** task5_terraform_state_backup_empty.png — output showing backup file empty after restore.

```
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $ cat terraform.tfstate.backup
{
  "version": 4,
  "terraform_version": "1.14.2",
  "serial": 30,
  "lineage": "de8de1fa-6be8-5994-0cae-cbc9fe3f0365",
  "outputs": {},
  "resources": [],
  "check_results": null
}
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $
```

5. List resources:

terraform state list

- **Save screenshot as:** task5_terraform_state_list.png — terraform state list output.

```
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $ terraform state list
data.aws_vpc.existing_vpc
aws_subnet.dev_subnet_1
aws_subnet.dev_subnet_1_existing
aws_vpc.development_vpc
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $
```

6. Show full attributes:

terraform state show <resource-name>

- **Save screenshot as:** task5_terraform_state_show_resource.png — state show output for one resource.

7. Note: Do **NOT** run the following (for information only):

   terraform state rm <resource-name>

   - Mentioned for knowledge — do not run.

Task 6 — Terraform Outputs & Attributes Reporting

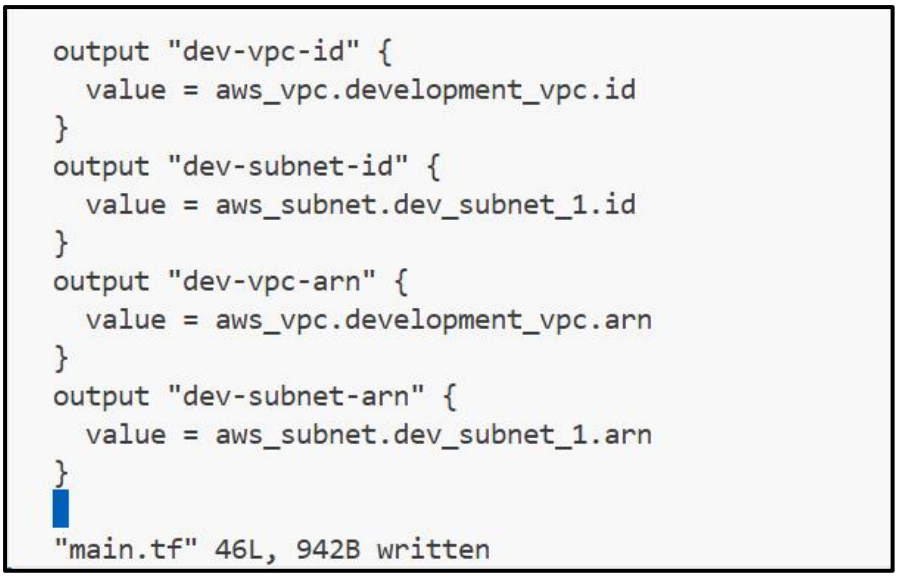
1. Add outputs in main.tf:

output "dev-vpc-id" {

  value = aws_vpc.development_vpc.id

```
}
output "dev-subnet-id" {
  value = aws_subnet.dev_subnet_1.id
}
output "dev-vpc-arn" {
  value = aws_vpc.development_vpc.arn
}
output "dev-subnet-arn" {
  value = aws_subnet.dev_subnet_1.arn
}
```

- **Save screenshot as:** task6_terraform_outputs_basic.png — output values for ids and arns after apply.



```
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $ terraform apply
aws_subnet.dev_subnet_1_existing: Refreshing state... [id=subnet-07d1f61bed03bf8c8]
aws_subnet.dev_subnet_1: Refreshing state... [id=subnet-0dc8bebc53d654fbd]

Changes to Outputs:
  + dev-subnet-arn = "arn:aws:ec2:me-central-1:709867665899:subnet/subnet-0dc8bebc53d654fbd"
  + dev-subnet-id  = "subnet-0dc8bebc53d654fbd"
  + dev-vpc-arn    = "arn:aws:ec2:me-central-1:709867665899:vpc/vpc-017ecd6df52ad1c6c"
  + dev-vpc-id     = "vpc-017ecd6df52ad1c6c"

You can apply this plan to save these new output values to the Terraform state, without changing any real infrastructure.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes


Apply complete! Resources: 0 added, 0 changed, 0 destroyed.

Outputs:

dev-subnet-arn = "arn:aws:ec2:me-central-1:709867665899:subnet/subnet-0dc8bebc53d654fbd"
dev-subnet-id = "subnet-0dc8bebc53d654fbd"
dev-vpc-arn = "arn:aws:ec2:me-central-1:709867665899:vpc/vpc-017ecd6df52ad1c6c"
dev-vpc-id = "vpc-017ecd6df52ad1c6c"
@AbihaNadeem005 ➜/workspaces/Lab10Repo (main) $
```

Return ALL of the following output attributes from your VPC and Subnet resources:

a) cidr_block

b) region

c) tags.Name

d) tags_all

Expand output section in main.tf accordingly:

```
output "dev-vpc-cidr_block" {
  value = aws_vpc.development_vpc.cidr_block
}
output "dev-vpc-region" {
  value = aws_vpc.development_vpc.region
}
output "dev-vpc-tags_name" {
  value = aws_vpc.development_vpc.tags["Name"]
}
output "dev-vpc-tags_all" {
```

```
  value = aws_vpc.development_vpc.tags_all
}
output "dev-subnet-cidr_block" {
  value = aws_subnet.dev_subnet_1.cidr_block
}
output "dev-subnet-region" {
  value = aws_subnet.dev_subnet_1.availability_zone
}
output "dev-subnet-tags_name" {
  value = aws_subnet.dev_subnet_1.tags["Name"]
}
output "dev-subnet-tags_all" {
  value = aws_subnet.dev_subnet_1.tags_all
}
```

Run terraform apply and record outputs.

- **Save screenshot as:** task6_expanded_outputs.png — output values for each required attribute after apply.

```
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $ terraform apply
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes


Apply complete! Resources: 0 added, 0 changed, 0 destroyed.

Outputs:

dev-subnet-arn = "arn:aws:ec2:me-central-1:709867665899:subnet/subnet-0dc8bebc53d654fbd"
dev-subnet-cidr_block = "10.0.10.0/24"
dev-subnet-id = "subnet-0dc8bebc53d654fbd"
dev-subnet-region = "me-central-1a"
dev-subnet-tags_all = tomap({
  "Name" = "subnet-1-dev"
})
dev-subnet-tags_name = "subnet-1-dev"
dev-vpc-arn = "arn:aws:ec2:me-central-1:709867665899:vpc/vpc-017ecd6df52ad1c6c"
dev-vpc-cidr_block = "10.0.0.0/16"
dev-vpc-id = "vpc-017ecd6df52ad1c6c"
dev-vpc-region = "me-central-1"
dev-vpc-tags_all = tomap({
  "Name" = "development"
})
dev-vpc-tags_name = "development"
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $
```

Cleanup — Delete Resources & State Verification

1. Destroy all resources:

terraform destroy

- **Save screenshot as:** cleanup_destroy_resources.png — destroy output.

```
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $ terraform destroy
      - Name = "subnet-1-dev"
    } -> null
  - dev-subnet-tags_name  = "subnet-1-dev" -> null
  - dev-vpc-arn           = "arn:aws:ec2:me-central-1:709867665899:vpc/vpc-017ecd6df52ad1c6c" -> null
  - dev-vpc-cidr_block    = "10.0.0.0/16" -> null
  - dev-vpc-id            = "vpc-017ecd6df52ad1c6c" -> null
  - dev-vpc-region        = "me-central-1" -> null
  - dev-vpc-tags_all      = {
      - Name = "development"
    } -> null
  - dev-vpc-tags_name     = "development" -> null

Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

aws_subnet.dev_subnet_1: Destroying... [id=subnet-0dc8bebc53d654fbd]
aws_subnet.dev_subnet_1_existing: Destroying... [id=subnet-07d1f61bed03bf8c8]
aws_subnet.dev_subnet_1: Destruction complete after 1s
aws_vpc.development_vpc: Destroying... [id=vpc-017ecd6df52ad1c6c]
aws_subnet.dev_subnet_1_existing: Destruction complete after 1s
aws_vpc.development_vpc: Destruction complete after 1s

Destroy complete! Resources: 3 destroyed.
@AbihaNadeem005 →/workspaces/Lab10Repo (main) $
```

2. Inspect state files:

cat terraform.tfstate

cat terraform.tfstate.backup

- **Save screenshot as:** cleanup_state_files.png — outputs showing file contents after clean-up.

```
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $ cat terraform.tfstate.backup
            "ipv6_ipam_pool_id": "",
            "ipv6_netmask_length": 0,
            "main_route_table_id": "rtb-0aac206e52aa4a043",
            "owner_id": "709867665899",
            "region": "me-central-1",
            "tags": {
              "Name": "development"
            },
            "tags_all": {
              "Name": "development"
            }
          },
          "sensitive_attributes": [],
          "identity_schema_version": 0,
          "identity": {
            "account_id": "709867665899",
            "id": "vpc-017ecd6df52ad1c6c",
            "region": "me-central-1"
          },
          "private": "eyJzY2hlbWFfdmVyc2lvbiI6IjEifQ=="
        }
      ]
    }
  ],
  "check_results": null
}
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $
```

3. Reapply then compare state and backup files to see differences.

```
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $ cat terraform.tfstate
            }
          },
          "sensitive_attributes": [],
          "identity_schema_version": 0,
          "identity": {
            "account_id": "709867665899",
            "id": "vpc-020f4c0aa07ef7d4f",
            "region": "me-central-1"
          },
          "private": "eyJzY2hlbWFfdmVyc2lvbiI6IjEifQ=="
        }
      ]
    }
  ],
  "check_results": null
}
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $ cat terraform.tfstate.backup
{
  "version": 4,
  "terraform_version": "1.14.2",
  "serial": 41,
  "lineage": "de8de1fa-6be8-5994-0cae-cbc9fe3f0365",
  "outputs": {},
  "resources": [],
  "check_results": null
}
@AbihaNadeem005 ➜ /workspaces/Lab10Repo (main) $
```

****