

5. Write IaC using terraform to create EC2 machine on aws or azure or google cloud. (Compulsory to use Input and output variable files)

- Name: Om Deore
- PRN: 22110318
- Div: TY(A)
- RollNo: 321022
- github: [All Assignments](#)

What is Terraform

HashiCorp Terraform is an **infrastructure as code** tool that lets you define both cloud and on-prem resources in human-readable configuration files that you can version, reuse, and share. You can then use a consistent workflow to provision and manage all of your infrastructure throughout its lifecycle. Terraform can manage low-level components like compute, storage, and networking resources, as well as high-level components like DNS entries and SaaS features.

Step-by-step screenshot to install and configure Terraform.

Step1) Installation

To install Terraform on debian linux use following scripts

```
sudo apt-get update && sudo apt-get install -y gnupg software-properties-common
wget -O- https://apt.releases.hashicorp.com/gpg | \
gpg --dearmor | \
sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg > /dev/null
gpg --no-default-keyring \
--keyring /usr/share/keyrings/hashicorp-archive-keyring.gpg \
--fingerprint
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] \
https://apt.releases.hashicorp.com $(lsb_release -cs) main" | \
sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update
sudo apt-get install terraform
```

```

om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> sudo apt-get update && sudo apt-get install -y gnupg software-properties-common
[sudo] password for om_d:
Hit:1 http://security.debian.org/debian-security bookworm-security InRelease
Hit:2 http://deb.debian.org/debian bookworm InRelease
Hit:3 http://deb.debian.org/debian bookworm-updates InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
gnupg is already the newest version (2.2.40-1.1).
gnupg set to manually installed.
software-properties-common is already the newest version (0.99.30-4).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.

om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> wget -O https://apt.releases.hashicorp.com/gpg | \
gpg --dearmor | \
sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg > /dev/null
--2024-04-18 13:42:04-- https://apt.releases.hashicorp.com/gpg
Resolving apt.releases.hashicorp.com (apt.releases.hashicorp.com)... 2600:9000:264c:2e00:18:566b:ecc0:93a1, 2600:9000:264c:9a00:18:566b:ecc0:93a1, 2600:9000:264c:7600:18:566b:ecc0:93a1, ...
Connecting to apt.releases.hashicorp.com (apt.releases.hashicorp.com)|2600:9000:264c:2e00:18:566b:ecc0:93a1|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3980 (3.9K) [binary/octet-stream]
Saving to: 'STDOUT'

[  100%[=====] 3.89K --.-KB/s   in 0s

2024-04-18 13:42:05 (643 MB/s) - written to stdout [3980/3980]

om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> gpg --no-default-keyring \
--keyring /usr/share/keyrings/hashicorp-archive-keyring.gpg \
--fingerprint
/usr/share/keyrings/hashicorp-archive-keyring.gpg
-----[CC] 0:[tmux]*                                     "deblinu" 13:43 18-Apr-24

om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> sudo apt-get 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 0 not upgraded.
Need to get 27.6 MB of archives.
After this operation, 88.2 MB of additional disk space will be used.
Get:1 https://apt.releases.hashicorp.com bookworm/main amd64 terraform amd64 1.8.1-1 [27.6 MB]
Fetched 27.6 MB in 2s (12.1 MB/s)
Selecting previously unselected package terraform.
(Reading database ... 285961 files and directories currently installed.)
Preparing to unpack .../terraform_1.8.1-1_amd64.deb ...
Unpacking terraform (1.8.1-1) ...
Setting up terraform (1.8.1-1) ...

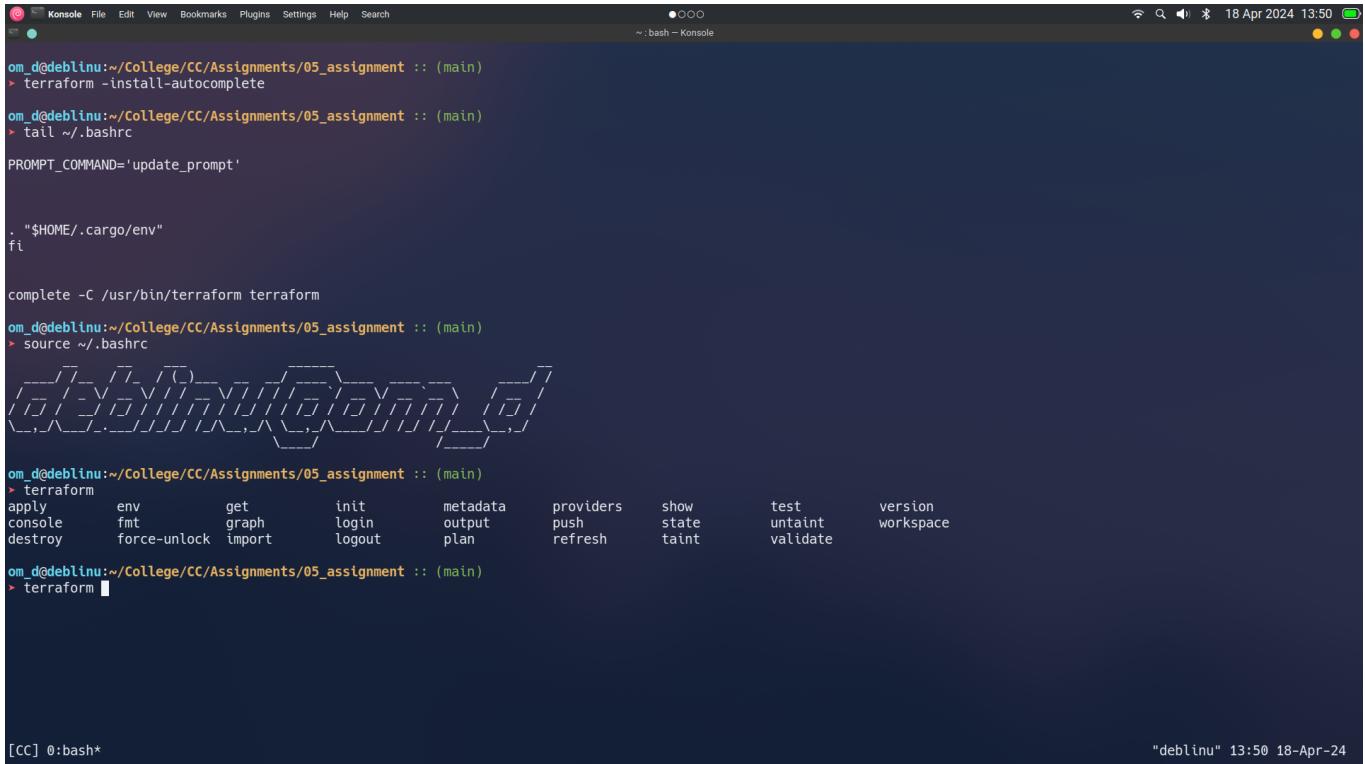
om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> terraform --version
Terraform v1.8.1
on linux_amd64
[CC] 0:bash*                                         "deblinu" 13:45 18-Apr-24

```

To check if it's installed correctly run

```
$ terraform --version
```

to enable auto complete run following command and source .bashrc to activate it



The screenshot shows a terminal window titled "Konsola" with the command "bash - Konsola". The terminal content is as follows:

```
om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> terraform -install-autocomplete
om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> tail ~/.bashrc
PROMPT_COMMAND='update_prompt'

. "$HOME/.cargo/env"
fi

complete -C /usr/bin/terraform terraform
om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> source ~/.bashrc
[Autocompletion icons: several arrows pointing right and down]
om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> terraform
apply   env      get      init      metadata  providers  show      test      version
console fmt      graph    login     output    push      state    untaint validate
destroy force-unlock import  logout    plan      refresh  taint
om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> terraform [TAB]
```

[CC] 0:bash* "deblinu" 13:50 18-Apr-24

Here we can see autocomplete being enabled after typing terraform and pressing tab key

Step 2) Install and Configure AWS cli

To install AWS cli run following script

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

To get AWS credentials from your account go to AWS console, click on profile>Security credentials>Access keys

Click on create Access key to create new access key. It will generate you new "access key" and a "secret access key"

```
export AWS_ACCESS_KEY_ID=<YOUR_ACCESS_KEY>
export AWS_SECRET_ACCESS_KEY_ID=<YOUR_SECRET_ACCESS_KEY>
```

Step 3) Terraform configuration

The set of files used to describe infrastructure in Terraform is known as a *Terraform configuration*. You will write your first configuration to define a single AWS EC2 instance.

Each Terraform configuration must be in its own working directory. Create a directory for your configuration.

in that directory create a `main.tf` file

```
terraform {  
    required_providers {  
        aws = {  
            source  = "hashicorp/aws"  
            version = "~> 4.16"  
        }  
    }  
  
    required_version = ">= 1.2.0"  
}  
  
provider "aws" {  
    region = "eu-north-1"  
}  
  
resource "aws_instance" "app_server" {  
    ami           = var.instance_ami_id  
    instance_type = var.instance_type  
  
    tags = {  
        Name = var.instance_name  
    }  
}
```

In this file we can see `var` being used. This is for input variables. In same directory create another `variables.tf` file

```
variable "instance_name" {  
    description = "Value of the Name tag for the EC2 instance"  
    type        = string  
    default     = "EC2_using_Terraform"  
}  
  
variable "instance_ami_id" {  
    description = "Amazon machine image id"  
    type        = string  
    default     = "ami-0914547665e6a707c"  
}
```

```
variable "instance_type" {
  description = "EC2 instance type"
  type        = string
  default     = "t3.micro"
}
```

Here we have created few variables `a` `instance_name`, `instance_ami_id`, `instance_type`. These variables are later used in `main.tf` file. Input variables are useful as they make terraform code modular and increase readability of our code.

To get outputs from terraform we use `outputs.tf` file

```
output "instance_id" {
  description = "ID of the EC2 instance"
  value       = aws_instance.app_server.id
}

output "instance_public_ip" {
  description = "Public IP address of the EC2 instance"
  value       = aws_instance.app_server.public_ip
}
```

In this demo we create two output variables `instance_id` & `instance_pubilc_ip`. When we apply our terraform configuration we get values of these output variables.

We have our configuration ready and we can launch our EC2 instance from terraform.

step 4) launch instance using Terraform

We have created all the configurations required to create our EC2 instance using terraform. Now only thing left are few commands

1. `terraform init`

When you create a new configuration — or check out an existing configuration from version control — you need to initialize the directory with `terraform init`. Initializing a configuration directory downloads and installs the providers defined in the configuration, which in this case is the `aws` provider. You can also make sure your configuration is syntactically valid and internally consistent by using the `terraform validate` command.

```

Konssole File Edit View Bookmarks Plugins Settings Help Search ●○○○
~:bash - Konssole
om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching ">= 4.16"...
- Installing hashicorp/aws v4.67.0... (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.

om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> terraform validate
Success! The configuration is valid.

om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> [CC] 0:bash*

```

"deblinu" 16:29 18-Apr-24

2. terraform apply

To launch instance use this command. Apply command applies any changes we made to existing configuration or launches new infrastructure if it's first time being run. It will display all the changes it will make to infrastructure and prompt you to say yes to reflect these changes.

```

Konsolle File Edit View Bookmarks Plugins Settings Help Search ●○○○
~:bash - Konssole
+ secondary_private_ips          = (known after apply)
+ security_groups                = (known after apply)
+ source_dest_check              = true
+ subnet_id                      = (known after apply)
+ tags                           =
  + "Name" = "EC2_using_Terraform"
+ tags_all                       = {
  + "Name" = "EC2_using_Terraform"
}
+ tenancy                         = (known after apply)
+ user_data                       = (known after apply)
+ user_data_base64                = (known after apply)
+ user_data_replace_on_change    = false
+ vpc_security_group_ids          = (known after apply)
}

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

Changes to Outputs:
+ instance_id        = (known after apply)
+ instance_public_ip = (known after apply)

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_instance.app_server: Creating...
aws_instance.app_server: Still creating... [10s elapsed]
aws_instance.app_server: Creation complete after 19s [id=i-02d76e30b955d4b01]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

instance_id = "i-02d76e30b955d4b01"
instance_public_ip = "13.49.64.94"

om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> terraform output
instance_id = "i-02d76e30b955d4b01"
instance_public_ip = "13.49.64.94"

om_d@deblinu:~/College/CC/Assignments/05_assignment :: (main)
> [CC] 0:bash*

```

"deblinu" 16:43 18-Apr-24

Once this command runs successfully, we can see values to our `output` variables being printed. To see these variables again run `terraform output` command

The state of our infrastructure is saved in file called `terraform.tfstate`

We can see our instance being launched

Before apply command

A screenshot of the AWS EC2 Instances page. The left sidebar shows navigation options like EC2 Dashboard, Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Images (AMIs, Catalog), Elastic Block Store (Volumes, Snapshots, Lifecycle Manager), and Network & Security (Security Groups, Elastic IPs). The main content area shows a table titled "Instances (1/1) Info" with one row. The instance is named "CC-Assignment" with ID "i-058d3efc0d0d1d19c". It is currently "Stopped". The "Actions" dropdown menu is open, showing options like "Launch instances", "Stop", "Start", "Reboot", "Terminate", "Detach volume", "Change instance type", "Change subnet", "Change security group", "Change key pair", "Change network interface", and "Change volume type". Below the table, there is a detailed view for the selected instance.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
CC-Assignment	i-058d3efc0d0d1d19c	Stopped	t3.micro	-	View alarms +	eu-north-1b	-

Instance: i-058d3efc0d0d1d19c (CC-Assignment)

Allow tags in instance metadata Disabled	Use RBN as guest OS hostname Disabled	Answer RBN DNS hostname IPv4 Enabled
Host and placement group		
Host ID -	Affinity -	Placement group -
Host resource group name -	Tenancy default	Placement group ID -
Virtualization type hvm	Reservation r-0136d9fc655f3a6d6	Partition number -

After apply command

A screenshot of the AWS EC2 Instances page after applying changes. The left sidebar is identical to the previous screenshot. The main content area shows a table titled "Instances (1/2) Info" with two rows. The first instance is "EC2_using_Terraform" with ID "i-02d76e30b955d4b01", currently "Running". The second instance is "CC-Assignment" with ID "i-058d3efc0d0d1d19c", currently "Stopped". The "Actions" dropdown menu is open, showing options like "Launch instances", "Stop", "Start", "Reboot", "Terminate", "Detach volume", "Change instance type", "Change subnet", "Change security group", "Change key pair", "Change network interface", and "Change volume type". Below the table, there is a detailed view for the running instance.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
EC2_using_Terraform	i-02d76e30b955d4b01	Running	t3.micro	Initializing	View alarms +	eu-north-1a	ec2-13-49-64-94.eu.no
CC-Assignment	i-058d3efc0d0d1d19c	Stopped	t3.micro	-	View alarms +	eu-north-1b	-

Instance: i-02d76e30b955d4b01 (EC2_using_Terraform)

Details	Status and alarms	Monitoring	Security	Networking	Storage	Tags
Instance summary						
Instance details						
Platform Ubuntu (Inferred)	AMI ID ami-0914547665e6a707c	Monitoring disabled				
Platform details Linux/UNIX	AMI name ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-20240301	Termination protection Disabled				
Stop protection Disabled	Launch time Thu Apr 18 2024 16:41:22 GMT+0530 (India Standard Time) (1 minute)	AMI location amazon/ubuntu/images/hvm-ssd/ubuntu-jammy-22.04-amd64-server-20240301				
Instance auto-recovery Default	Lifecycle normal	Stop-hibernate behavior Disabled				
AMI launch index	Key pair assigned at launch	State transition reason				