export AWS\_ACCESS\_KEY\_ID=AKIAQE43KAB4DADFZN5H

export AWS\_SECRET\_ACCESS\_KEY=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

.env

<https://github.com/paulgitrepo/mprashant-terraform-configs.git>

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_1

$ source .env

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_1

$ aws iam list-users

{

"Users": []

}

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

main.tf

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_instance" "myserver" {

  ami = "ami-0e35ddab05955cf57"

  instance\_type = "t2.micro"

  tags = {

    Name = "sample server"

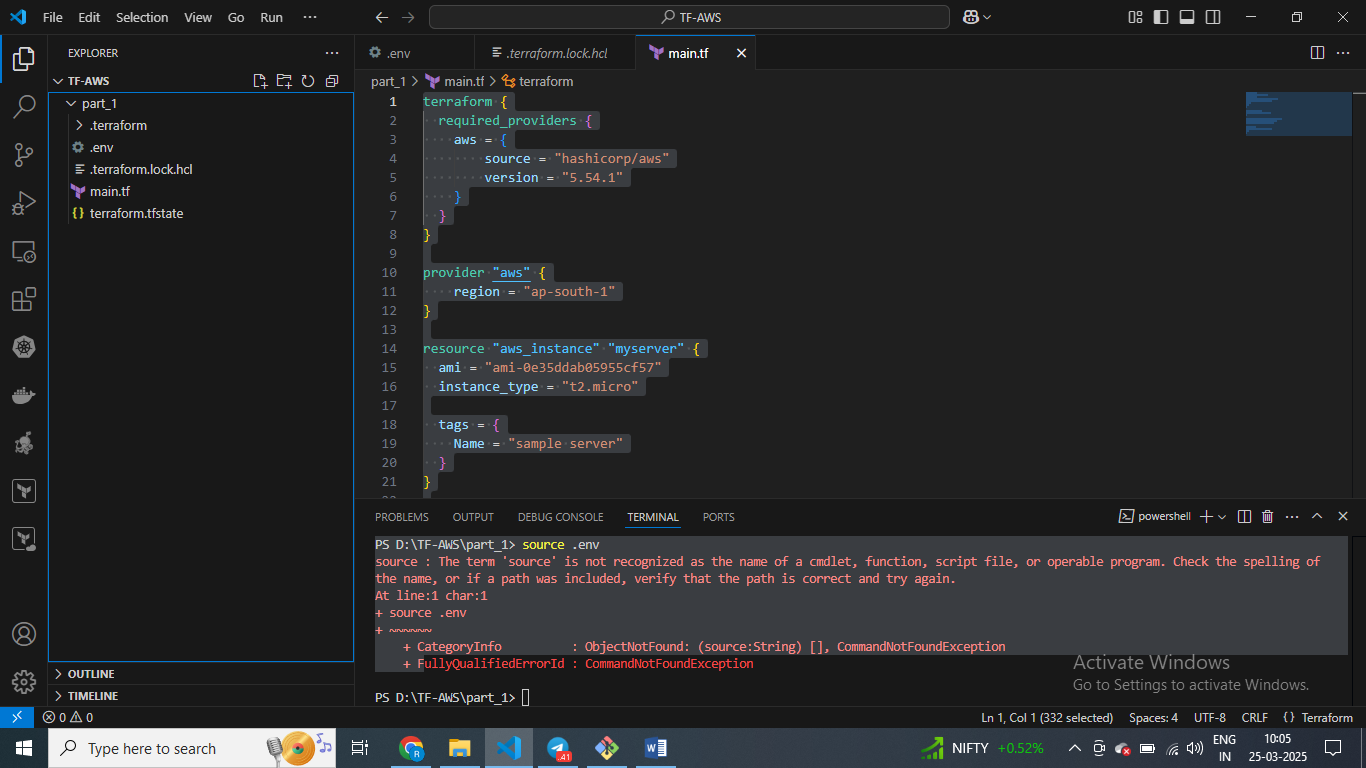
  }

}

Terraform init

Terraform plan

Terraform apply



# just create main.tf file and and .env file

RESORUCE CHANGES

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_instance" "myserver" {

  ami = "ami-0e35ddab05955cf57"

  instance\_type = "t3.micro"

  tags = {

    Name = "sample server"

  }

}

Main.tf

Terraform plan

Terraform apply

$ terraform destroy

**aws\_instance.myserver: Refreshing state... [id=i-054ae078dcfdc01f8]**

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

}

providrr "aws" {

    region = "ap-south-1"

}

resource "aws\_instance" "myserver" {

  ami = "ami-0e35ddab05955cf57"

  instance\_type = "t3.micro"

  tags = {

    Name = "sample server"

  }

}

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_1

$ terraform validate

╷

│ **Error: Unsupported block type**

│

│ on main.tf line 10:

│ 10: providrr "aws" {

│

│ Blocks of type "providrr" are not expected here. Did you mean "provider"?

╵

Part\_2 variables

variable "region" {

  description = "value of region"

  type = string

  default = "ap-south-1"

}

Main.tf

variable "region" {

  description = "value of region"

  type = string

  default = "ap-south-1"

}

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

}

provider "aws" {

    region = var.region

}

resource "aws\_instance" "myserver" {

  ami = "ami-0e35ddab05955cf57"

  instance\_type = "t3.micro"

  tags = {

    Name = "sample server"

  }

}

Main.tf

ser121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS

$ cd part\_2\_variables

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_2\_variables

$ terraform validate

╷

│ **Error: Missing required provider**

│

│ This configuration requires provider registry.terraform.io/hashicorp/aws, but that provider isn't available. You may be able to install it automatically by running:

│ terraform init

╵

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_2\_variables

$ terraform init

**Initializing the backend...**

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_2\_variables

$ terraform validate

**Success!** The configuration is valid.

variable "region" {

  description = "value of region"

  type = string

  default = "ap-south-1"

}

Variables.tf

Terraform validate

Part 3 OUTPUT Block

output "aws\_instance\_public\_ip" {

    value = aws\_instance.myserver.public\_ip

}

Outputs.tf

Terraform init

Terraform validate

Terraform plan

Terraform apply

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS

$ cd part\_3\_outputs

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_3\_outputs

$ terraform init

**Initializing the backend...**

**Initializing provider plugins...**

- Finding hashicorp/aws versions matching "5.54.1"...

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_3\_outputs

$ terraform apply

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:

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_3\_outputs

$ terraform apply

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

+ create

AWS S3 PART 4

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_s3\_bucket" "demo-buckets" {

  bucket = "demo-buckets-rusvesv"

}

Main.tf

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS

$ cd aws\_s3

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3

$ cd part\_1

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$ terraform init

**Initializing the backend...**

**Initializing provider plugins...**

- Finding hashicorp/aws versions matching "5.54.1"...

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$ terraform plan

**No changes. Your infrastructure matches the configuration.**

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$ terraform apply

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:

**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\_s3\_bucket.demo-buckets: Creating...**

**aws\_s3\_bucket.demo-buckets: Creation complete after 2s [id=demo-buckets-rusvesv]**

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

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_s3\_bucket" "demo-buckets" {

  bucket = "demo-buckets-rusvesv"

}

resource "aws\_s3\_object" "bucket-data" {

  bucket = aws\_s3\_bucket.demo-buckets.bucket

  source = "./myfile.txt" #upl

  key = "mydata.txt"

}

#uploading file to s3 buecket as mydata.txt

Terraform apply

#Rndom provider

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

    #using random provider

     random = {

      source = "hashicorp/random"

      version = "3.5.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_s3\_bucket" "demo-buckets" {

  bucket = "demo-buckets-rusvesv"

}

resource "aws\_s3\_object" "bucket-data" {

  bucket = aws\_s3\_bucket.demo-buckets.bucket

  source = "./myfile.txt" #upl

  key = "mydata.txt"

}

resource "random\_id" "rand\_id" {

  byte\_length = 8

}

output "name" {

  value = random\_id.rand\_id.b64\_url

}

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$ terraform apply

╷

│ **Error: Inconsistent dependency lock file**

│

│ The following dependency selections recorded in the lock file are inconsistent with the current configuration:

│ - provider registry.terraform.io/hashicorp/random: required by this co

╵

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$ terraform init -upgrade

**Initializing the backend...**

**Initializing provider plugins...**

- Finding hashicorp/random versions matching "3.5.1"...

- Finding hashicorp/aws versions matching "5.54.1"...

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$ terraform apply

**aws\_s3\_bucket.demo-buckets: Refreshing state... [id=demo-buckets-rusvesv]**

**aws\_s3\_object.bucket-data: Refreshing state... [id=mydata.txt]**

+ dec = (known after apply)

+ hex = (known after apply)

+ id = (known after apply)

}

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

Changes to Outputs:

+ name = (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

**random\_id.rand\_id: Creating...**

**random\_id.rand\_id: Creation complete after 0s [id=HfS31J3kmlI]**

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

**Outputs:**

name = "HfS31J3kmlI"

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

    #using random provider

     random = {

      source = "hashicorp/random"

      version = "3.5.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_s3\_bucket" "demo-buckets" {

  bucket = "demo-buckets-rusvesv"

}

resource "aws\_s3\_object" "bucket-data" {

  bucket = aws\_s3\_bucket.demo-buckets.bucket

  source = "./myfile.txt" #upl

  key = "mydata.txt"

}

resource "random\_id" "rand\_id" {

  byte\_length = 8

}

output "name" {

  value = random\_id.rand\_id.hex

}

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$ terraform apply

**aws\_s3\_bucket.demo-buckets: Refreshing state... [id=demo-buckets-rusvesv]**

**aws\_s3\_object.bucket-data: Refreshing state... [id=mydata.txt]**

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

**Outputs:**

name = "1df4b7d49de49a52"

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

    #using random provider

     random = {

      source = "hashicorp/random"

      version = "3.5.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_s3\_bucket" "demo-buckets" {

  bucket = "demo-buckets-rusvesv-${random\_id.rand\_id.hex}"

}

resource "aws\_s3\_object" "bucket-data" {

  bucket = aws\_s3\_bucket.demo-buckets.bucket

  source = "./myfile.txt" #upl

  key = "mydata.txt"

}

resource "random\_id" "rand\_id" {

  byte\_length = 8

}

output "name" {

  value = random\_id.rand\_id.hex

}

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/aws\_s3/part\_1

$ terraform apply

**random\_id.rand\_id: Refreshing state... [id=HfS31J3kmlI]**

**aws\_s3\_bucket.demo-buckets: Refreshing state... [id=demo-buckets-rusvesv]**

**aws\_s3\_object.bucket-data: Refreshing state... [id=mydata.txt]**

part 5 rmoete state maagemetn

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

  backend "s3" {

    bucket = "demo-buckets-rusvesv-1df4b7d49de49a52"

    key = "backend.tfstate"

    region = "ap-south-1"

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_instance" "myserver" {

  ami = "ami-0e35ddab05955cf57"

  instance\_type = "t3.micro"

  tags = {

    Name = "sample server"

  }

}

Main.tf

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS

$ cd part\_5\_remote\_tf\_backend

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_5\_remote\_tf\_backend

$ terraform init

'**Initializing the backend...**

Successfully configured the backend "s3"! Terraform will automatically

use this backend unless the backend configuration changes.

**Initializing provider plugins...**

- Finding hashicorp/aws versions matching "5.54.1"...

- Installing hashicorp/aws v5.54.1...

- Installed hashicorp/aws v5.54.1 (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.

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_5\_remote\_tf\_backend

$ terraform apply'

> ^C

user121@DESKTOP-JFJ21I5 MINGW64 /d/TF-AWS/part\_5\_remote\_tf\_backend

$ terraform apply

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

+ create

Part- 6 project static website hosting using terraform

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

    #using random provider

     random = {

      source = "hashicorp/random"

      version = "3.5.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_s3\_bucket" "demo-buckets" {

  bucket = "demo-buckets-${random\_id.rand\_id.hex}"

}

resource "aws\_s3\_object" "index\_html" {

  bucket = aws\_s3\_bucket.demo-buckets.bucket

  source = "./index.html" #upl

  key = "index.html"

}

resource "aws\_s3\_object" "style\_css" {

  bucket = aws\_s3\_bucket.demo-buckets.bucket

  source = "./style.css"

  key = "style.css"

}

resource "random\_id" "rand\_id" {

  byte\_length = 8

}

output "name" {

  value = random\_id.rand\_id.hex

}

resource "aws\_s3\_bucket\_public\_access\_block" "example" {

  bucket = aws\_s3\_bucket.demo-buckets.id

  block\_public\_acls = false

  block\_public\_policy = false

  ignore\_public\_acls = false

  restrict\_public\_buckets = false

}

Source.env

Terraform init

Plan

Validate

Applu

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

    random = {

      source = "hashicorp/random"

      version = "3.5.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_s3\_bucket" "demo\_buckets" {

  bucket = "demo-buckets-${random\_id.rand\_id.hex}"

}

resource "aws\_s3\_object" "index\_html" {

  bucket = aws\_s3\_bucket.demo\_buckets.bucket

  source = "./index.html"

  key    = "index.html"

}

resource "aws\_s3\_object" "style\_css" {

  bucket = aws\_s3\_bucket.demo\_buckets.bucket

  source = "./style.css"

  key    = "style.css"

}

resource "random\_id" "rand\_id" {

  byte\_length = 8

}

output "name" {

  value = random\_id.rand\_id.hex

}

resource "aws\_s3\_bucket\_public\_access\_block" "example" {

  bucket                  = aws\_s3\_bucket.demo\_buckets.id

  block\_public\_acls       = false

  block\_public\_policy     = false

  ignore\_public\_acls      = false

  restrict\_public\_buckets = false

}

resource "aws\_s3\_bucket\_policy" "demo-buckets" {

  bucket = aws\_s3\_bucket.demo\_buckets.id

  policy = jsonencode({

    "Version": "2012-10-17",

    "Statement": [

      {

        "Sid": "PublicReadGetObject",

        "Effect": "Allow",

        "Principal": "\*",

        "Action": "s3:GetObject",

        "Resource": "arn:aws:s3:::${aws\_s3\_bucket.demo\_buckets.id}/\*"

      }

    ]

  })

}

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

    random = {

      source = "hashicorp/random"

      version = "3.5.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_s3\_bucket" "demo\_buckets" {

  bucket = "demo-buckets-${random\_id.rand\_id.hex}"

}

resource "aws\_s3\_object" "index\_html" {

  bucket = aws\_s3\_bucket.demo\_buckets.bucket

  source = "./index.html"

  key    = "index.html"

  content\_type = "text/html"

}

resource "aws\_s3\_object" "style\_css" {

  bucket = aws\_s3\_bucket.demo\_buckets.bucket

  source = "./style.css"

  key    = "style.css"

  content\_type = "text/css"

}

resource "random\_id" "rand\_id" {

  byte\_length = 8

}

output "name" {

  value = random\_id.rand\_id.hex

}

resource "aws\_s3\_bucket\_public\_access\_block" "example" {

  bucket                  = aws\_s3\_bucket.demo\_buckets.id

  block\_public\_acls       = false

  block\_public\_policy     = false

  ignore\_public\_acls      = false

  restrict\_public\_buckets = false

}

resource "aws\_s3\_bucket\_policy" "demo-buckets" {

  bucket = aws\_s3\_bucket.demo\_buckets.id

  policy = jsonencode({

    "Version": "2012-10-17",

    "Statement": [

      {

        "Sid": "PublicReadGetObject",

        "Effect": "Allow",

        "Principal": "\*",

        "Action": "s3:GetObject",

        "Resource": "arn:aws:s3:::${aws\_s3\_bucket.demo\_buckets.id}/\*"

      }

    ]

  })

}

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

    random = {

      source = "hashicorp/random"

      version = "3.5.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_s3\_bucket" "demo\_buckets" {

  bucket = "demo-buckets-${random\_id.rand\_id.hex}"

}

resource "aws\_s3\_object" "index\_html" {

  bucket = aws\_s3\_bucket.demo\_buckets.bucket

  source = "./index.html"

  key    = "index.html"

  content\_type = "text/html"

}

resource "aws\_s3\_object" "style\_css" {

  bucket = aws\_s3\_bucket.demo\_buckets.bucket

  source = "./style.css"

  key    = "style.css"

  content\_type = "text/css"

}

resource "random\_id" "rand\_id" {

  byte\_length = 8

}

output "name" {

  value = random\_id.rand\_id.hex

}

resource "aws\_s3\_bucket\_public\_access\_block" "example" {

  bucket                  = aws\_s3\_bucket.demo\_buckets.id

  block\_public\_acls       = false

  block\_public\_policy     = false

  ignore\_public\_acls      = false

  restrict\_public\_buckets = false

}

resource "aws\_s3\_bucket\_policy" "demo-buckets" {

  bucket = aws\_s3\_bucket.demo\_buckets.id

  policy = jsonencode({

    "Version": "2012-10-17",

    "Statement": [

      {

        "Sid": "PublicReadGetObject",

        "Effect": "Allow",

        "Principal": "\*",

        "Action": "s3:GetObject",

        "Resource": "arn:aws:s3:::${aws\_s3\_bucket.demo\_buckets.id}/\*"

      }

    ]

  })

}

# output "name1" {

#   value = aws\_s3\_bucket\_website\_configuration.demo\_buckets.website\_endpoints

# }

resource "aws\_s3\_bucket\_website\_configuration" "website\_config" {

  bucket = aws\_s3\_bucket.demo\_buckets.id

  index\_document {

    suffix = "index.html"

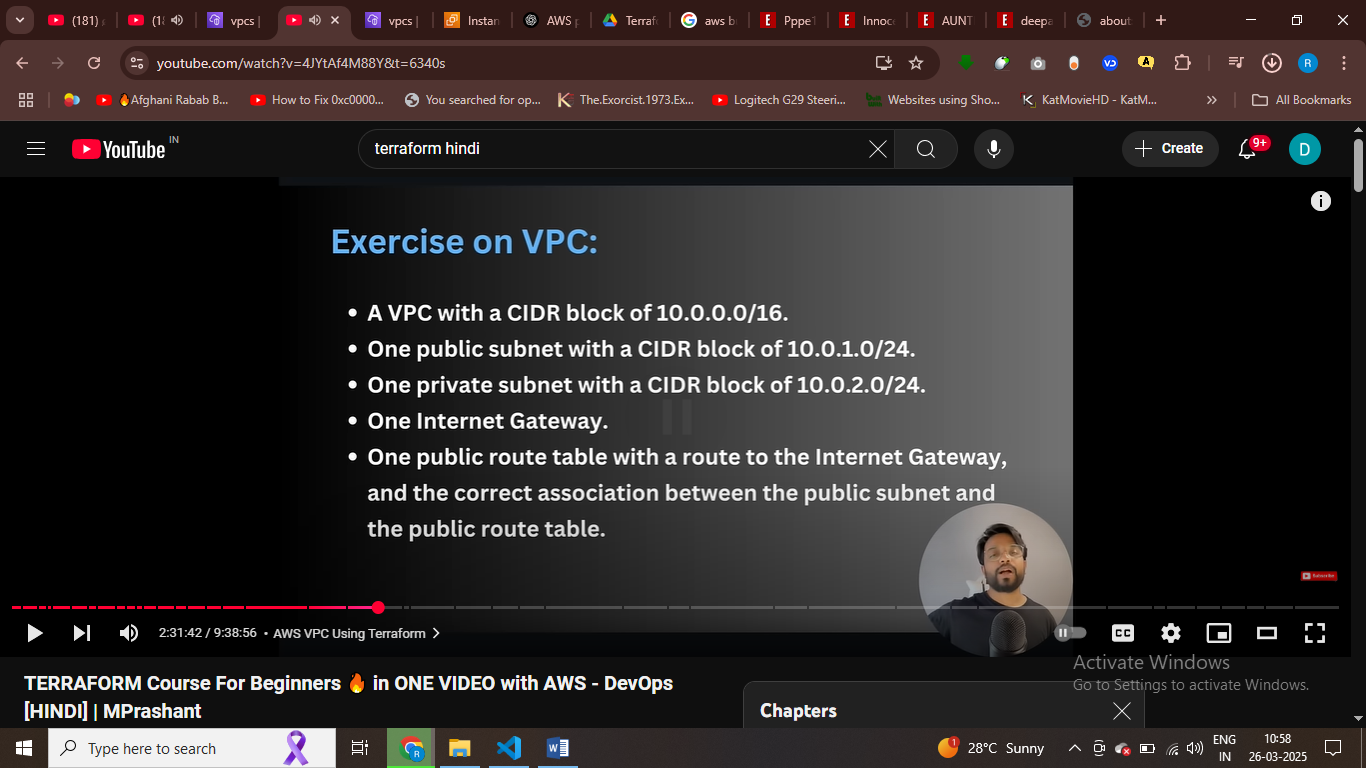
  }

}

output "website\_url" {

  value = aws\_s3\_bucket\_website\_configuration.website\_config.website\_endpoint

}



Isndie vpc we define cdr which hai 10.0.0.0/16 – 65k ip addres , /24 – 32k ip addres

These ip addres divide into subnet

Part 7 crateing a vpc and attach to subents and route tables

terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_vpc" "my\_vpc" {

  cidr\_block = "10.0.0.0/16" #65k ip address it has

  tags = {

    Name = "my\_vpc"

  }

}

resource "aws\_subnet" "private-subnet" {

  cidr\_block = "10.0.1.0/24"

  vpc\_id = aws\_vpc.my\_vpc.id #part of this vpc

  tags = {

    Name = "private-subnet"

  }

}

resource "aws\_subnet" "public-subnet" {

  cidr\_block = "10.0.2.0/24"

  vpc\_id = aws\_vpc.my\_vpc.id #part of this vpc

  tags = {

    Name = "public-subnet"

  }

}

#Internet gateway

resource "aws\_internet\_gateway" "my-igw" {

  vpc\_id = aws\_vpc.my\_vpc.id #internet gateway attach to this vpc

  tags = {

    Name = "my-igw"

  }

}

#Routing table -  to use ingtente gateway

resource "aws\_route\_table" "my-rt" {

  vpc\_id = aws\_vpc.my\_vpc.id #work for this vpc

  route  {

    cidr\_block = "0.0.0.0/0"

    gateway\_id = aws\_internet\_gateway.my-igw.id

  }

}

#which subent work with diffrent routing table

resource "aws\_route\_table\_association" "public\_sub" {

  route\_table\_id = aws\_route\_table.my-rt.id

  subnet\_id = aws\_subnet.public-subnet.id

}

Terraform {

  required\_providers {

    aws = {

        source = "hashicorp/aws"

        version = "5.54.1"

    }

  }

}

provider "aws" {

    region = "ap-south-1"

}

resource "aws\_vpc" "my\_vpc" {

  cidr\_block = "10.0.0.0/16" #65k ip address it has

  tags = {

    Name = "my\_vpc"

  }

}

resource "aws\_subnet" "private-subnet" {

  cidr\_block = "10.0.1.0/24"

  vpc\_id = aws\_vpc.my\_vpc.id #part of this vpc

  tags = {

    Name = "private-subnet"

  }

}

resource "aws\_subnet" "public-subnet" {

  cidr\_block = "10.0.2.0/24"

  vpc\_id = aws\_vpc.my\_vpc.id #part of this vpc

  tags = {

    Name = "public-subnet"

  }

}

#Internet gateway

resource "aws\_internet\_gateway" "my-igw" {

  vpc\_id = aws\_vpc.my\_vpc.id #internet gateway attach to this vpc

  tags = {

    Name = "my-igw"

  }

}

#Routing table -  to use ingtente gateway

resource "aws\_route\_table" "my-rt" {

  vpc\_id = aws\_vpc.my\_vpc.id #work for this vpc

  route  {

    cidr\_block = "0.0.0.0/0"

    gateway\_id = aws\_internet\_gateway.my-igw.id

  }

}

#which subent work with diffrent routing table

resource "aws\_route\_table\_association" "public\_sub" {

  route\_table\_id = aws\_route\_table.my-rt.id

  subnet\_id = aws\_subnet.public-subnet.id

}

#creating an instance

resource "aws\_instance" "myserver" {

  ami = "ami-0e35ddab05955cf57"

  instance\_type = "t3.micro"

  subnet\_id = aws\_subnet.public-subnet.id

  tags = {

    Name = "sample server"

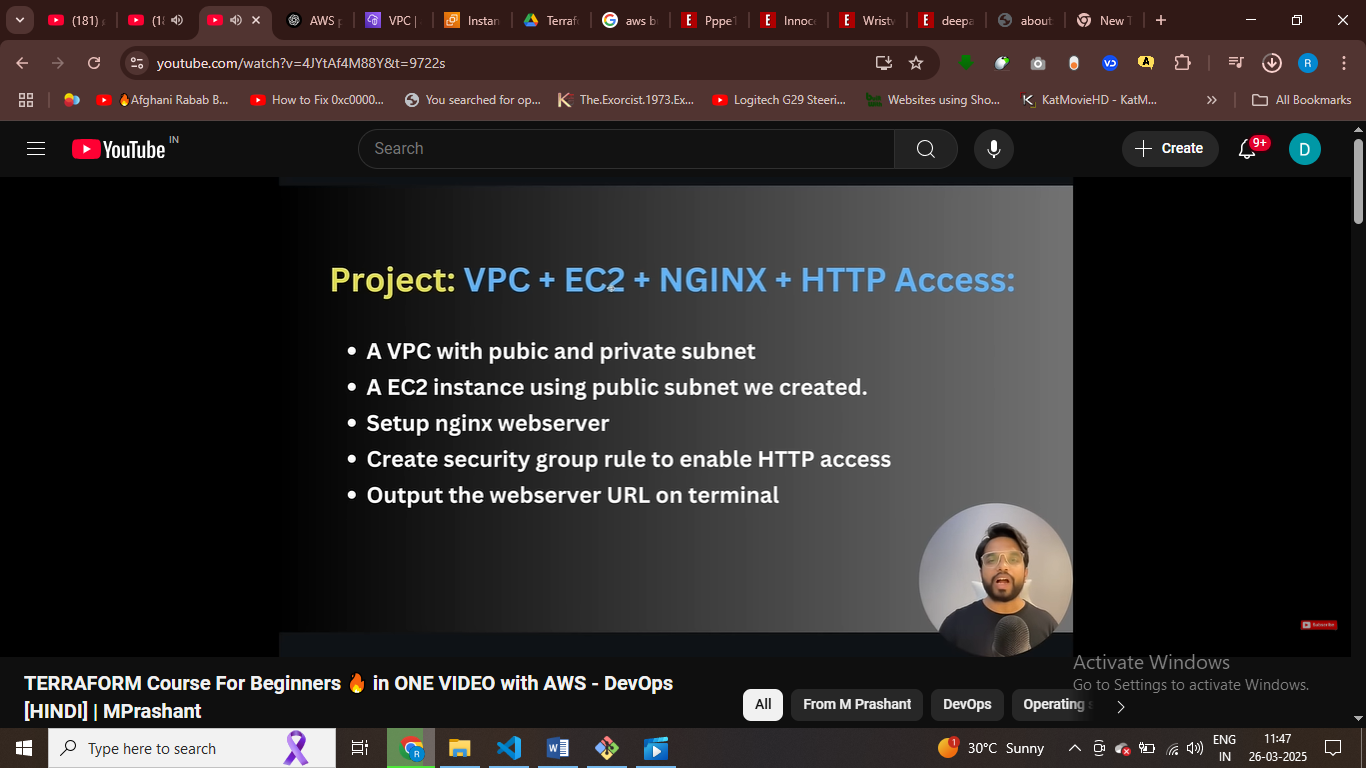
  }

}

Init

Plan

Apply



Part 6 proejct

Part 7 data soruces