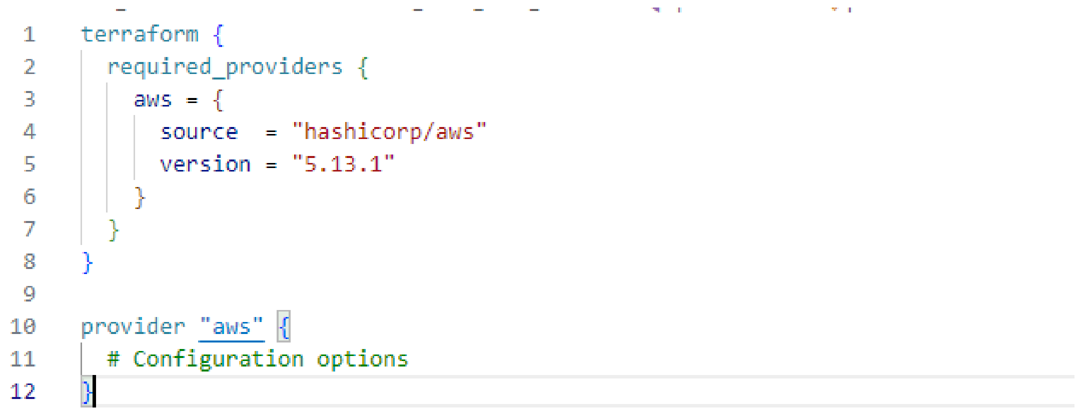
Que 1 →

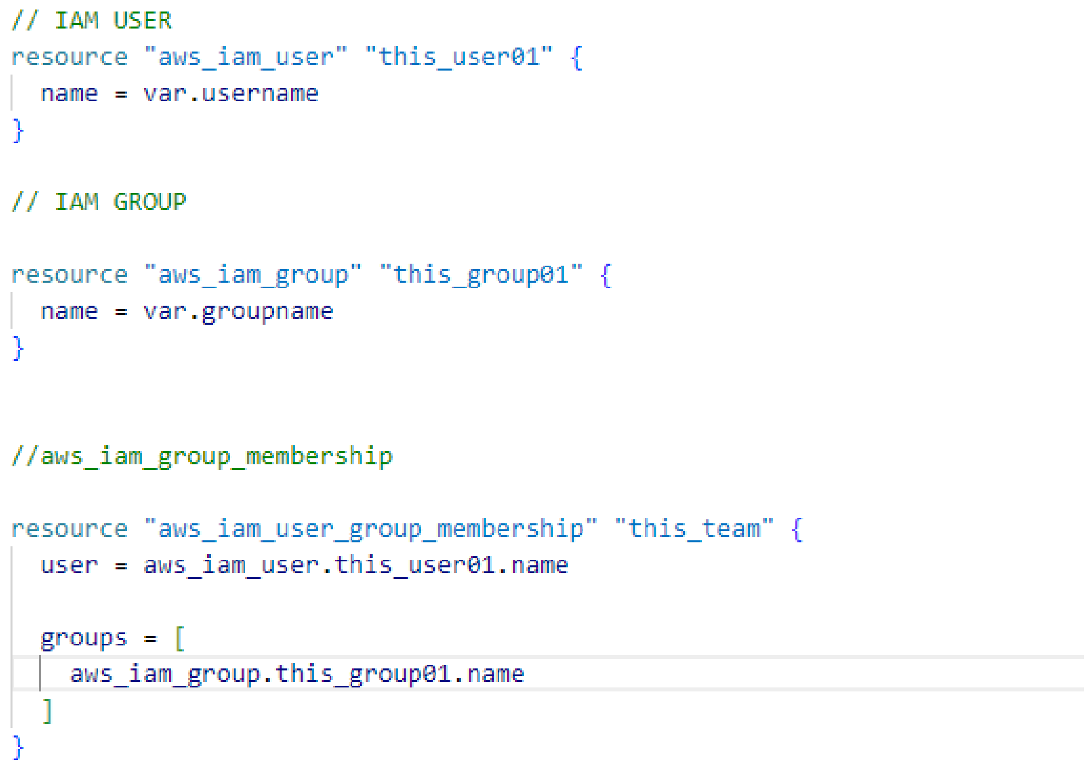
- Create one IAM user and one IAM Group using Terraform.
- Make sure you will use variables for names of IAM users and Group.
- Note: - Below files are required. –

Providers.tf



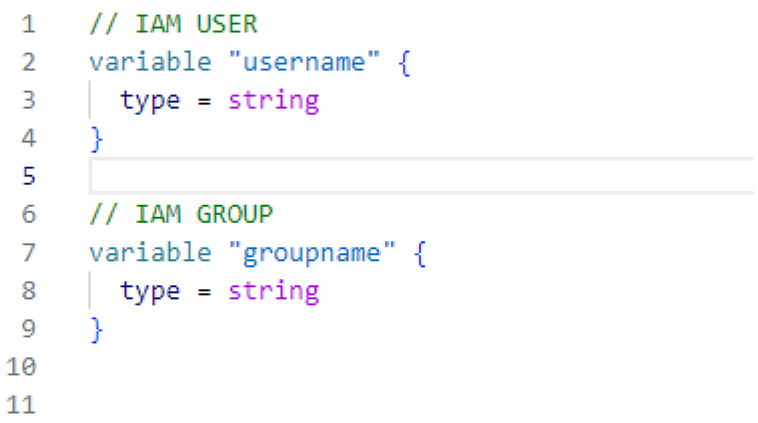
```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "5.13.1"
    }
  }
}

provider "aws" {
  # Configuration options
}
```

Main.tf



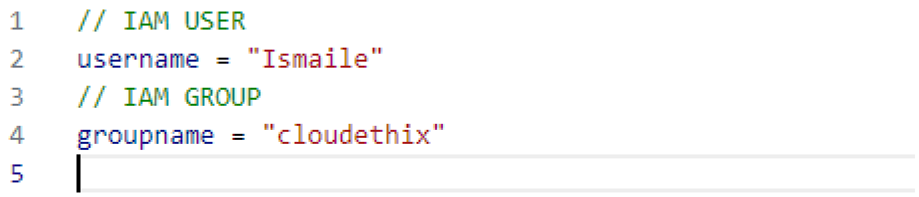
```
// IAM USER
resource "aws_iam_user" "this_user01" {
  name = var.username
}

// IAM GROUP

resource "aws_iam_group" "this_group01" {
  name = var.groupname
}


//aws_iam_group_membership

resource "aws_iam_user_group_membership" "this_team" {
  user = aws_iam_user.this_user01.name

  groups = [
    aws_iam_group.this_group01.name
  ]
}
```

Variables.tf



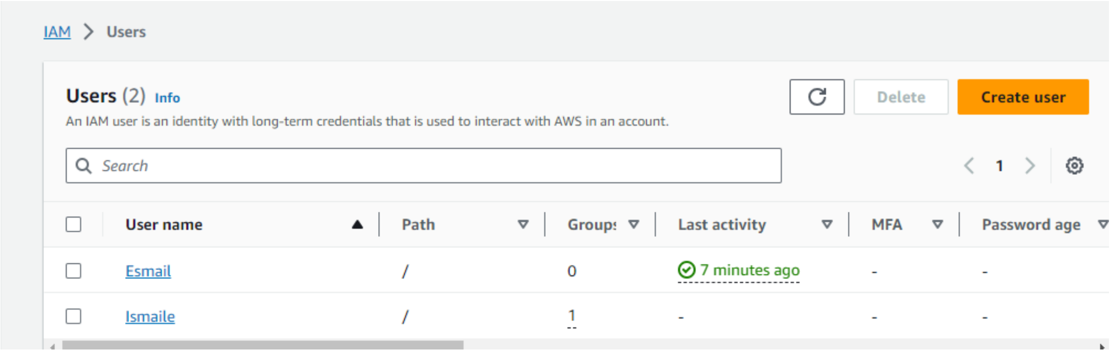
```
// IAM USER
variable "username" {
  type = string
}

// IAM GROUP
variable "groupname" {
  type = string
}


```

esmail_custom.tfvars



```
// IAM USER
username = "Ismaile"
// IAM GROUP
groupname = "cloudethix"

```

**Result –**

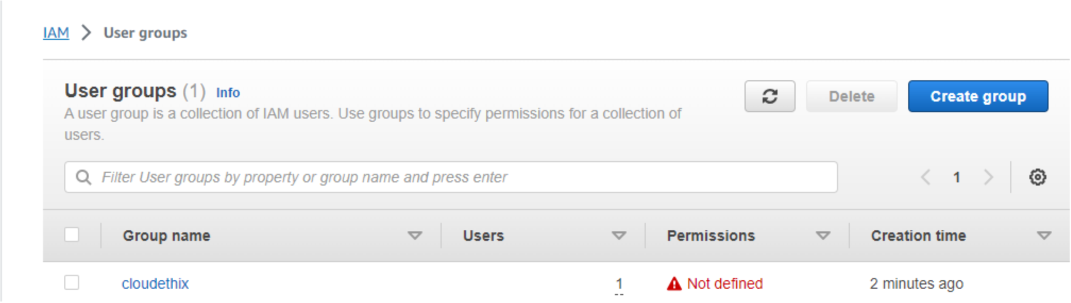


IAM > Users

Users (2) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

| □ | User name | Path | Groups | Last activity | MFA | Password age |
|---|---|---|---|---|---|---|
| □ | Esmail | / | 0 | ✓ 7 minutes ago | - | - |
| □ | Ismaile | / | 1 | - | - | - |



IAM > User groups

User groups (1) Info

A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.

Delete
Create group
Filter User groups by property or group name and press enter
1

| □ | Group name | Users | Permissions | Creation time |
|---|---|---|---|---|
| □ | cloudethix | 1 | ⚠ Not defined | 2 minutes ago |

Que 2 →

● Create one EC2 Instance and Elastic IP using Terraform and Map elastic IP with EC2 instance.

● Also please make sure you will use a combination of both variables in the main.tf file.

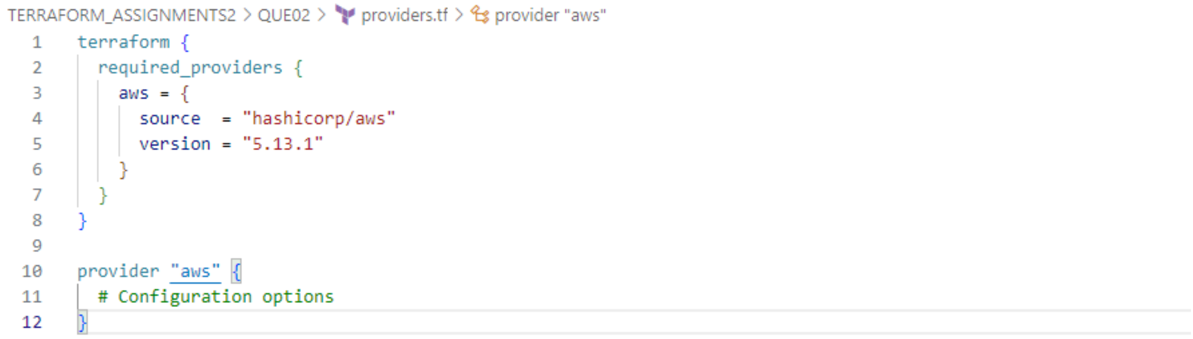
- i.e. local and variable from variables.tf and custom.tfvars file.

● Also use output values to print EC2 instances Public DNS name, Private DNS name, Private IP and Public IP.

● Note: -

- Here you will require one locals in the main.tf file.
- Also, four output values in the main.tf file

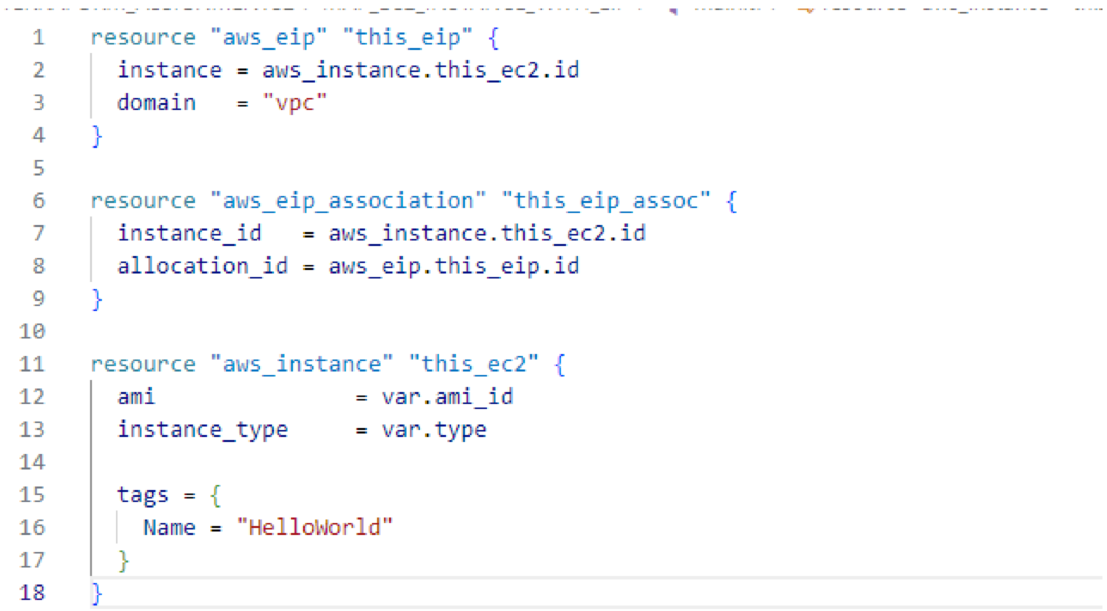
Providers.tf



TERRAFORM_ASSIGNMENTS2 > QUE02 > providers.tf > provider "aws"

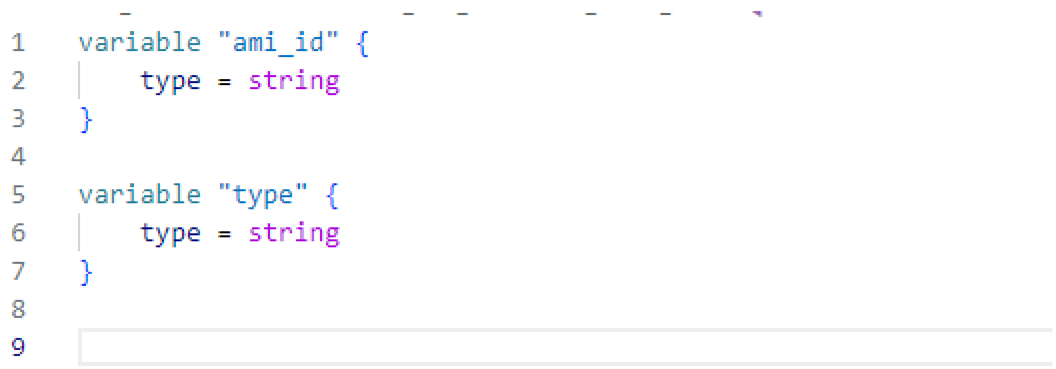
```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "5.13.1"
    }
  }
}

provider "aws" {
  # Configuration options
}
```

Main.tf



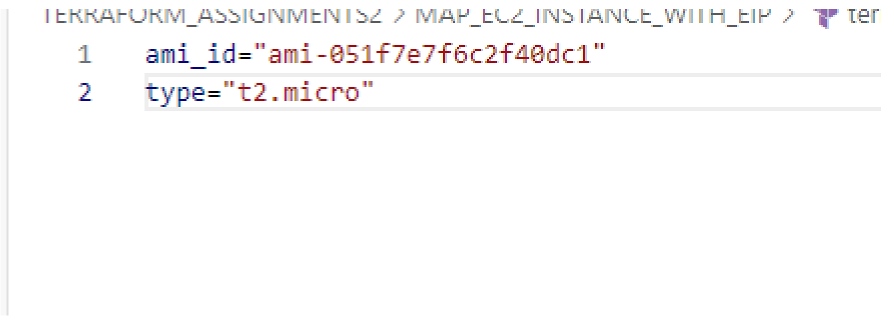
```
resource "aws_eip" "this_eip" {
  instance = aws_instance.this_ec2.id
  domain   = "vpc"
}

resource "aws_eip_association" "this_eip_assoc" {
  instance_id   = aws_instance.this_ec2.id
  allocation_id = aws_eip.this_eip.id
}

resource "aws_instance" "this_ec2" {
  ami                  = var.ami_id
  instance_type        = var.type

  tags = {
    Name = "HelloWorld"
  }
}
```

Variables.tf



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

variable "type" {
    type = string
}

```

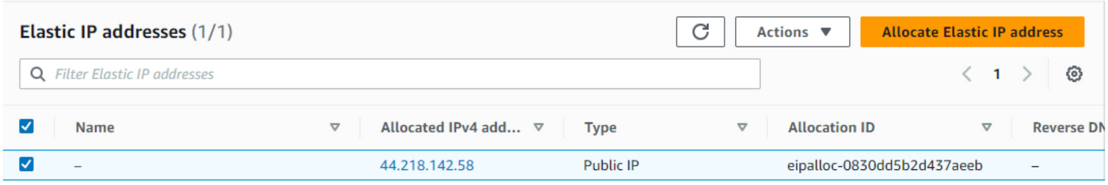
## Terraform.tfvars



TERRAFORM_ASSIGNMENTS2 > MAP_EC2_INSTANCE_WITH_EIP > ter

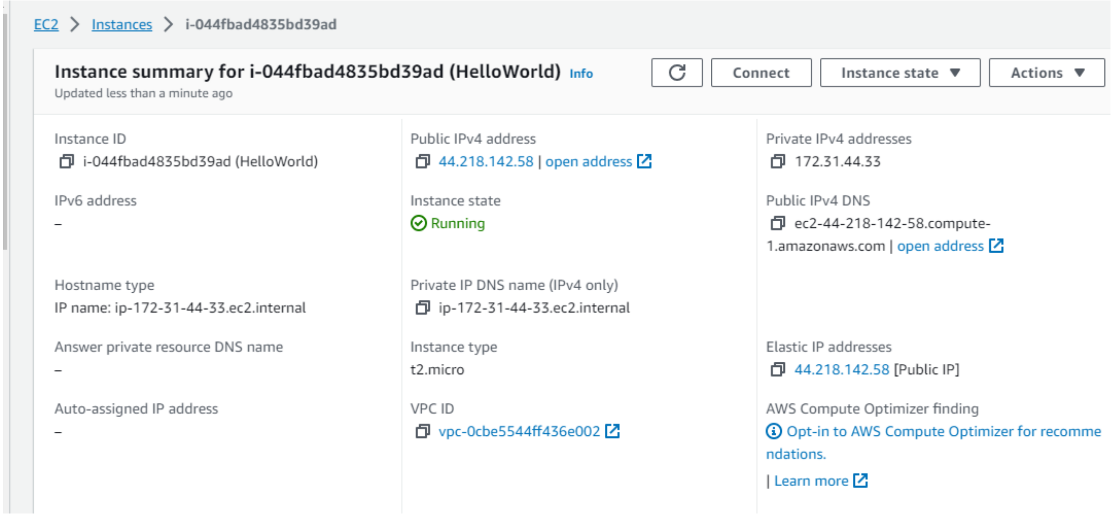
```
ami_id="ami-051f7e7f6c2f40dc1"
type="t2.micro"
```

## Result



Elastic IP addresses (1/1)
Actions ▼
Allocate Elastic IP address
Filter Elastic IP addresses
1

| ☑ | Name | Allocated IPv4 add... | Type | Allocation ID | Reverse DN |
|---|---|---|---|---|---|
| ☑ | – | 44.218.142.58 | Public IP | eipalloc-0830dd5b2d437aeeb | – |



Instance summary for i-044fbad4835bd39ad (HelloWorld) Info

Updated less than a minute ago

Instance ID
i-044fbad4835bd39ad (HelloWorld)

Public IPv4 address
44.218.142.58 | open address

Private IPv4 addresses
172.31.44.33

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-44-218-142-58.compute-1.amazonaws.com | open address

Hostname type
IP name: ip-172-31-44-33.ec2.internal

Private IP DNS name (IPv4 only)
ip-172-31-44-33.ec2.internal

Answer private resource DNS name
–

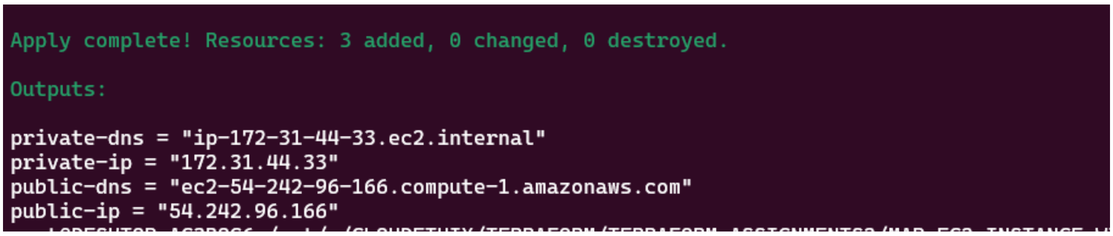
Instance type
t2.micro

Elastic IP addresses
44.218.142.58 [Public IP]

Auto-assigned IP address
–

VPC ID
vpc-0cbe5544ff436e002

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
| Learn more



```
Apply complete! Resources: 3 added, 0 changed, 0 destroyed.

Outputs:

private-dns = "ip-172-31-44-33.ec2.internal"
private-ip = "172.31.44.33"
public-dns = "ec2-54-242-96-166.compute-1.amazonaws.com"
public-ip = "54.242.96.166"
```

Que 3 →

● Create AWS VPC with Terraform.

● Please follow the given link for more on AWS VPC creation.

1. Create a VPC

2. Create 2 Public Subnet & Create 2 Private Subnet

3. Create IGW (Internet Gateway) & Attach to the VPC

4. Create Public and Private Route Table

5. Add IGW in Public Route table (0.0.0.0/0) 6. Add Public Subnet (1a & 1b) in Route table

7. Create a NAT Gateway in Public Subnet

8. Add NAT GW into the Private Route Table

9. Add Private Subnet in Private Route Table

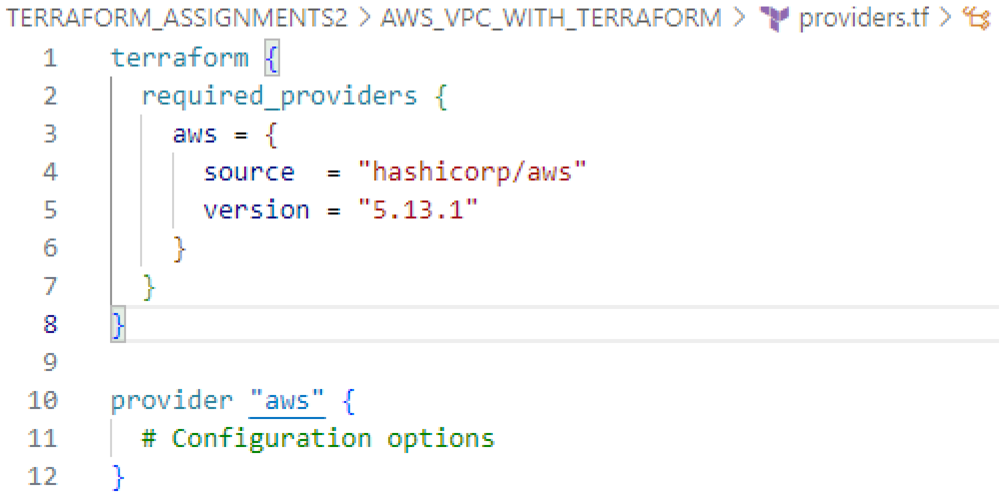
● Link: -

https://varunmanik1.medium.com/how-to-create-aws-vpc-in-10- steps-less-than-5-min-a49ac12064aa

● Note: -

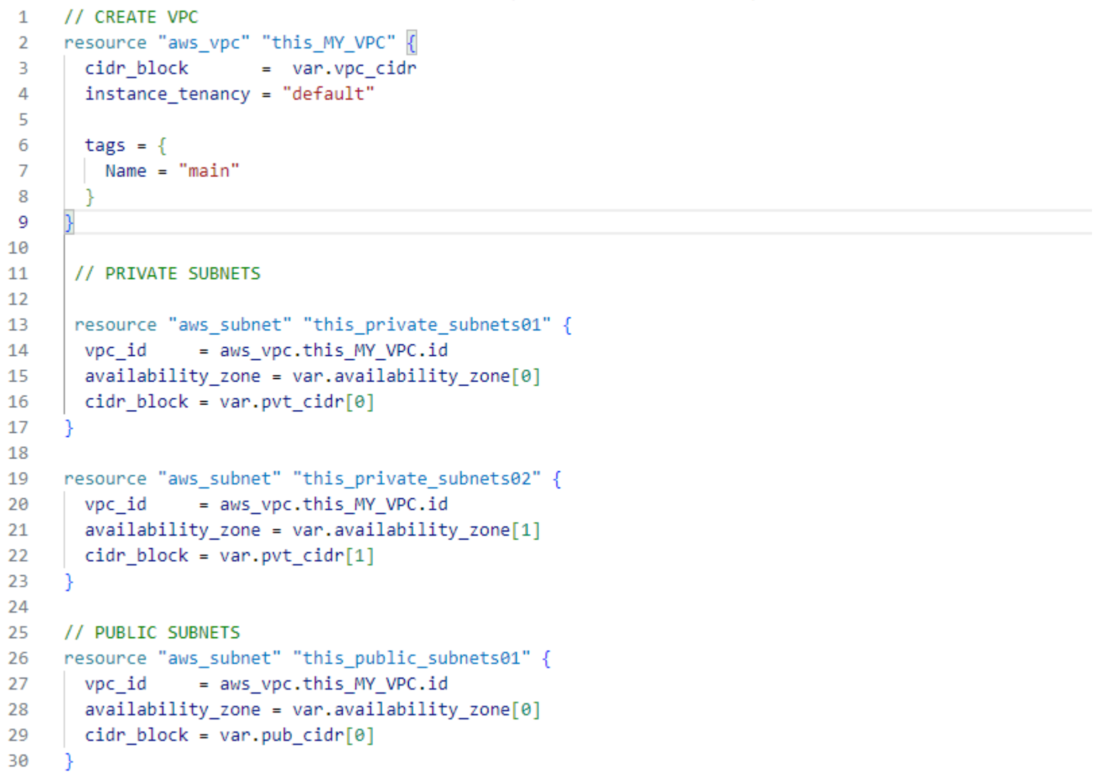
○ Try to create VPC manually to understand the concepts and then go for Terraform automation.

Providers.tf

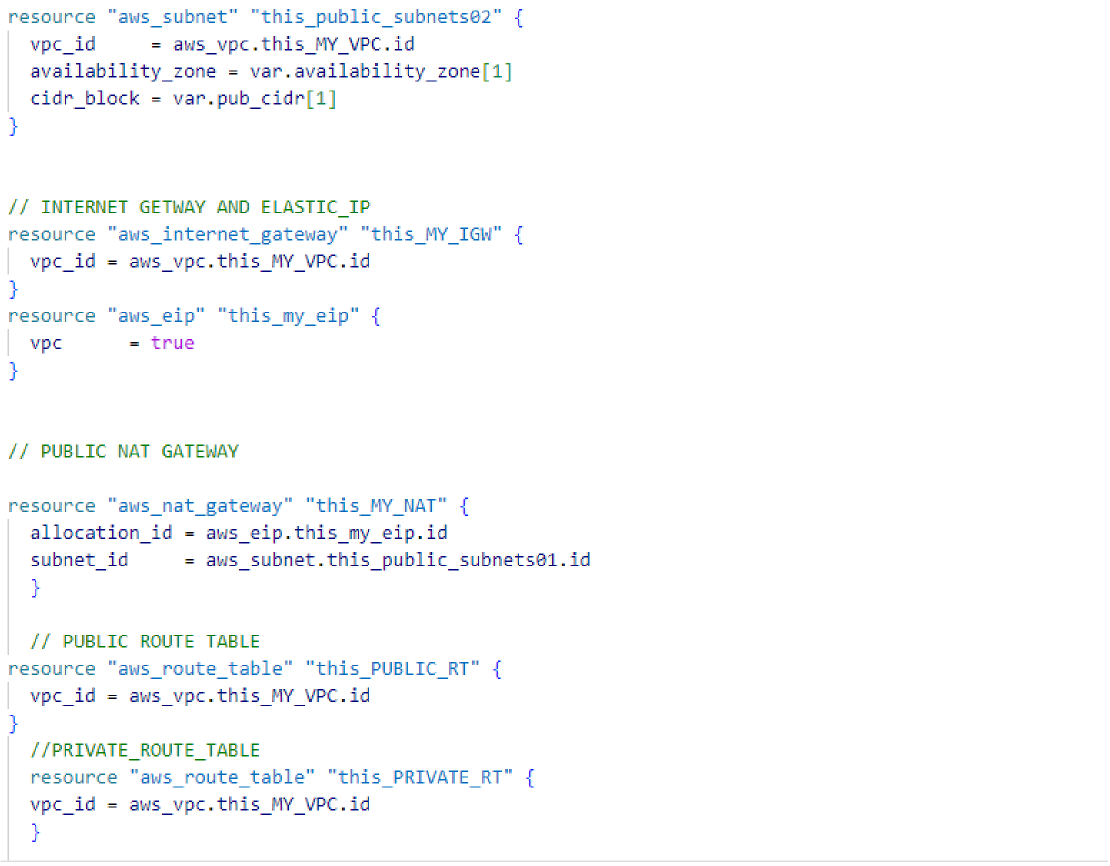

TERRAFORM_ASSIGNMENTS2 > AWS_VPC_WITH_TERRAFORM > providers.tf >

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "5.13.1"
    }
  }
}

provider "aws" {
  # Configuration options
}
```

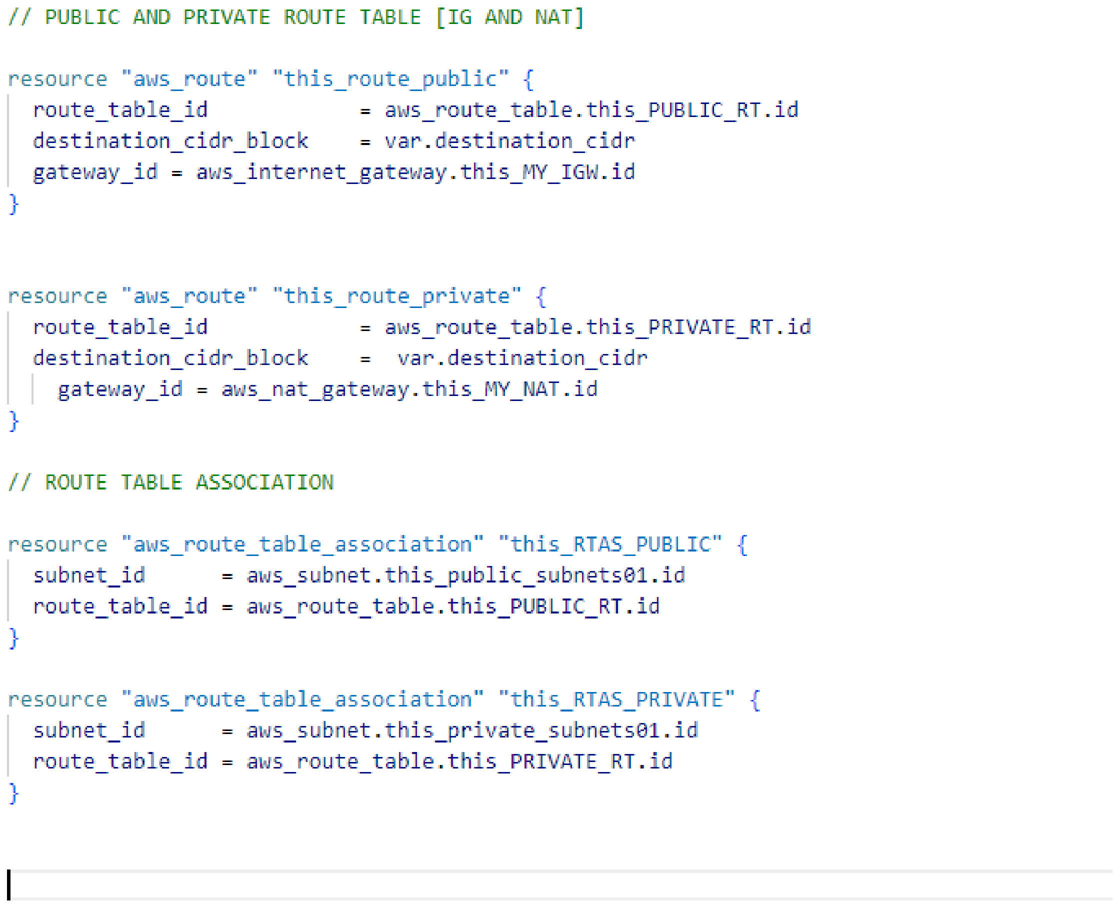
Main.tf



```
// CREATE VPC
resource "aws_vpc" "this_MY_VPC" {
  cidr_block       =  var.vpc_cidr
  instance_tenancy = "default"

  tags = {
    Name = "main"
  }
}

 // PRIVATE SUBNETS

 resource "aws_subnet" "this_private_subnets01" {
  vpc_id      = aws_vpc.this_MY_VPC.id
  availability_zone = var.availability_zone[0]
  cidr_block = var.pvt_cidr[0]
}

resource "aws_subnet" "this_private_subnets02" {
  vpc_id      = aws_vpc.this_MY_VPC.id
  availability_zone = var.availability_zone[1]
  cidr_block = var.pvt_cidr[1]
}

// PUBLIC SUBNETS
resource "aws_subnet" "this_public_subnets01" {
  vpc_id      = aws_vpc.this_MY_VPC.id
  availability_zone = var.availability_zone[0]
  cidr_block = var.pub_cidr[0]
}
```

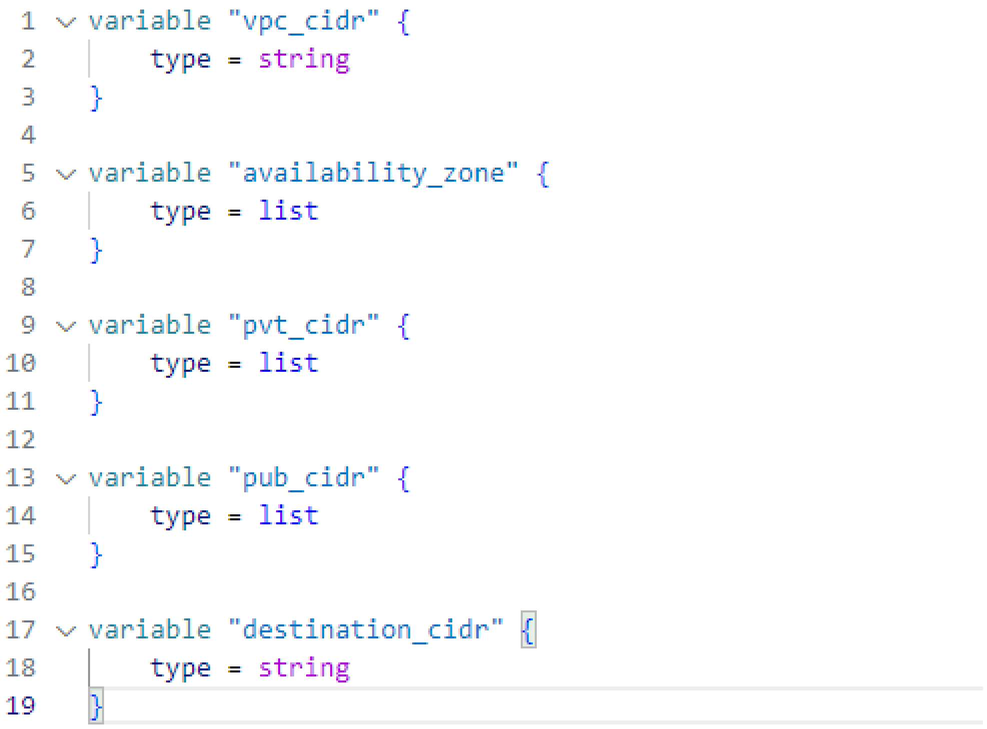


```
resource "aws_subnet" "this_public_subnets02" {
  vpc_id      = aws_vpc.this_MY_VPC.id
  availability_zone = var.availability_zone[1]
  cidr_block = var.pub_cidr[1]
}


// INTERNET GETWAY AND ELASTIC_IP
resource "aws_internet_gateway" "this_MY_IGW" {
  vpc_id = aws_vpc.this_MY_VPC.id
}
resource "aws_eip" "this_my_eip" {
  vpc      = true
}


// PUBLIC NAT GATEWAY

resource "aws_nat_gateway" "this_MY_NAT" {
  allocation_id = aws_eip.this_my_eip.id
  subnet_id     = aws_subnet.this_public_subnets01.id
  }

  // PUBLIC ROUTE TABLE
resource "aws_route_table" "this_PUBLIC_RT" {
  vpc_id = aws_vpc.this_MY_VPC.id
}
  //PRIVATE_ROUTE_TABLE
  resource "aws_route_table" "this_PRIVATE_RT" {
  vpc_id = aws_vpc.this_MY_VPC.id
  }
```



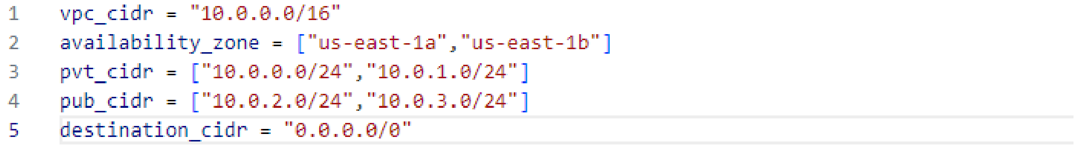
```
// PUBLIC AND PRIVATE ROUTE TABLE [IG AND NAT]

resource "aws_route" "this_route_public" {
  route_table_id            = aws_route_table.this_PUBLIC_RT.id
  destination_cidr_block    = var.destination_cidr
  gateway_id = aws_internet_gateway.this_MY_IGW.id
}


resource "aws_route" "this_route_private" {
  route_table_id            = aws_route_table.this_PRIVATE_RT.id
  destination_cidr_block    =  var.destination_cidr
    gateway_id = aws_nat_gateway.this_MY_NAT.id
}

// ROUTE TABLE ASSOCIATION

resource "aws_route_table_association" "this_RTAS_PUBLIC" {
  subnet_id      = aws_subnet.this_public_subnets01.id
  route_table_id = aws_route_table.this_PUBLIC_RT.id
}

resource "aws_route_table_association" "this_RTAS_PRIVATE" {
  subnet_id      = aws_subnet.this_private_subnets01.id
  route_table_id = aws_route_table.this_PRIVATE_RT.id
}
```

## Variables.tf



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

variable "availability_zone" {
    type = list
}

variable "pvt_cidr" {
    type = list
}

variable "pub_cidr" {
    type = list
}

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

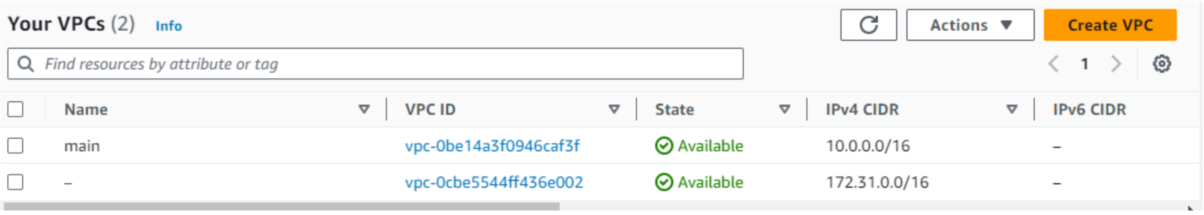
## Terraform.tfvars



```
vpc_cidr = "10.0.0.0/16"
availability_zone = ["us-east-1a","us-east-1b"]
pvt_cidr = ["10.0.0.0/24","10.0.1.0/24"]
pub_cidr = ["10.0.2.0/24","10.0.3.0/24"]
destination_cidr = "0.0.0.0/0"
```

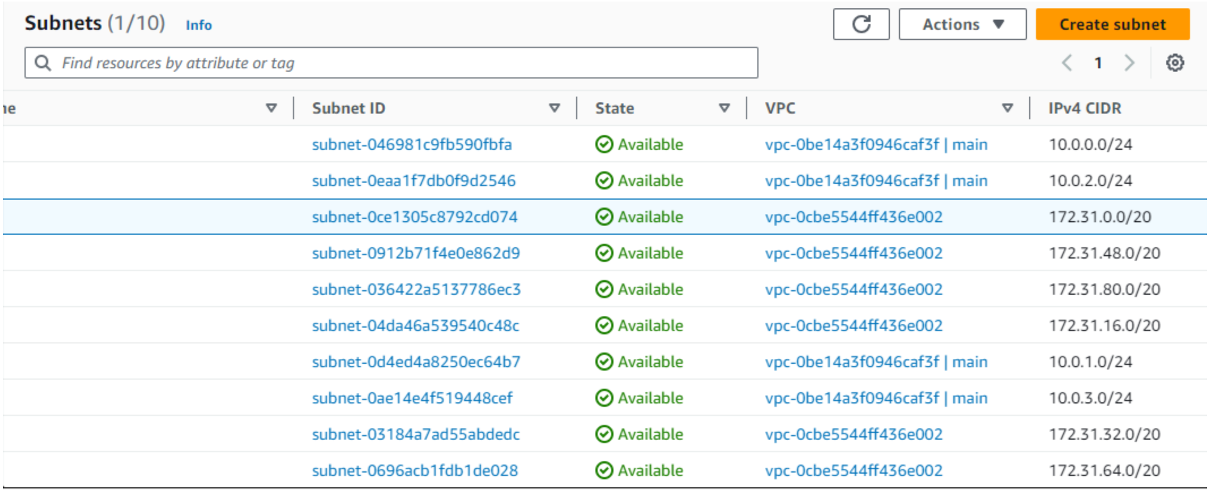
## Result

## VPC



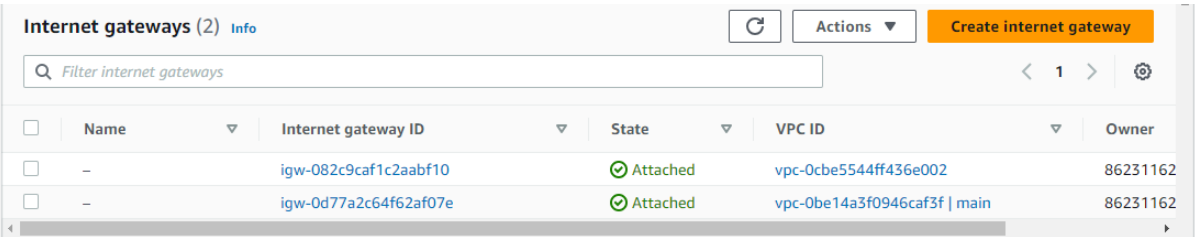
| □ | Name | VPC ID | State | IPv4 CIDR | IPv6 CIDR |
|---|---|---|---|---|---|
| □ | main | vpc-0be14a3f0946caf3f | ✓ Available | 10.0.0.0/16 | – |
| □ | – | vpc-0cbe5544ff436e002 | ✓ Available | 172.31.0.0/16 | – |

## Subnets



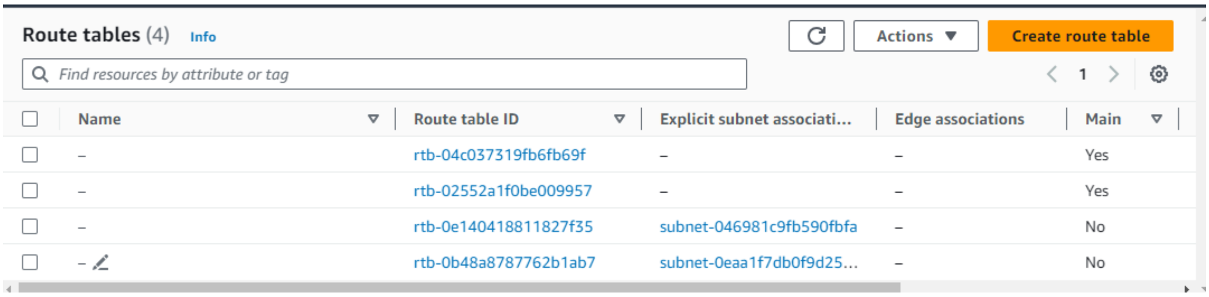
| ne | Subnet ID | State | VPC | IPv4 CIDR |
|---|---|---|---|---|
| | subnet-046981c9fb590fbfa | ✓ Available | vpc-0be14a3f0946caf3f \| main | 10.0.0.0/24 |
| | subnet-0eaa1f7db0f9d2546 | ✓ Available | vpc-0be14a3f0946caf3f \| main | 10.0.2.0/24 |
| | subnet-0ce1305c8792cd074 | ✓ Available | vpc-0cbe5544ff436e002 | 172.31.0.0/20 |
| | subnet-0912b71f4e0e862d9 | ✓ Available | vpc-0cbe5544ff436e002 | 172.31.48.0/20 |
| | subnet-036422a5137786ec3 | ✓ Available | vpc-0cbe5544ff436e002 | 172.31.80.0/20 |
| | subnet-04da46a539540c48c | ✓ Available | vpc-0cbe5544ff436e002 | 172.31.16.0/20 |
| | subnet-0d4ed4a8250ec64b7 | ✓ Available | vpc-0be14a3f0946caf3f \| main | 10.0.1.0/24 |
| | subnet-0ae14e4f519448cef | ✓ Available | vpc-0be14a3f0946caf3f \| main | 10.0.3.0/24 |
| | subnet-03184a7ad55abdedc | ✓ Available | vpc-0cbe5544ff436e002 | 172.31.32.0/20 |
| | subnet-0696acb1fdb1de028 | ✓ Available | vpc-0cbe5544ff436e002 | 172.31.64.0/20 |

## Internet Gateway



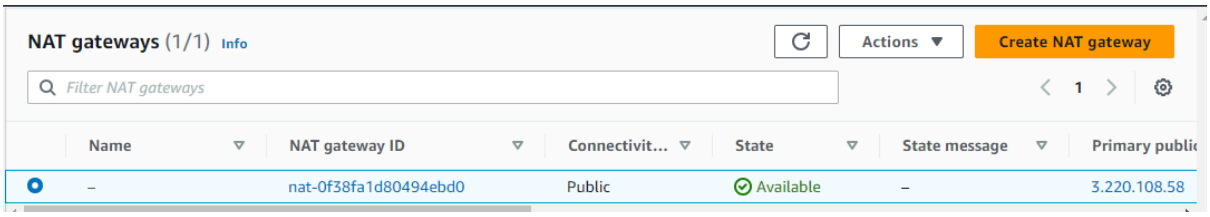
| □ | Name | Internet gateway ID | State | VPC ID | Owner |
|---|---|---|---|---|---|
| □ | – | igw-082c9caf1c2aabf10 | ✓ Attached | vpc-0cbe5544ff436e002 | 86231162 |
| □ | – | igw-0d77a2c64f62af07e | ✓ Attached | vpc-0be14a3f0946caf3f \| main | 86231162 |

## Route Table



| □ | Name | Route table ID | Explicit subnet associati... | Edge associations | Main |
|---|---|---|---|---|---|
| □ | – | rtb-04c037319fb6fb69f | – | – | Yes |
| □ | – | rtb-02552a1f0be009957 | – | – | Yes |
| □ | – | rtb-0e140418811827f35 | subnet-046981c9fb590fbfa | – | No |
| □ | – | rtb-0b48a8787762b1ab7 | subnet-0eaa1f7db0f9d25... | – | No |

## NAT gateway



| | Name | NAT gateway ID | Connectivit... | State | State message | Primary public |
|---|---|---|---|---|---|---|
| ● | – | nat-0f38fa1d80494ebd0 | Public | ✓ Available | – | 3.220.108.58 |

**Que 4 →**

● Create EC2 instance one of the public Subnets of VPC that you have created & Validate your Connection using ssh

● For this You need to create below AWS resources using Terraform.

1. EC2 Instance.

2. SSH Key

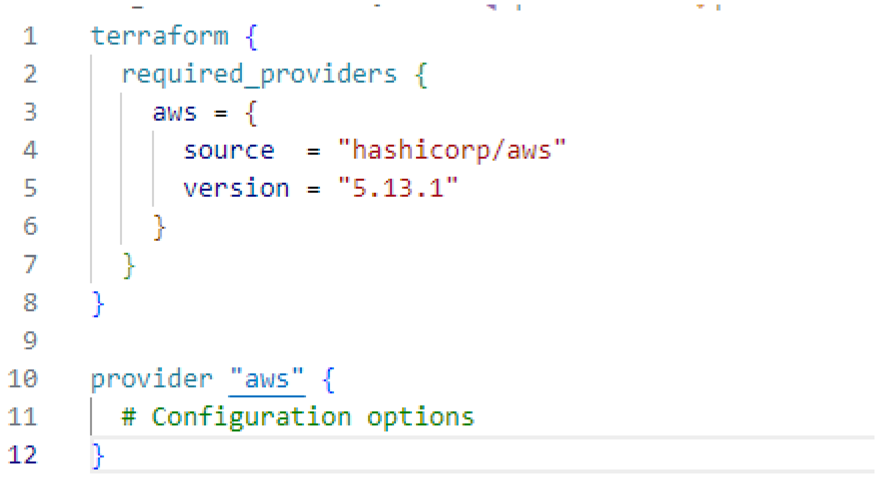
3. Security Group.

● Note: -

○ Attach SSH key and Security Group to EC2 Instance using attribute reference.

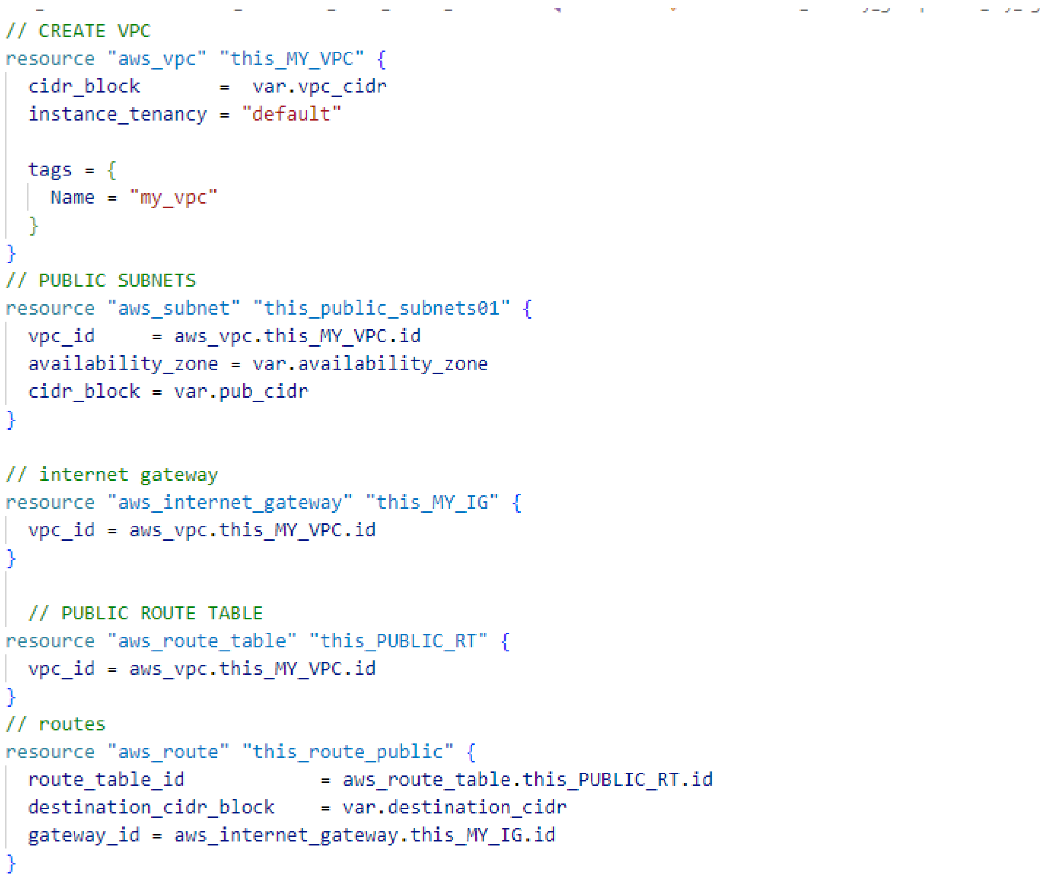
○ Then try to access it from an EC2 instance using the SSH key that you have created

Providers.tf

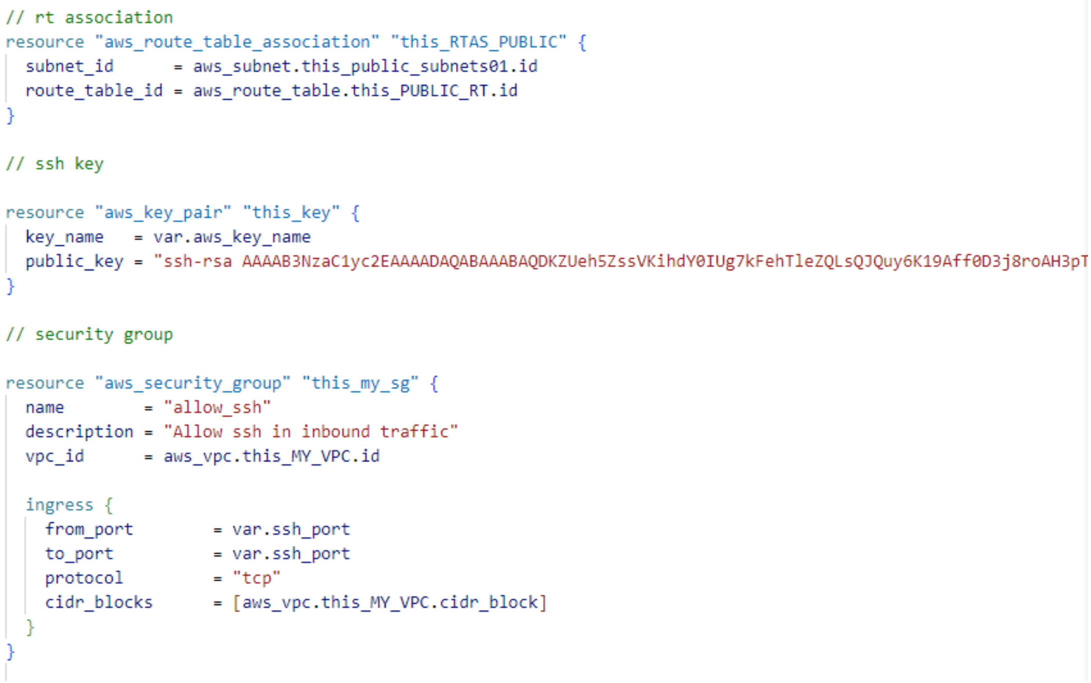


```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "5.13.1"
    }
  }
}

provider "aws" {
  # Configuration options
}
```

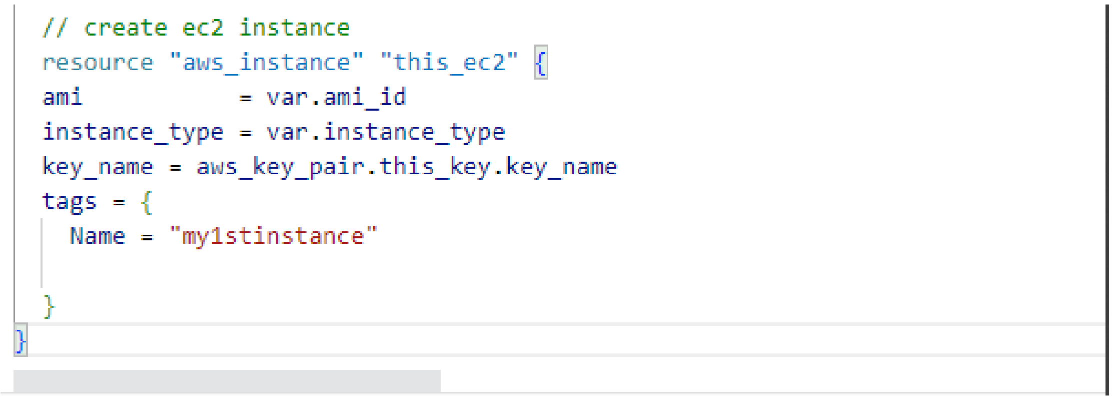
Main.tf



```
// CREATE VPC
resource "aws_vpc" "this_MY_VPC" {
  cidr_block       =  var.vpc_cidr
  instance_tenancy = "default"

  tags = {
    Name = "my_vpc"
  }
}
// PUBLIC SUBNETS
resource "aws_subnet" "this_public_subnets01" {
  vpc_id      = aws_vpc.this_MY_VPC.id
  availability_zone = var.availability_zone
  cidr_block = var.pub_cidr
}

// internet gateway
resource "aws_internet_gateway" "this_MY_IG" {
  vpc_id = aws_vpc.this_MY_VPC.id
}

  // PUBLIC ROUTE TABLE
resource "aws_route_table" "this_PUBLIC_RT" {
  vpc_id = aws_vpc.this_MY_VPC.id
}
// routes
resource "aws_route" "this_route_public" {
  route_table_id            = aws_route_table.this_PUBLIC_RT.id
  destination_cidr_block    = var.destination_cidr
  gateway_id = aws_internet_gateway.this_MY_IG.id
}
```

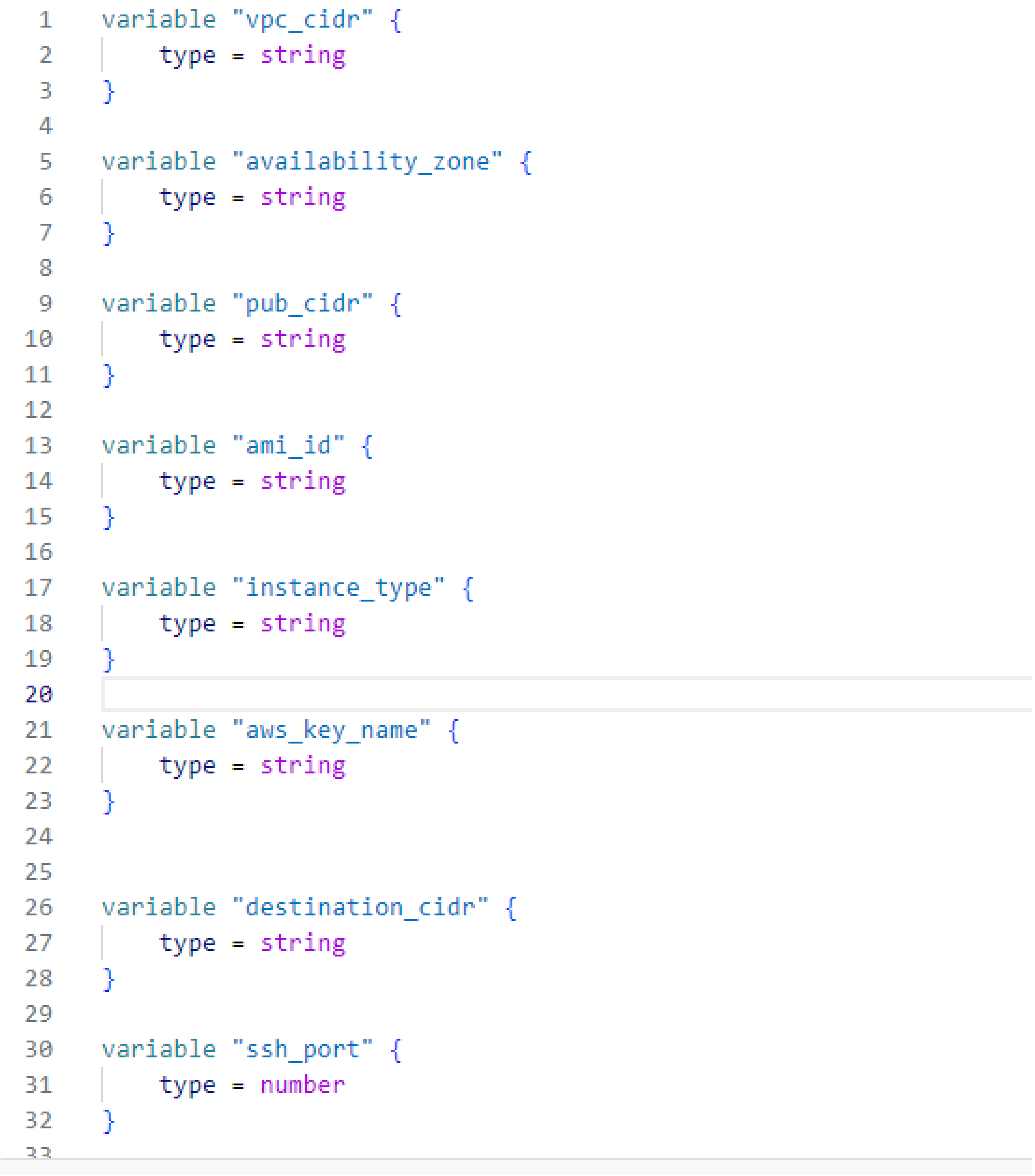


```
// rt association
resource "aws_route_table_association" "this_RTAS_PUBLIC" {
  subnet_id      = aws_subnet.this_public_subnets01.id
  route_table_id = aws_route_table.this_PUBLIC_RT.id
}

// ssh key

resource "aws_key_pair" "this_key" {
  key_name   = var.aws_key_name
  public_key = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDKZUeh5ZssVKihdY0IUg7kFehTleZQLsQJQuy6K19Aff0D3j8roAH3pT
}

// security group

resource "aws_security_group" "this_my_sg" {
  name        = "allow_ssh"
  description = "Allow ssh in inbound traffic"
  vpc_id      = aws_vpc.this_MY_VPC.id

  ingress {
    from_port       = var.ssh_port
    to_port         = var.ssh_port
    protocol        = "tcp"
    cidr_blocks     = [aws_vpc.this_MY_VPC.cidr_block]
  }
}
```



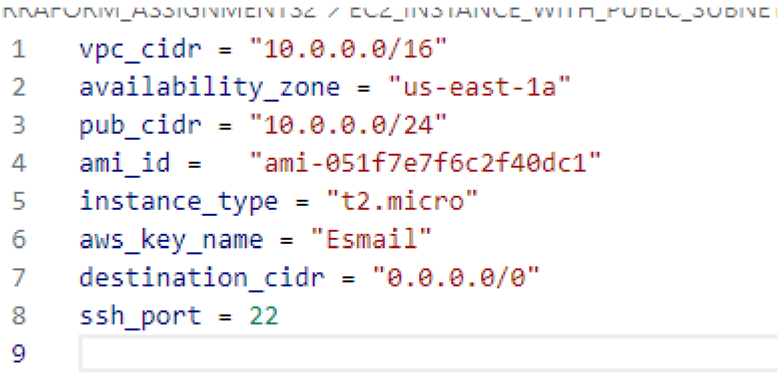
```
// create ec2 instance
resource "aws_instance" "this_ec2" {
ami           = var.ami_id
instance_type = var.instance_type
key_name = aws_key_pair.this_key.key_name
tags = {
  Name = "my1stinstance"

}
}
```

Variables.tf



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

variable "availability_zone" {
    type = string
}

variable "pub_cidr" {
    type = string
}

variable "ami_id" {
    type = string
}

variable "instance_type" {
    type = string
}

variable "aws_key_name" {
    type = string
}


variable "destination_cidr" {
    type = string
}

variable "ssh_port" {
    type = number
}
```

Terraform.tfvars

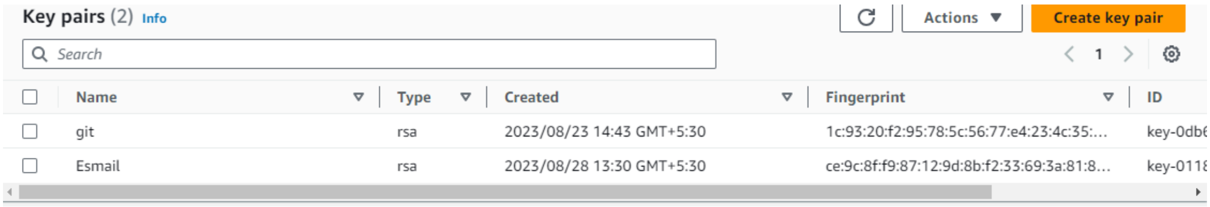


RRAFORM_ASSIGNMENT32 / EC2_INSTANCE_WITH_PUBLIC_SUBNET

```
vpc_cidr = "10.0.0.0/16"
availability_zone = "us-east-1a"
pub_cidr = "10.0.0.0/24"
ami_id =   "ami-051f7e7f6c2f40dc1"
instance_type = "t2.micro"
aws_key_name = "Esmail"
destination_cidr = "0.0.0.0/0"
ssh_port = 22

```

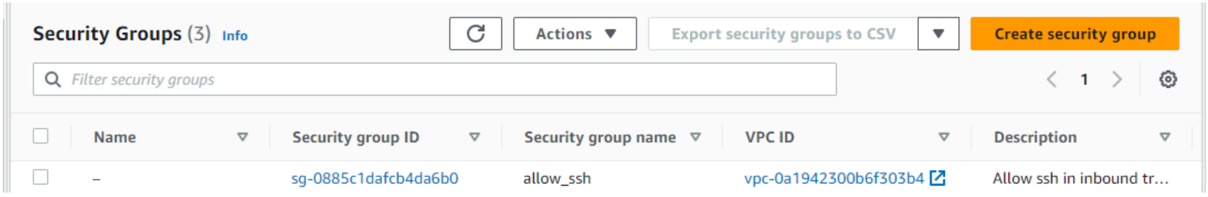
## Result

## KAY PAIR



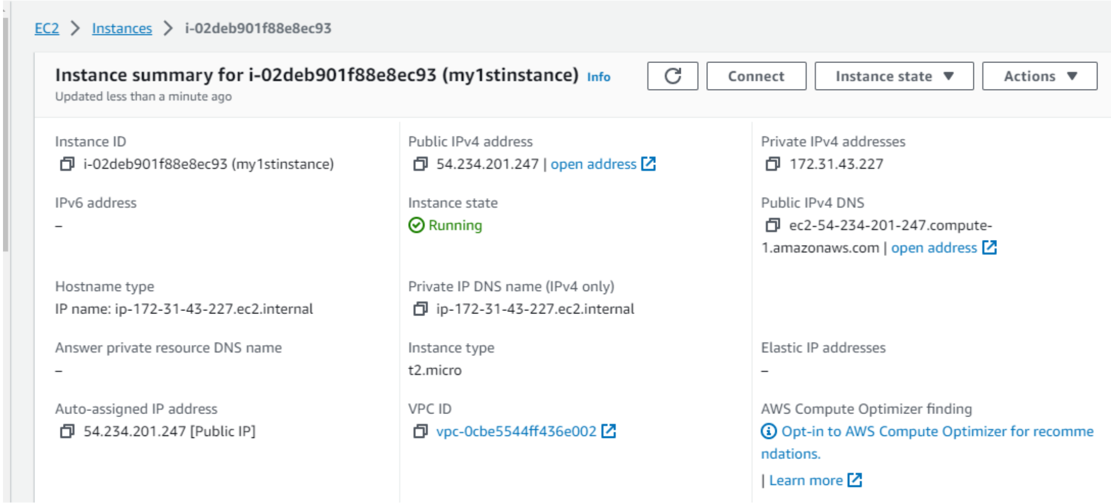
| □ | Name | Type | Created | Fingerprint | ID |
|---|---|---|---|---|---|
| □ | git | rsa | 2023/08/23 14:43 GMT+5:30 | 1c:93:20:f2:95:78:5c:56:77:e4:23:4c:35:... | key-0db6 |
| □ | Esmail | rsa | 2023/08/28 13:30 GMT+5:30 | ce:9c:8f:f9:87:12:9d:8b:f2:33:69:3a:81:8... | key-0118 |

## Security Group



| □ | Name | Security group ID | Security group name | VPC ID | Description |
|---|---|---|---|---|---|
| □ | – | sg-0885c1dafcb4da6b0 | allow_ssh | vpc-0a1942300b6f303b4 | Allow ssh in inbound tr... |

## Instance



Instance summary for i-02deb901f88e8ec93 (my1stinstance)

Updated less than a minute ago

Instance ID
i-02deb901f88e8ec93 (my1stinstance)

Public IPv4 address
54.234.201.247 | open address

Private IPv4 addresses
172.31.43.227

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-54-234-201-247.compute-1.amazonaws.com | open address

Hostname type
IP name: ip-172-31-43-227.ec2.internal

Private IP DNS name (IPv4 only)
ip-172-31-43-227.ec2.internal

Answer private resource DNS name
–

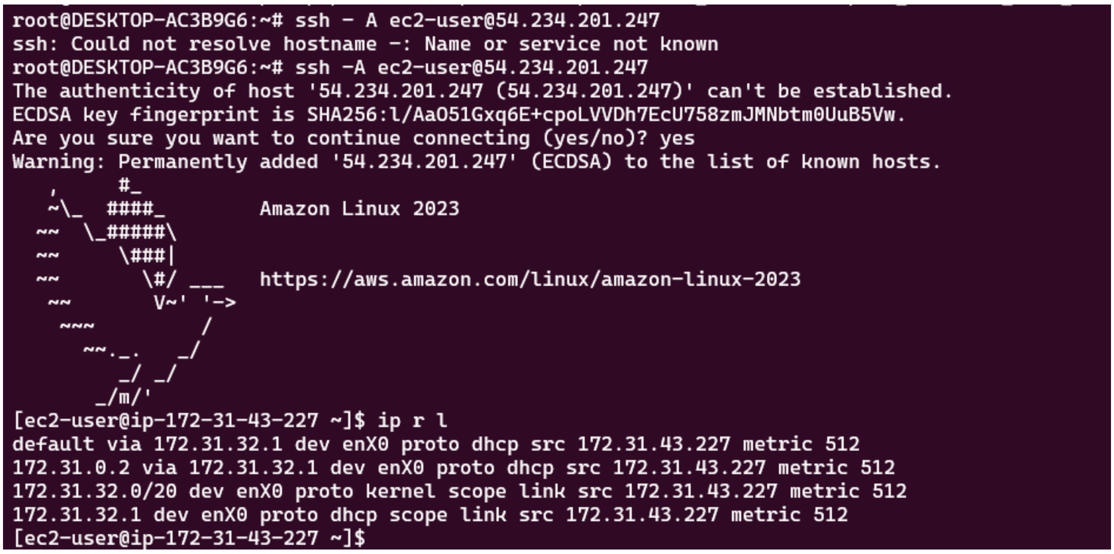
Instance type
t2.micro

Elastic IP addresses
–

Auto-assigned IP address
54.234.201.247 [Public IP]

VPC ID
vpc-0cbe5544ff436e002

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
| Learn more



```
root@DESKTOP-AC3B9G6:~# ssh - A ec2-user@54.234.201.247
ssh: Could not resolve hostname -: Name or service not known
root@DESKTOP-AC3B9G6:~# ssh -A ec2-user@54.234.201.247
The authenticity of host '54.234.201.247 (54.234.201.247)' can't be established.
ECDSA key fingerprint is SHA256:l/AaO51Gxq6E+cpoLVVDh7EcU758zmJMNbtm0UuB5Vw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.234.201.247' (ECDSA) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-43-227 ~]$ ip r l
default via 172.31.32.1 dev enX0 proto dhcp src 172.31.43.227 metric 512
172.31.0.2 via 172.31.32.1 dev enX0 proto dhcp src 172.31.43.227 metric 512
172.31.32.0/20 dev enX0 proto kernel scope link src 172.31.43.227 metric 512
172.31.32.1 dev enX0 proto dhcp scope link src 172.31.43.227 metric 512
[ec2-user@ip-172-31-43-227 ~]$
```