Terraform Assignment 02 on AWS - Haneef Shaikh

Que 1 →

➔ Create below resources using Terraform using count * count.index.
- 3 IAM Users (yourname_0 , yourname_1 , yourname_2)
- 3 IAM Groups (dev_0 , dev_1 , dev_2)

➔ Map these IAM users in IAM groups that you have created using **aws_iam_user_group_membership** Terraform resource.

➔ Note :-
- yourname_0 should be part of dev_0 group.
- yourname_1 should be part of dev_1 group.
- yourname_2 should be part of dev_2 group.

Main.tf

```
#AWS Provider
terraform {
 required_providers {
   aws = {
     source = "hashicorp/aws"
     version = "4.52.0"
   }
 }
}

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

IAM.tf

```

// IAM GROUP

resource "aws_iam_group" "application_group" {
 name = var.iam_group_name[count.index]
 path = var.iam_group_path
 count = 3
}

// IAM USER

resource "aws_iam_user" "application_users" {
 name = var.iam_user_name[count.index]
 path = var.iam_user_path
 count = 3
}

// IAM GROUP MEMBER

resource "aws_iam_user_group_membership" "application_group_members" {
 user = aws_iam_user.application_users[count.index].name
 groups = [aws_iam_group.application_group[count.index].name]
 count = 3
}
```

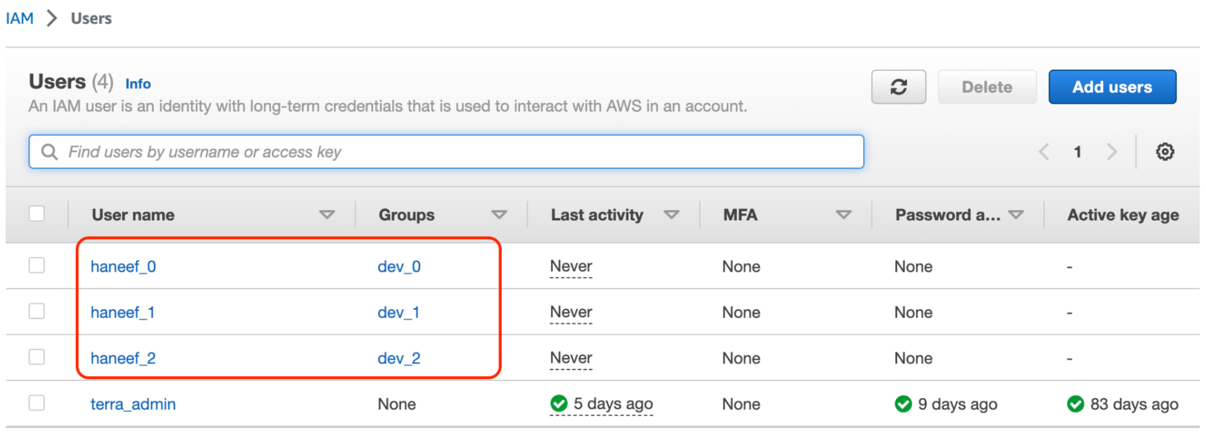
Variable.tf

```

// IAM GROUP

variable "iam_group_name" {
   type = list
}

variable "iam_group_path" {
   type = string
}

// IAM USER

variable "iam_user_name" {
   type = list
}

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

Terraform.tfvars

```
// IAM GROUP
iam_group_name = ["dev_0","dev_1","dev_2"]
iam_group_path = "/users/"

// IAM USER
iam_user_name = ["haneef_0","haneef_1","haneef_2"]
iam_user_path = "/system/"
```

OUTPUT:



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

User name	Groups	Last activity	MFA	Password a...	Active key age
haneef_0	dev_0	Never	None	None	-
haneef_1	dev_1	Never	None	None	-
haneef_2	dev_2	Never	None	None	-
terra_admin	None	✓ 5 days ago	None	✓ 9 days ago	✓ 83 days ago

Que 2 →

- ➔ Create below resources using Terraform based on conditions.
- ➔ Create a variable name ENV with any of these values (DEV/QA).

- ● If ENV is DEV then
  - ○ Create 2 EC2 instances.
  - ○ 1 Security Group and allow traffic from port 22,80,443.
  - ○ Map Security Group to EC2 Instance.
- ● If ENV is QA then.
  - ○ Create 1 EC2 instance.
  - ○ 1 Security Group and allow traffic from port 22,8080,3306.
  - ○ Map Security Group to EC2 Instance.

➔ Note :-

- ○ Use data sources to fetch the AMI ID for Dev and QA instances.
- ○ Dev instance should be created with filter
  "ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-*"
- ○ QA instance should be created with filter
  "ubuntu/images/hvm-ssd/ubuntu-bionic-18.04-amd64-server-"
- ○ Use Dynamic block for security group ingress & egress to add rules.
- ○ Also Make sure these resources must be created in your own VPC.

Main.tf

```
#AWS Provider
terraform {
 required_providers {
   aws = {
     source  = "hashicorp/aws"
     version = "4.52.0"
   }
 }
}

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

VPC.tf

```

locals {
 common_tags = {
   user = "vpc"
 }
}

#VPC
resource "aws_vpc" "cloudethix-vpc" {
 cidr_block       = var.vpc_cidr_block
 instance_tenancy = "default"
 tags             = local.common_tags
}

#private subnets
resource "aws_subnet" "cloudethix-sub-private01" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = var.private_subnet_cidr[0]
 availability_zone       = var.availability_zone[0]
 map_public_ip_on_launch = true
 tags                    = local.common_tags
}

resource "aws_subnet" "cloudethix-sub-private02" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = var.private_subnet_cidr[1]
 availability_zone       = var.availability_zone[1]
 map_public_ip_on_launch = true
 tags                    = local.common_tags
}


#public subnets
resource "aws_subnet" "cloudethix-sub-public01" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = var.public_subnet_cidr[0]
 availability_zone       = var.availability_zone[0]
 map_public_ip_on_launch = true
 tags                    = local.common_tags
}

resource "aws_subnet" "cloudethix-sub-public02" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = var.public_subnet_cidr[1]
 availability_zone       = var.availability_zone[1]
 map_public_ip_on_launch = true
 tags                    = local.common_tags
}


#Elastic IP
resource "aws_eip" "cloudethix-eip" {
 vpc  = true
 tags = local.common_tags
}


#IGW
resource "aws_internet_gateway" "cloudethix-igw" {
 vpc_id = aws_vpc.cloudethix-vpc.id
 tags   = local.common_tags

}


#Public NAT
resource "aws_nat_gateway" "cloudethix-nat" {
 allocation_id = aws_eip.cloudethix-eip.id
 subnet_id     = aws_subnet.cloudethix-sub-public01.id
 tags          = local.common_tags
}


#Route Table
resource "aws_route_table" "cloudethix-RT-public" {
 vpc_id = aws_vpc.cloudethix-vpc.id
 tags   = local.common_tags
}

resource "aws_route_table" "cloudethix-RT-private" {
 vpc_id = aws_vpc.cloudethix-vpc.id
 tags   = local.common_tags
}


#Route
```

```
resource "aws_route" "cloudethix-route-public" {
 route_table_id         = aws_route_table.cloudethix-RT-public.id
 destination_cidr_block = var.destination_cidr_block
 gateway_id             = aws_internet_gateway.cloudethix-igw.id
}

resource "aws_route" "cloudethix-route-private" {
 route_table_id         = aws_route_table.cloudethix-RT-private.id
 destination_cidr_block = var.destination_cidr_block
 gateway_id             = aws_nat_gateway.cloudethix-nat.id
}

#Route Table Association
resource "aws_route_table_association" "cloudethix-RTASS-public" {
 subnet_id      = aws_subnet.cloudethix-sub-public01.id
 route_table_id = aws_route_table.cloudethix-RT-public.id
}

resource "aws_route_table_association" "cloudethix-RTASS-private" {
 subnet_id      = aws_subnet.cloudethix-sub-private01.id
 route_table_id = aws_route_table.cloudethix-RT-private.id
}
```

Varibale.tf

```
// IAM USER
variable "iam_user_name" {
 type = list(any)
}

variable "iam_user_path" {
 type = string
}

//EC2
variable "ENV" {
 type = string
}

variable "dev_type" {
 type = string
}

variable "qa_type" {
 type = string
}

//SG
variable "dev_port" {
 type = list(any)
}

variable "qa_port" {
}

// VPC
variable "availability_zone" {
 type = list(any)
}

variable "vpc_cidr_block" {
 type = string
}

variable "public_subnet_cidr" {
 type = list(any)
}

variable "private_subnet_cidr" {
 type = list(any)
}

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

EC2.tf

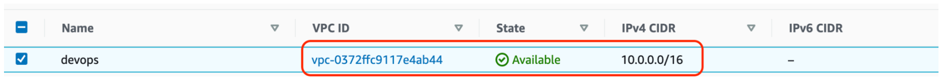
```
data "aws_ami" "dev_ami" {
 most_recent = true
 filter {
   name   = "name"
   values = ["ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-*"]
 }
 filter {
   name   = "virtualization-type"
   values = ["hvm"]
 }
 owners = ["099720109477"]
}

data "aws_ami" "qa_ami" {
 most_recent = true
 filter {
   name   = "name"
   values = ["ubuntu/images/hvm-ssd/ubuntu-bionic-18.04-amd64-server-*"]
 }
 filter {
   name   = "virtualization-type"
   values = ["hvm"]
 }
 owners = ["099720109477"]
}

resource "aws_instance" "ec2_dev" {
 ami                    = data.aws_ami.dev_ami.id
 instance_type          = var.dev_type
 vpc_security_group_ids = [aws_security_group.dev_sg.id]
 subnet_id              = aws_subnet.cloudethix-sub-private01.id
 count                  = var.ENV == "DEV" ? 2 : 0
}

resource "aws_instance" "ec2_qa" {
 ami                    = data.aws_ami.qa_ami.id
 instance_type          = var.qa_type
 vpc_security_group_ids = [aws_security_group.qa_sg.id]
 subnet_id              = aws_subnet.cloudethix-sub-private02.id
 count                  = var.ENV == "QA" ? 1 : 0
}
```

Sg.tf

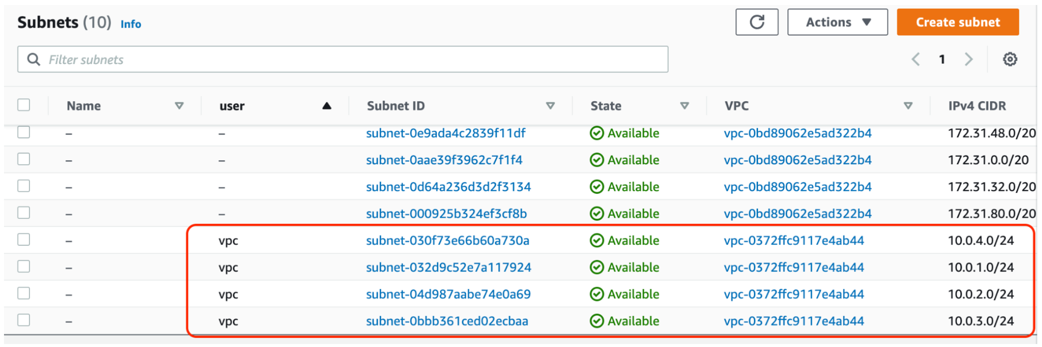
```
# Security Group

resource "aws_security_group" "dev_sg" {
 name        = "dev_sg"
 description = "Allow dev inbound traffic"
 vpc_id      = aws_vpc.cloudethix-vpc.id

 dynamic "ingress" {
   for_each = var.dev_port
   iterator = port
   content {
     from_port   = port.value
     to_port     = port.value
     protocol    = "tcp"
     cidr_blocks = ["0.0.0.0/0"]
   }
 }

 dynamic "egress" {
   for_each = var.dev_port
   content {
     from_port   = egress.value
     to_port     = egress.value
     protocol    = "tcp"
     cidr_blocks = ["0.0.0.0/0"]
   }
 }

 tags = {
   Name = "dynamic"
 }
}



resource "aws_security_group" "qa_sg" {
 name        = "qa_sg"
 description = "Allow QA inbound traffic"
 vpc_id      = aws_vpc.cloudethix-vpc.id

 dynamic "ingress" {
   for_each = var.qa_port
   iterator = port
   content {
     from_port   = port.value
     to_port     = port.value
     protocol    = "tcp"
     cidr_blocks = ["0.0.0.0/0"]
   }
 }

 dynamic "egress" {
   for_each = var.qa_port
   content {
     from_port   = egress.value
     to_port     = egress.value
     protocol    = "tcp"
     cidr_blocks = ["0.0.0.0/0"]
   }
 }

 tags = {
   Name = "dynamic"
 }
}
```

Terraform.tfvars

```

//EC2
ENV      = "DEV"
dev_type = "t2.micro"
qa_type  = "t2.small"

//SG
dev_port = [22, 80, 443]
qa_port  = [22, 8080, 3306]

// VPC
availability_zone   = ["us-east-1a", "us-east-1b"]
vpc_cidr_block      = "10.0.0.0/16"
public_subnet_cidr  = ["10.0.1.0/24", "10.0.2.0/24"]
private_subnet_cidr = ["10.0.3.0/24", "10.0.4.0/24"]
destination_cidr_block = "0.0.0.0/0"
```

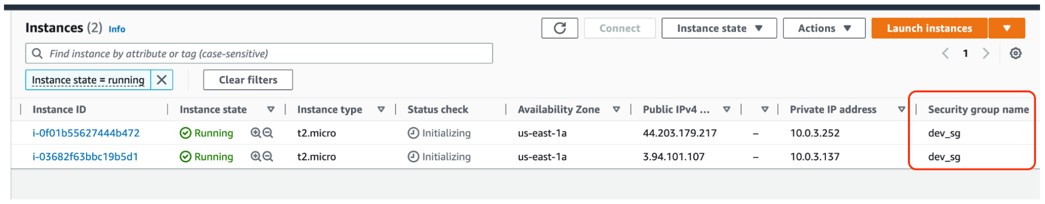


Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR
devops	vpc-0372ffc9117e4ab44	Available	10.0.0.0/16	–



Subnets (10) Info

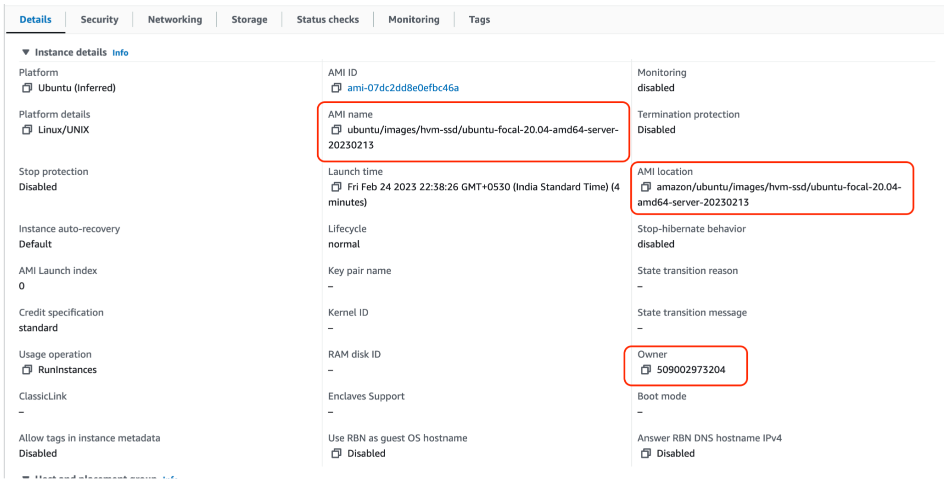
Name	user	Subnet ID	State	VPC	IPv4 CIDR
–	–	subnet-0e9ada4c2839f11df	Available	vpc-0bd89062e5ad322b4	172.31.48.0/20
–	–	subnet-0aae39f3962c7f1f4	Available	vpc-0bd89062e5ad322b4	172.31.0.0/20
–	–	subnet-0d64a236d3d2f3134	Available	vpc-0bd89062e5ad322b4	172.31.32.0/20
–	–	subnet-000925b324ef3cf8b	Available	vpc-0bd89062e5ad322b4	172.31.80.0/20
–	vpc	subnet-030f73e66b60a730a	Available	vpc-0372ffc9117e4ab44	10.0.4.0/24
–	vpc	subnet-032d9c52e7a117924	Available	vpc-0372ffc9117e4ab44	10.0.1.0/24
–	vpc	subnet-04d987aabe74e0a69	Available	vpc-0372ffc9117e4ab44	10.0.2.0/24
–	vpc	subnet-0bbb361ced02ecbaa	Available	vpc-0372ffc9117e4ab44	10.0.3.0/24



Instances (2) Info

Instance state = running

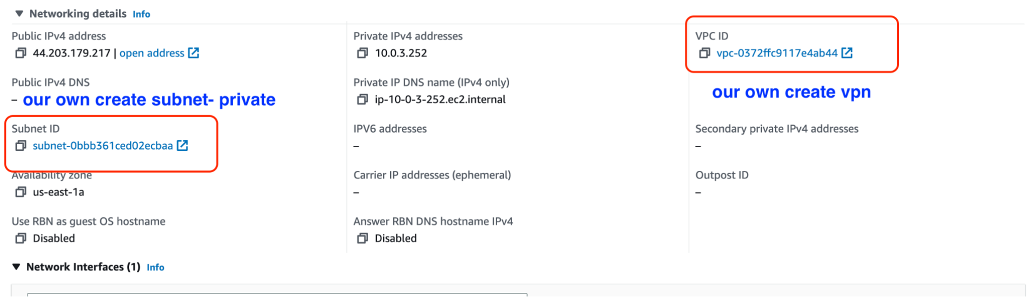
Instance ID	Instance state	Instance type	Status check	Availability Zone	Public IPv4 ...		Private IP address	Security group name
i-0f01b55627444b472	Running	t2.micro	Initializing	us-east-1a	44.203.179.217	–	10.0.3.252	dev_sg
i-03682f63bbc19b5d1	Running	t2.micro	Initializing	us-east-1a	3.94.101.107	–	10.0.3.137	dev_sg



Details | Security | Networking | Storage | Status checks | Monitoring | Tags

Instance details Info

Platform: Ubuntu (Inferred)	AMI ID: ami-07dc2dd8e0efbc46a	Monitoring: disabled
Platform details: Linux/UNIX	AMI name: ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-20230213	Termination protection: Disabled
Stop protection: Disabled	Launch time: Fri Feb 24 2023 22:38:26 GMT+0530 (India Standard Time) (4 minutes)	AMI location: amazon/ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-20230213
Instance auto-recovery: Default	Lifecycle: normal	Stop-hibernate behavior: disabled
AMI Launch index: 0	Key pair name: –	State transition reason: –
Credit specification: standard	Kernel ID: –	State transition message: –
Usage operation: RunInstances	RAM disk ID: –	Owner: 509002973204
ClassicLink: –	Enclaves Support: –	Boot mode: –
Allow tags in instance metadata: Disabled	Use RBN as guest OS hostname: Disabled	Answer RBN DNS hostname IPv4: Disabled



Networking details Info

Public IPv4 address: 44.203.179.217 | open address	Private IPv4 addresses: 10.0.3.252	VPC ID: vpc-0372ffc9117e4ab44
Public IPv4 DNS: –	Private IP DNS name (IPv4 only): ip-10-0-3-252.ec2.internal	
Subnet ID: subnet-0bbb361ced02ecbaa	IPV6 addresses: –	Secondary private IPv4 addresses: –
Availability zone: us-east-1a	Carrier IP addresses (ephemeral): –	Outpost ID: –
Use RBN as guest OS hostname: Disabled	Answer RBN DNS hostname IPv4: Disabled	

our own create subnet- private

our own create vpn

Network Interfaces (1) Info

Que 3 →

➔ Create below resources using Terraform.

● Create VPC with 2 Public and 2 Private subnets including IGW / NATGW / RT & Subnet association.

● Create 1 Security Group named alb-sg. Allow traffic on port 80 & 443 from 0.0.0.0/0

● Create 1 Application Load Balancer in the public Subnet of your VPC & attach alb-sg security group to ALB.

● Create 1 Security Group named web-sg. Allow traffic on port 80 from 0.0.0.0/0

● Create 1 EC2 instance named web-ec2 in public Subnet & attach web-sg security group to EC2 instance.

➔ Note :-
  - ○ Use data sources to fetch the AMI ID.
  - ○ Use Dynamic block for security group ingress & egress to add rules.
  - ○ Generate ssh key using aws_key_pair resource & map to EC2 instances.

Main.tf

```
#AWS Provider
terraform {
 required_providers {
   aws = {
     source  = "hashicorp/aws"
     version = "4.52.0"
   }
 }
}

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

Vpc.tf

```

locals {
 common_tags = {
   user = "vpc"
 }
}

#VPC
resource "aws_vpc" "cloudethix-vpc" {
 cidr_block       = var.vpc_cidr_block
 instance_tenancy = "default"
 tags             = local.common_tags
}

#private subnets
resource "aws_subnet" "cloudethix-sub-private01" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = var.private_subnet_cidr[0]
 availability_zone       = var.availability_zone[0]
 map_public_ip_on_launch = true
 tags                    = local.common_tags
}

resource "aws_subnet" "cloudethix-sub-private02" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = var.private_subnet_cidr[1]
 availability_zone       = var.availability_zone[1]
 map_public_ip_on_launch = true
 tags                    = local.common_tags
}

#public subnets
resource "aws_subnet" "cloudethix-sub-public01" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = var.public_subnet_cidr[0]
 availability_zone       = var.availability_zone[0]
 map_public_ip_on_launch = true
 tags                    = local.common_tags
}

resource "aws_subnet" "cloudethix-sub-public02" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = var.public_subnet_cidr[1]
 availability_zone       = var.availability_zone[1]
 map_public_ip_on_launch = true
 tags                    = local.common_tags
}

#Elastic IP
resource "aws_eip" "cloudethix-eip" {
 vpc  = true
 tags = local.common_tags
}

#IGW
resource "aws_internet_gateway" "cloudethix-igw" {
 vpc_id = aws_vpc.cloudethix-vpc.id
 tags   = local.common_tags
}

#Public NAT
resource "aws_nat_gateway" "cloudethix-nat" {
 allocation_id = aws_eip.cloudethix-eip.id
 subnet_id     = aws_subnet.cloudethix-sub-public01.id
 tags          = local.common_tags
}

#Route Table
resource "aws_route_table" "cloudethix-RT-public" {
 vpc_id = aws_vpc.cloudethix-vpc.id
 tags   = local.common_tags
}

resource "aws_route_table" "cloudethix-RT-private" {
 vpc_id = aws_vpc.cloudethix-vpc.id
 tags   = local.common_tags
}

#Route
resource "aws_route" "cloudethix-route-public" {
 route_table_id         = aws_route_table.cloudethix-RT-public.id
 destination_cidr_block = var.destination_cidr_block
 gateway_id             = aws_internet_gateway.cloudethix-igw.id
}

resource "aws_route" "cloudethix-route-private" {
 route_table_id         = aws_route_table.cloudethix-RT-private.id
 destination_cidr_block = var.destination_cidr_block
 gateway_id             = aws_nat_gateway.cloudethix-nat.id
}

#Route Table Association
resource "aws_route_table_association" "cloudethix-RTASS-public" {
 subnet_id      = aws_subnet.cloudethix-sub-public01.id
 route_table_id = aws_route_table.cloudethix-RT-public.id
}

resource "aws_route_table_association" "cloudethix-RTASS-private" {
 subnet_id      = aws_subnet.cloudethix-sub-private01.id
 route_table_id = aws_route_table.cloudethix-RT-private.id
}
```

EC2.tf

```
data "aws_ami" "ec2_ami" {
 most_recent = true
 filter {
   name   = "name"
   values = ["ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-*"]
 }
 filter {
   name   = "virtualization-type"
   values = ["hvm"]
 }
 owners = ["099720109477"]
}

resource "aws_instance" "web_ec2" {
 ami                    = data.aws_ami.ec2_ami.id
 instance_type          = var.ec2_instance_type
 key_name               = aws_key_pair.cloudethix-key-pair.key_name
 vpc_security_group_ids = [aws_security_group.web_sg.id]
 subnet_id              = aws_subnet.cloudethix-sub-public01.id

}
```

Sg.tf

```
# Security Group

resource "aws_security_group" "web_sg" {
 name        = "web"
 description = "Allow web inbound traffic"
 vpc_id      = aws_vpc.cloudethix-vpc.id

 dynamic "ingress" {
   for_each = var.web_sg_port
   iterator = port
   content {
     from_port   = port.value
     to_port     = port.value
     protocol    = "tcp"
     cidr_blocks = ["0.0.0.0/0"]
   }
 }

 dynamic "egress" {
   for_each = var.web_sg_port
   content {
     from_port   = egress.value
     to_port     = egress.value
     protocol    = "tcp"
     cidr_blocks = ["0.0.0.0/0"]
   }
 }

 tags = {
   name = "dynamic-WEB"
 }
}



#Load Balancer Security Group

resource "aws_security_group" "alb_sg" {
 name        = "allow_lb"
 description = "Allow lb inbound traffic"
 vpc_id       = aws_vpc.cloudethix-vpc.id

 dynamic "ingress" {
   for_each = var.alb_sg_port
   iterator = port
   content {
     from_port   = port.value
     to_port     = port.value
     protocol    = "tcp"
     cidr_blocks = ["0.0.0.0/0"]
   }
 }

 dynamic "egress" {
   for_each = var.alb_sg_port
   content {
     from_port   = egress.value
     to_port     = egress.value
     protocol    = "tcp"
     cidr_blocks = ["0.0.0.0/0"]
   }
 }

 tags = {
   Name = "dynamic-ALB"
 }
}
```

ALB.TF

```
#Load Balancer for WEB
resource "aws_lb" "cloudethix-lb" {
 name                       = "application-lb"
 internal                   = false
 load_balancer_type         = "application"
 security_groups            = [aws_security_group.alb_sg.id]
 subnets                    = [aws_subnet.cloudethix-sub-public01.id,
aws_subnet.cloudethix-sub-public02.id]
 enable_deletion_protection = true
 tags = {
   Environment = "ALB"
 }
}
```

Ssh_key.tf

```
#Key Pair to Access EC2
resource "aws_key_pair" "cloudethix-key-pair" {
 key_name    = var.ssh_key_name
 public_key = "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDkc/q0xTIzecyMPE/sjWmR9g8sP8/Xj7itL9kXRzHtYLT3T13E2OAfVC
t4zZ/eQIoTJuQWstL+slKG9anXKkrwKf4qF/2wxsZZ8Z9hUYV21KIGz9lDgmkueB3MKi07VyFhpBO1S2inbpjl
lkp0hp1AcYVOS0ulMhCC+X4y8yE5amG53/qiSLPnF0dBCa9icku0YYj6RZrjKfeL2S8uwBIMeTnPbpxn8BxkKI
djRErZjfuxASH39SYmWa7lpW3m2VReFc7t23ZjlEKFOaZWbwSK88L0EduRPV7+JbJDyCO/UxA+8E5/oJ9j9rt8
/MmE1YV5Nnf8UiHrGhH3WJkMBDZN"
}
```

Variable.tf

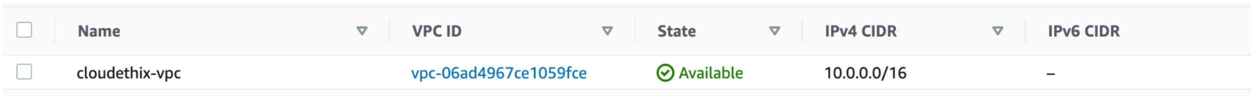
```
// SSH KEY PAIR
variable "ssh_key_name" {
 type = string
}

//EC2
variable "ec2_instance_type" {
 type = string
}

//SG
variable "alb_sg_port" {
 type = list(any)
}
variable "web_sg_port" {
 type = list(any)
}

// VPC
variable "availability_zone" {
 type = list(any)
}
```

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

variable "public_subnet_cidr" {
 type = list(any)
}

variable "private_subnet_cidr" {
 type = list(any)
}

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

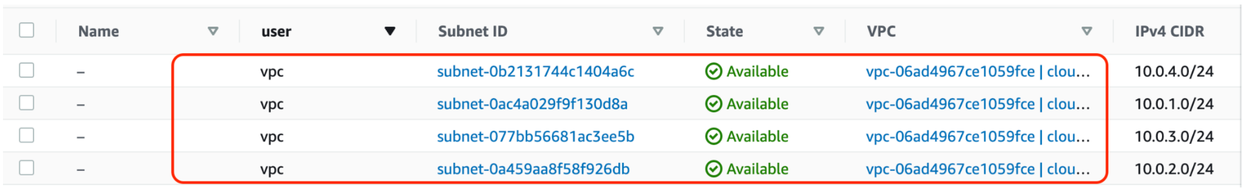
Terraform.tfvars

```
// SSH KEY PAIR
ssh_key_name = "cloud-ssh-key"

//EC2
ec2_instance_type = "t2.micro"

//SG
alb_sg_port = [80, 443]
web_sg_port = [22, 80, 443]



// VPC
availability_zone      = ["us-east-1a", "us-east-1b"]
vpc_cidr_block         = "10.0.0.0/16"
public_subnet_cidr     = ["10.0.1.0/24", "10.0.2.0/24"]
private_subnet_cidr    = ["10.0.3.0/24", "10.0.4.0/24"]
destination_cidr_block = "0.0.0.0/0"
```

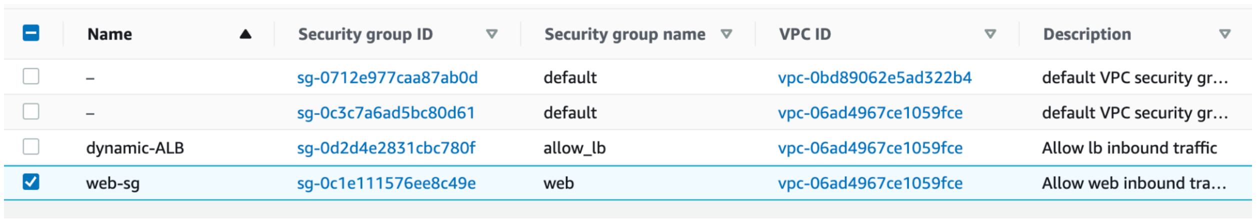
Output



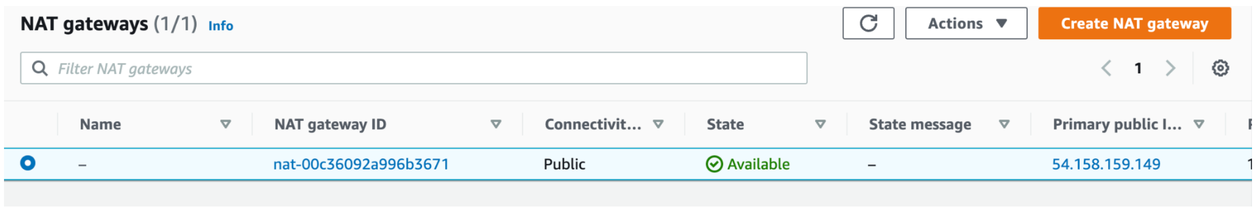
Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR
cloudethix-vpc	vpc-06ad4967ce1059fce	✓ Available	10.0.0.0/16	–



Name	user	Subnet ID	State	VPC	IPv4 CIDR
–	vpc	subnet-0b2131744c1404a6c	✓ Available	vpc-06ad4967ce1059fce | clou…	10.0.4.0/24
–	vpc	subnet-0ac4a029f9f130d8a	✓ Available	vpc-06ad4967ce1059fce | clou…	10.0.1.0/24
–	vpc	subnet-077bb56681ac3ee5b	✓ Available	vpc-06ad4967ce1059fce | clou…	10.0.3.0/24
–	vpc	subnet-0a459aa8f58f926db	✓ Available	vpc-06ad4967ce1059fce | clou…	10.0.2.0/24

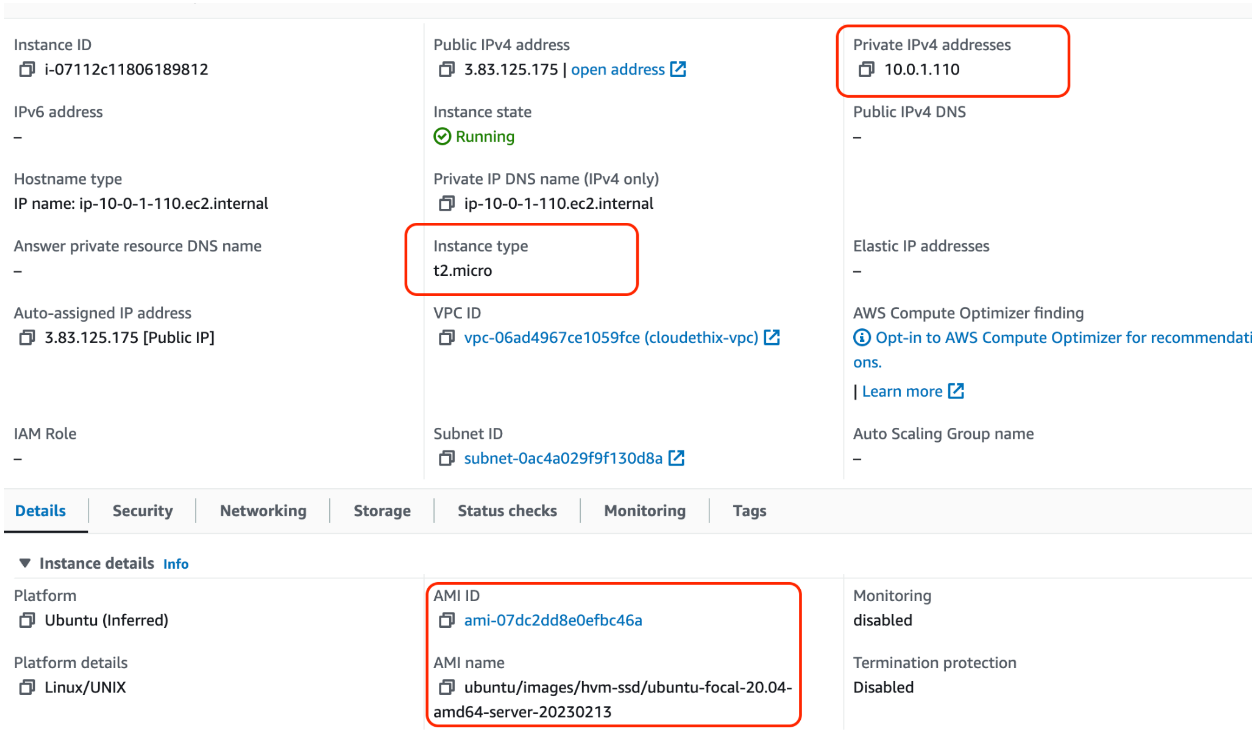


Name	Security group ID	Security group name	VPC ID	Description
–	sg-0712e977caa87ab0d	default	vpc-0bd89062e5ad322b4	default VPC security gr…
–	sg-0c3c7a6ad5bc80d61	default	vpc-06ad4967ce1059fce	default VPC security gr…
dynamic-ALB	sg-0d2d4e2831cbc780f	allow_lb	vpc-06ad4967ce1059fce	Allow lb inbound traffic
web-sg	sg-0c1e111576ee8c49e	web	vpc-06ad4967ce1059fce	Allow web inbound tra…



NAT gateways (1/1) Info

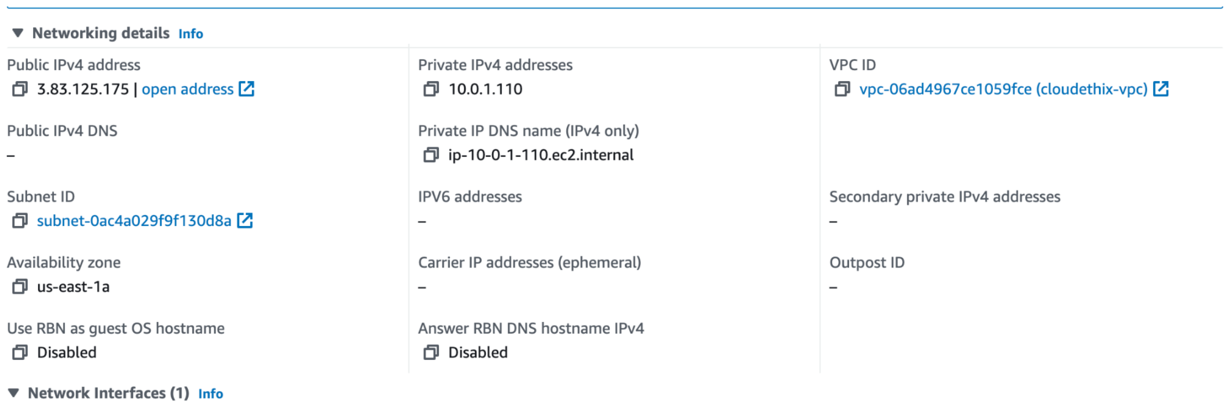
Name	NAT gateway ID	Connectivit…	State	State message	Primary public I…
–	nat-00c36092a996b3671	Public	✓ Available	–	54.158.159.149



Instance ID i-07112c11806189812	Public IPv4 address 3.83.125.175 | open address	Private IPv4 addresses 10.0.1.110
IPv6 address –	Instance state ✓ Running	Public IPv4 DNS –
Hostname type IP name: ip-10-0-1-110.ec2.internal	Private IP DNS name (IPv4 only) ip-10-0-1-110.ec2.internal	
Answer private resource DNS name –	Instance type t2.micro	Elastic IP addresses –
Auto-assigned IP address 3.83.125.175 [Public IP]	VPC ID vpc-06ad4967ce1059fce (cloudethix-vpc)	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role –	Subnet ID subnet-0ac4a029f9f130d8a	Auto Scaling Group name –

▼ Instance details Info

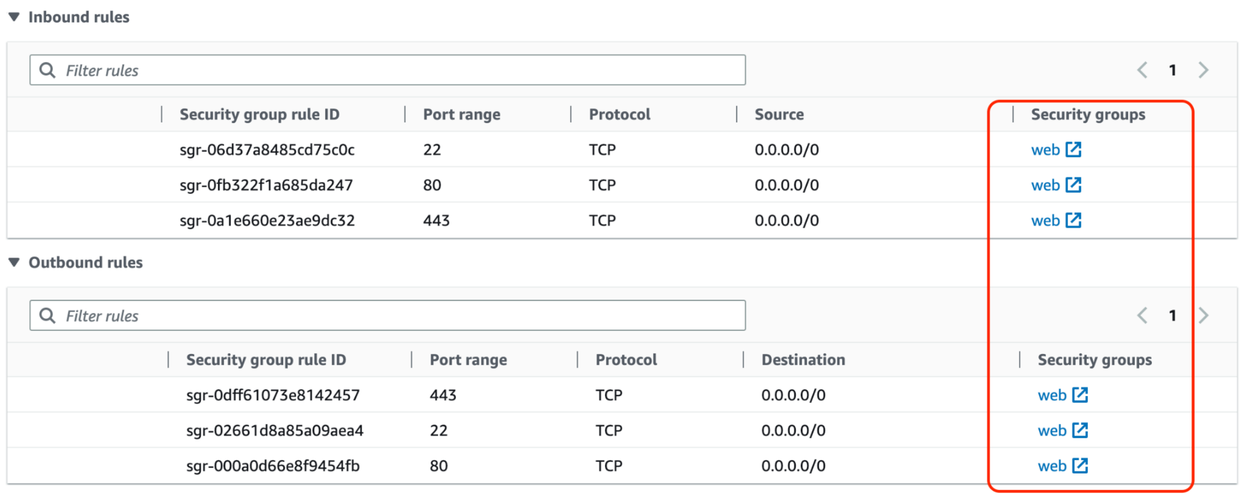
Platform Ubuntu (Inferred)	AMI ID ami-07dc2dd8e0efbc46a	Monitoring disabled
Platform details Linux/UNIX	AMI name ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-20230213	Termination protection Disabled



Networking details Info

Public IPv4 address 3.83.125.175 | open address	Private IPv4 addresses 10.0.1.110	VPC ID vpc-06ad4967ce1059fce (cloudethix-vpc)
Public IPv4 DNS –	Private IP DNS name (IPv4 only) ip-10-0-1-110.ec2.internal	
Subnet ID subnet-0ac4a029f9f130d8a	IPV6 addresses –	Secondary private IPv4 addresses –
Availability zone us-east-1a	Carrier IP addresses (ephemeral) –	Outpost ID –
Use RBN as guest OS hostname Disabled	Answer RBN DNS hostname IPv4 Disabled	

Network Interfaces (1) Info



Inbound rules

Filter rules

Security group rule ID	Port range	Protocol	Source	Security groups
sgr-06d37a8485cd75c0c	22	TCP	0.0.0.0/0	web
sgr-0fb322f1a685da247	80	TCP	0.0.0.0/0	web
sgr-0a1e660e23ae9dc32	443	TCP	0.0.0.0/0	web

Outbound rules

Filter rules

Security group rule ID	Port range	Protocol	Destination	Security groups
sgr-0dff61073e8142457	443	TCP	0.0.0.0/0	web
sgr-02661d8a85a09aea4	22	TCP	0.0.0.0/0	web
sgr-000a0d66e8f9454fb	80	TCP	0.0.0.0/0	web

Que 5 →

➔ Create three tier application architecture using Terraform.

➔ Three tier application architecture will require below resources.

VPC

- VPC with 2 Public and 2 Private subnets including IGW / NAT GW / RT & Subnet association.

ALB

- Create 1 Security Group named alb-sg. Allow traffic on port 80 & 443 from 0.0.0.0/0
- Create 1 Application Load Balancer in the public Subnet of your VPC & attach alb-sg security group to ALB.

WEB-SERVER

- Create 1 Security Group named web-sg. Allow traffic on port 80 from 0.0.0.0/0
- Create 1 EC2 instance named web-ec2 in public Subnet & attach web-sg security group to EC2 instance.

APPLICATION

- Create 1 Security Group named app-sg. Allow traffic on port 8080 from 0.0.0.0/0.
- Create 1 EC2 instance named app-ec2 in private Subnet & attach app-sg security group to EC2 instance.

DATABASE

- Create 1 Security Group named rds-sg. Allow traffic on port 3306 from 0.0.0.0/0.
- Create 1 MySQL RDS instance with aws_db_instance with type db.t3.medium. Attach rds-sg to RDS instance.

S3 BUCKET

- Create an S3 bucket to store the data.➔ Note :-
    - Use data sources to fetch the AMI ID.
    - Use Dynamic block for security group ingress & egress to add rules.
    - Generate ssh key using aws_key_pair resource & map to EC2 Instances.

Main.tf

```
#AWS Provider


terraform {
 required_providers {
   aws = {
     source = "hashicorp/aws"
     version = "4.52.0"
   }
 }
}

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

App.tf

```
#Application Security Group

resource "aws_security_group" "cloudethix-sg-app" {
 name        = "allow_app"
 description = "Allow app inbound traffic"
 vpc_id      = aws_vpc.cloudethix-vpc.id

 ingress {
   description      = "app from VPC"
   from_port        = 8080
   to_port          = 8080
   protocol         = "tcp"
   security_groups  = ["${aws_security_group.cloudethix-sg-web.id}"]
 }

 tags = {
   Name = "3T-app"
 }
}



#Application EC2

 resource "aws_instance" "app" {
 ami            = "ami-0aa7d40eeae50c9a9"
 instance_type = "t2.micro"
 key_name       = aws_key_pair.cloudethix-key-pair.key_name
 security_groups = ["${aws_security_group.cloudethix-sg-web.id}"]
 subnet_id      = aws_subnet.cloudethix-sub-private01.id

 tags = {
   Name = "3T-app"
 }
}
```

Web.tf

```
#WEB Security Group

resource "aws_security_group" "cloudethix-sg-web" {
 name        = "allow_web"
 description = "Allow web inbound traffic"
 vpc_id      = aws_vpc.cloudethix-vpc.id

 ingress {
   description      = "web from VPC"
   from_port        = 443
   to_port          = 443
   protocol         = "tcp"
   security_groups  = ["${aws_security_group.cloudethix-lb-sg.id}"]
 }

 ingress {
   description      = "web from VPC"
   from_port        = 80
   to_port          = 80
   protocol         = "tcp"
   security_groups  = ["${aws_security_group.cloudethix-lb-sg.id}"]
 }

 tags = {
   Name = "3T-WEB"
 }
}

#WEB EC2

 resource "aws_instance" "web" {
 ami            = "ami-0aa7d40eeae50c9a9"
 instance_type = "t2.micro"
 key_name      = "${aws_key_pair.cloudethix-key-pair.key_name}"
 security_groups = [aws_security_group.cloudethix-sg-web.id]
 subnet_id     = aws_subnet.cloudethix-sub-public01.id

 tags = {
   Name = "3T-WEB"
 }
}
```

Key_pair.tf

```
#Key Pair to Access EC2

resource "aws_key_pair" "cloudethix-key-pair" {
 key_name   = "3Tier-key"
 public_key = "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDkc/q0xTIzecyMPE/sjWmR9g8sP8/Xj7itL9kXRzHtYLT3T13E2OAfVC
t4zZ/eQIoTJuQWstL+slKG9anXKkrwKf4qF/2wxsZZ8Z9hUYV21KIGz9lDgmkueB3MKi07VyFhpBO1S2inbpjl
lkp0hp1AcYVOS0ulMhCC+X4y8yE5amG53/qiSLPnF0dBCa9icku0YYj6RZrjKfeL2S8uwBIMeTnPbpxn8BxkKI
djRErZjfuxASH39SYmWa7lpW3m2VReFc7t23ZjlEKFOaZWbwSK88L0EduRPV7+JbJDyCO/UxA+8E5/oJ9j9rt8
/MmE1YV5Nnf8UiHrGhH3WJkMBDZN"
}
```

Loadbalancer.tf

```

#Load Balancer Security Group

resource "aws_security_group" "cloudethix-lb-sg" {
 name        = "allow_lb"
 description = "Allow lb inbound traffic"
 vpc_id      = aws_vpc.cloudethix-vpc.id

 ingress {
   description      = "web from VPC"
   from_port        = 80
   to_port          = 80
   protocol         = "tcp"
   cidr_blocks      = ["0.0.0.0/0"]
 }

 ingress {
   description      = "web from VPC"
   from_port        = 443
   to_port          = 443
   protocol         = "tcp"
   cidr_blocks      = ["0.0.0.0/0"]
 }

 tags = {
   Name = "3T-LB"
 }
}

#Load Balancer for WEB

resource "aws_lb" "cloudethix-lb" {
 name               = "3T-application-lb"
 internal           = false
 load_balancer_type = "application"
 security_groups    = [aws_security_group.cloudethix-lb-sg.id]
 subnets            =
[aws_subnet.cloudethix-sub-public01.id,aws_subnet.cloudethix-sub-public02.id]
 enable_deletion_protection = true

 tags = {
   Environment = "3T-LB"
 }
}
```

Rds.tf

```
#RDS Security Group

resource "aws_security_group" "cloudethix-sg-rds" {
 name        = "allow_rds"
 description = "Allow rds inbound traffic"
 vpc_id      = aws_vpc.cloudethix-vpc.id

 ingress {
   description       = "rds from VPC"
   from_port         = 3306
   to_port           = 3306
   protocol          = "tcp"
   security_groups   = ["${aws_security_group.cloudethix-sg-app.id}"]
 }

 tags = {
   Name = "3T-RDS"
 }
}


#RDS DB Subnet Group

resource "aws_db_subnet_group" "cloudethix-rds-db-sub" {
 name       = "rds-db-sub"
 subnet_ids = ["${aws_subnet.cloudethix-sub-private01.id}",
"${aws_subnet.cloudethix-sub-private02.id}"]

 tags = {
   Name = "3T-RDS"
 }
}


#RDS Instance

resource "aws_db_instance" "cloudethix-rds" {
 allocated_storage    = 10
 db_name              = "mydb"
 engine               = "mysql"
 engine_version       = "5.7"
 instance_class       = "db.t3.micro"
 username             = "foo"
 password             = "foobarbaz"
 parameter_group_name = "default.mysql5.7"
 skip_final_snapshot  = true
 publicly_accessible  = false
 db_subnet_group_name = aws_db_subnet_group.cloudethix-rds-db-sub.name
 vpc_security_group_ids = ["${aws_security_group.cloudethix-sg-rds.id}"]
}
```

S3.tf

```
#S3 Bucket

resource "aws_s3_bucket" "cloudethix-s3-bucket" {
 bucket = "cloudethix-3tier-arch"

 tags = {
   Name        = "3T-S3"
 }
}

#S3 Public Access Block

resource "aws_s3_bucket_public_access_block" "cloudethix-s3-access" {
 bucket = aws_s3_bucket.cloudethix-s3-bucket.id

 block_public_acls       = true
 block_public_policy     = true
 ignore_public_acls      = true
 restrict_public_buckets = true
}
```

Vpc.tf

```
#VPC
resource "aws_vpc" "cloudethix-vpc" {
 cidr_block       = "10.0.0.0/16"
 instance_tenancy = "default"

 tags = {
   Name = "3T-VPC"
 }
}

#private subnets
resource "aws_subnet" "cloudethix-sub-private01" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = "10.0.1.0/24"
 availability_zone       = "us-east-1a"
 map_public_ip_on_launch = true

 tags = {
   Name = "3T-SUB-PRIVATE"
 }
}

resource "aws_subnet" "cloudethix-sub-private02" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = "10.0.3.0/24"
 availability_zone       = "us-east-1b"
 map_public_ip_on_launch = true

 tags = {
   Name = "3T-SUB-PRIVATE"
 }
}



#public subnets
resource "aws_subnet" "cloudethix-sub-public01" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = "10.0.2.0/24"
 availability_zone       = "us-east-1a"
 map_public_ip_on_launch = true

 tags = {
   Name = "3T-SUB-PUBLIC"
 }
}

resource "aws_subnet" "cloudethix-sub-public02" {
 vpc_id                  = aws_vpc.cloudethix-vpc.id
 cidr_block              = "10.0.4.0/24"
 availability_zone       = "us-east-1b"
 map_public_ip_on_launch = true

 tags = {
   Name = "3T-SUB-PUBLIC"
 }
}
```

```
#Elastic IP
resource "aws_eip" "cloudethix-eip" {
 vpc      = true

 tags = {
   Name = "3T-EIP"
 }
}



#IGW
resource "aws_internet_gateway" "cloudethix-igw" {
 vpc_id = aws_vpc.cloudethix-vpc.id

 tags = {
   Name = "3T-IGW"
 }
}



#Public NAT
resource "aws_nat_gateway" "cloudethix-nat" {
 allocation_id = aws_eip.cloudethix-eip.id
 subnet_id     = aws_subnet.cloudethix-sub-public01.id

 tags = {
   Name = "3T-NAT"
 }
}



#Route Table
resource "aws_route_table" "cloudethix-RT-public" {
 vpc_id = aws_vpc.cloudethix-vpc.id

 tags = {
   Name = "3T-RT-PUBLIC"
 }
}

resource "aws_route_table" "cloudethix-RT-private" {
 vpc_id = aws_vpc.cloudethix-vpc.id

 tags = {
   Name = "3T-RT-PRIVATE"
 }
}


#Route
resource "aws_route" "cloudethix-route-public" {
 route_table_id             = aws_route_table.cloudethix-RT-public.id
 destination_cidr_block     = "0.0.0.0/0"
 gateway_id                 = aws_internet_gateway.cloudethix-igw.id
}

resource "aws_route" "cloudethix-route-private" {
 route_table_id             = aws_route_table.cloudethix-RT-private.id
 destination_cidr_block     = "0.0.0.0/0"
 gateway_id                 = aws_nat_gateway.cloudethix-nat.id
}

#Route Table Association
resource "aws_route_table_association" "cloudethix-RTASS-public" {
 subnet_id      = aws_subnet.cloudethix-sub-public01.id
 route_table_id = aws_route_table.cloudethix-RT-public.id
}

resource "aws_route_table_association" "cloudethix-RTASS-private" {
 subnet_id     = aws_subnet.cloudethix-sub-private01.id
 route_table_id = aws_route_table.cloudethix-RT-private.id
}
```

GIT REPO 👍

https://github.com/haneefshaikh/Terraform_AWS_Assignment