Terraform project

Step -1

Install terraform by creating one directory

```
mkdir terraform
```

no winside it run these command

```
sudo yum install -y yum-utils shadow-utils
sudo yum-config-manager --add-repo
https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo
sudo yum -y install terraform
```

step-2

install aws cli and run aws configure

to access aws resources for these you need to create I am uuser and assign permission to create resources that you want

step 3-

start create terraform configuration file

main.tf

Configuration file i.e state file

```
provider "aws" {
  region = "us-east-1"  # Replace with your desired region
}

# Create a VPC
resource "aws_vpc" "Bickyvpc" {
  cidr_block = "10.0.0.0/16"

  tags = {
    Name = "Bickyvpc"
  }
}

# Create subnet 1 (Public)
resource "aws_subnet" "subnet1" {
  vpc_id                  = aws_vpc.Bickyvpc.id
  cidr_block              = "10.0.0.0/24"
  availability_zone       = "us-east-1a"
  map_public_ip_on_launch = true

  tags = {
    Name = "PublicSubnet1"
  }
}

# Create subnet 2 (Public)
resource "aws_subnet" "subnet2" {
  vpc_id                  = aws_vpc.Bickyvpc.id
  cidr_block              = "10.0.1.0/24"
  availability_zone       = "us-east-1b"
  map_public_ip_on_launch = true

  tags = {
    Name = "PublicSubnet2"
  }
}

# Create an Internet Gateway
resource "aws_internet_gateway" "igw" {
  vpc_id = aws_vpc.Bickyvpc.id

  tags = {
    Name = "BickyInternetGateway"
  }
}

# Create a Route Table
resource "aws_route_table" "route" {
  vpc_id = aws_vpc.Bickyvpc.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.igw.id
  }

  tags = {
    Name = "PublicRouteTable"
  }
}

# Associate Route Table with Subnet 1
resource "aws_route_table_association" "route1" {
  subnet_id      = aws_subnet.subnet1.id
  route_table_id = aws_route_table.route.id
}

# Associate Route Table with Subnet 2
resource "aws_route_table_association" "route2" {
  subnet_id      = aws_subnet.subnet2.id
  route_table_id = aws_route_table.route.id
}

# Create a Security Group to allow SSH and HTTP
resource "aws_security_group" "web_sg" {
  vpc_id = aws_vpc.Bickyvpc.id

  # Allow SSH
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  # Allow HTTP
  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  # Allow all outbound traffic
  egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    Name = "WebSecurityGroup"
  }
}
```

```
# Allocate an Elastic IP for NAT Gateway
resource "aws_eip" "nat_eip" {
  domain = "vpc"

  tags = {
    Name = "NAT-Gateway-EIP"
  }
}

# Create a NAT Gateway
resource "aws_nat_gateway" "nat" {
  allocation_id = aws_eip.nat_eip.id
  subnet_id     = aws_subnet.subnet1.id

  tags = {
    Name = "NATGateway"
  }
}

# Create an EC2 instance in Subnet 1
resource "aws_instance" "web_server" {
  ami                    = "ami-085ad6ae776d8f09c"  # Amazon Linux 2 AMI for us-east-1
  instance_type          = "t2.micro"
  subnet_id              = aws_subnet.subnet1.id
  vpc_security_group_ids = [aws_security_group.web_sg.id]  # ✅ Correct reference
  associate_public_ip_address = true

  tags = {
    Name = "WebServer"
  }
}
```

Your Terraform code will create the following AWS resources:

1️⃣ Networking Resources (VPC & Subnets)

VPC → Bickyvpc (CIDR: 10.0.0.0/16)

Subnet 1 → PublicSubnet1 (CIDR: 10.0.0.0/24, AZ: us-east-1a)

Subnet 2 → PublicSubnet2 (CIDR: 10.0.1.0/24, AZ: us-east-1b)

2️⃣ Internet & Routing

Internet Gateway → BickyInternetGateway (Allows public internet access)

Route Table → PublicRouteTable (Routes 0.0.0.0/0 traffic to the Internet Gateway)

Route Table Associations → Links subnet1 & subnet2 to PublicRouteTable

3️⃣ Security & Access

Security Group → WebSecurityGroup

✅ Allows SSH (Port 22) from anywhere (0.0.0.0/0)

✅ Allows HTTP (Port 80) from anywhere (0.0.0.0/0)

✅ Allows all outbound traffic

4️⃣ NAT & Elastic IP

Elastic IP → NAT-Gateway-EIP (For NAT Gateway)

NAT Gateway → NATGateway (To allow private instances to access the internet)

5️⃣ Compute (EC2 Instance)

EC2 Instance → WebServer

✅ AMI: ami-085ad6ae776d8f09c (Amazon Linux 2)

✅ Instance Type: t2.micro

✅ Subnet: PublicSubnet1

✅ Security Group: WebSecurityGroup

✅ Public IP: Assigned (map_public_ip_on_launch = true)

TERRAFORM INTERVOEW

What is Terraform?

Terraform is an open-source infrastructure as code (IaC) tool developed by HashiCorp that allow users to define and provision data center infrastructure using a high-level configuration language called HCL (HashiCorp Configuration Language) or JSON.

2. What are the main features of Terraform?

Terraform's main features include Infrastructure as Code (IaC), execution plans, resource graphs,change automation.

What is the difference between Terraform and other IaC tools Like Ansible, Puppet, and Chef?

Terraform focuses on infrastructure provisioning, is declarative, and uses HCL. Tools like Ansible, Puppet, and Chef focus on configuration management and are procedural.

4. What is a provider in Terraform?

A provider is a plugin that Terraform uses to manage an external

API. Providers define the resources and data sources available.

What is a State File in Terraform?

A state file in Terraform is a JSON-formatted file (terraform.tfstate) that stores metadata about the infrastructure managed by Terraform. It helps Terraform to track the resources,infrastructure deployed using terraform.

How can you secure the state file in Terraform?

State files can be secured by storing them in remote backends

with proper access controls.

What is the purpose of the terraform init command?

terraform init initializes a working directory containing

Terraform configuration files, downloads the necessary provider plugins, and prepares the environment

What does the terraform plan command do?

terraform plan creates an execution plan, showing what actions

Terraform will take to achieve the desired state defined in the configuration

What is the terraform apply command used for?

terraform apply applies the changes required to reach the

desired state of the configuration. It executes the plan created by terraform plan.

What is the purpose of the terraform destroy command?

terraform destroy is used to destroy the infrastructure managed

by Terraform. It removes all the resources defined in the terraform configuration file.