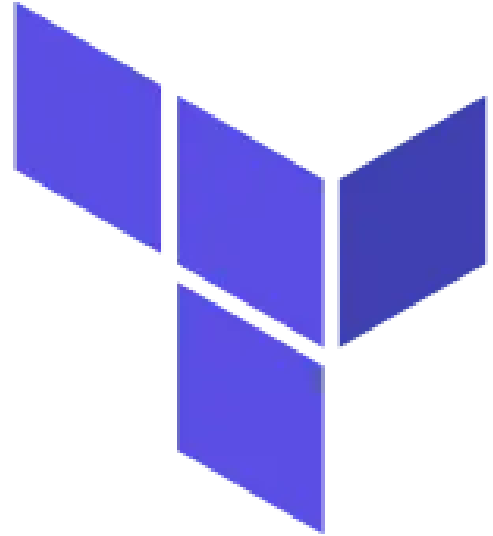


Infrastructure As A Code using Terraform

By: Salma Salah

Guidance & Support : Eng. Saad El-Kenawy



Why IAAC



1.

Increased consistency
and execution speed

2.

Reusing code for
infrastructure replication

3.

Validating code before
executing using code
review and test cases

4.

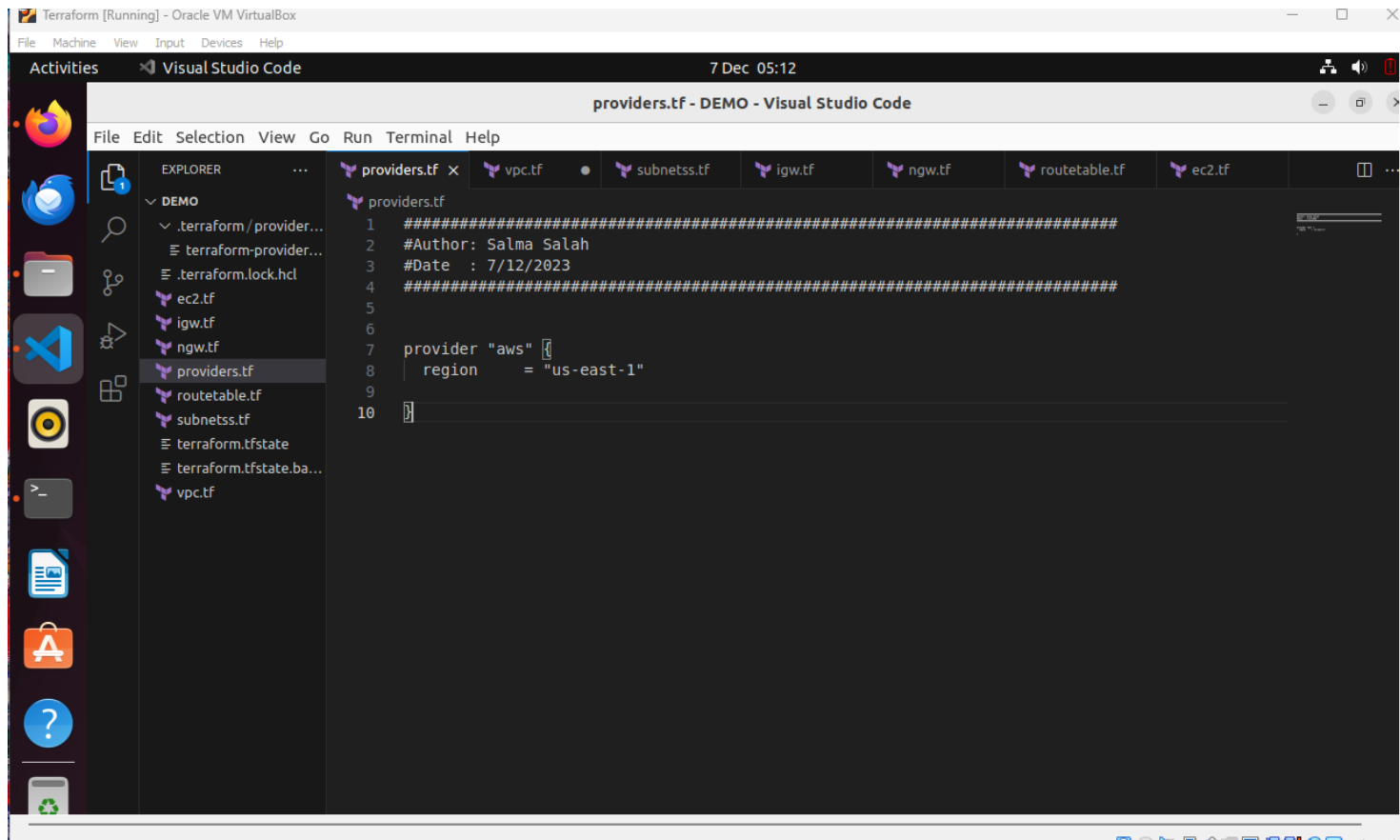
Versioning the IaC helps
us to rollback to the
previous version

Task Scenario

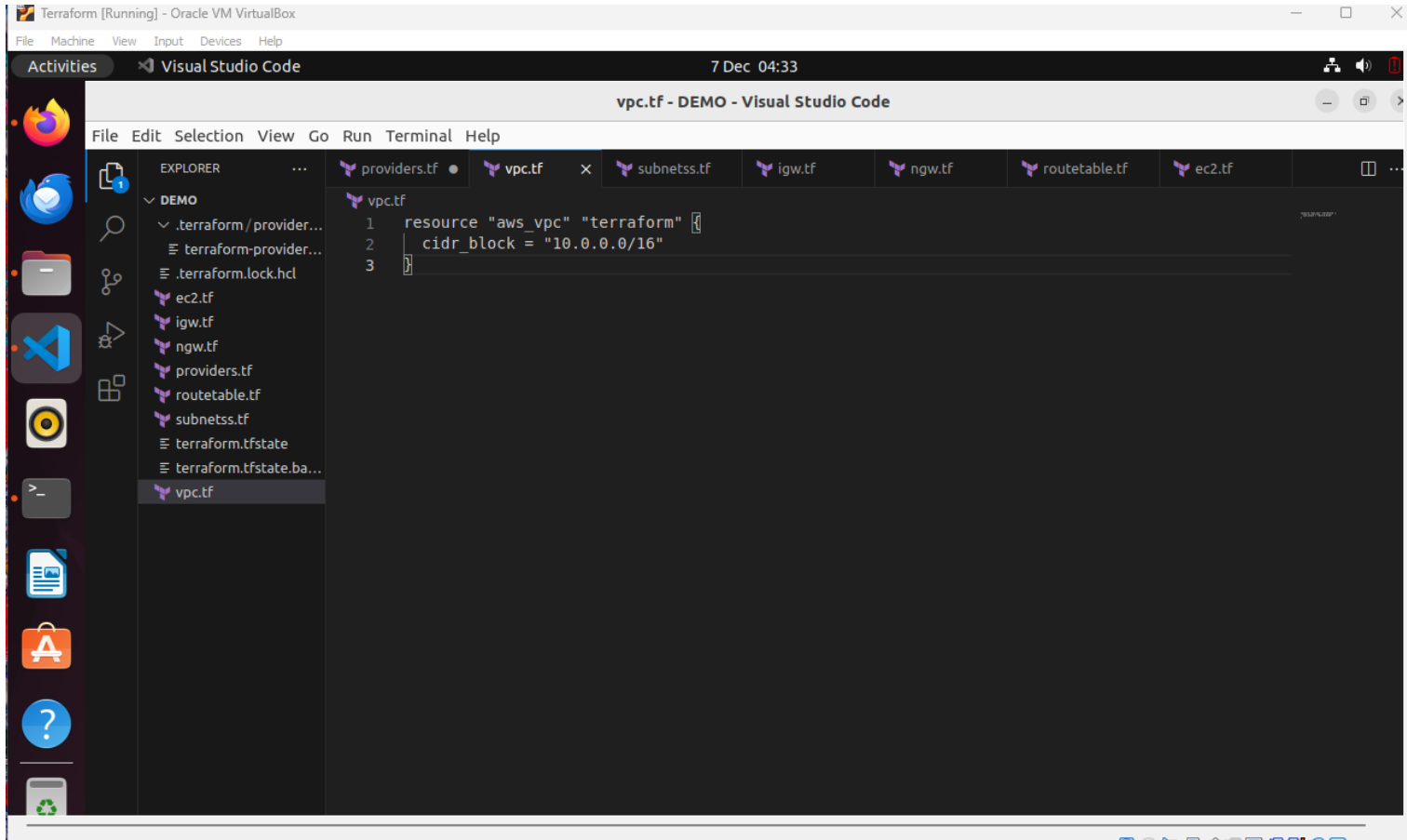
Create a simple Highly Available and Secured Network Architecture with Two Public Subnets; Two Private subnets . Internet Gateway , Nat Gateway, Route Tables and an EC2 instance.

Using Terraform

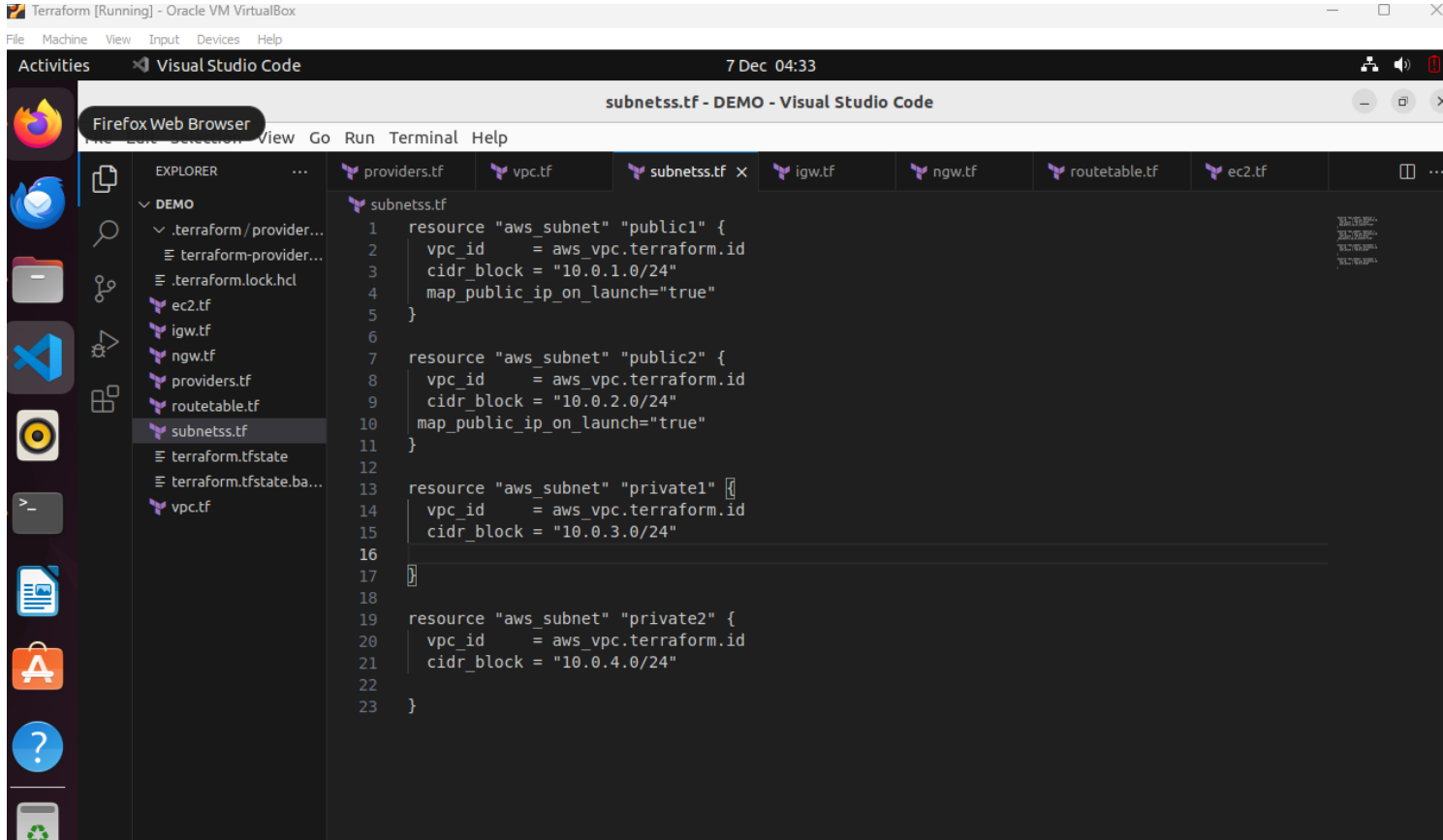
1) Providers



2) VPC



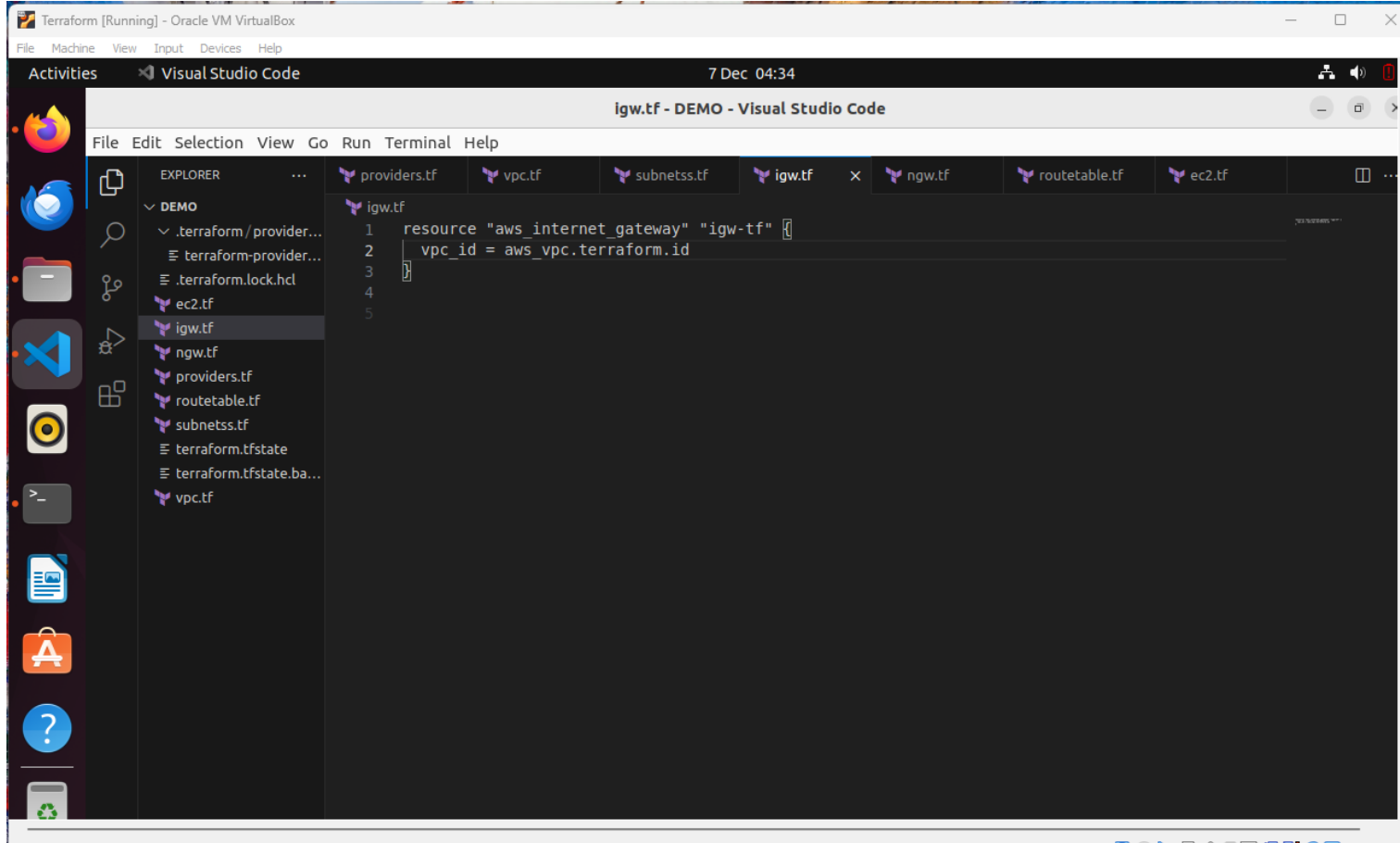
3) Two Public subnets & Two Private subnets



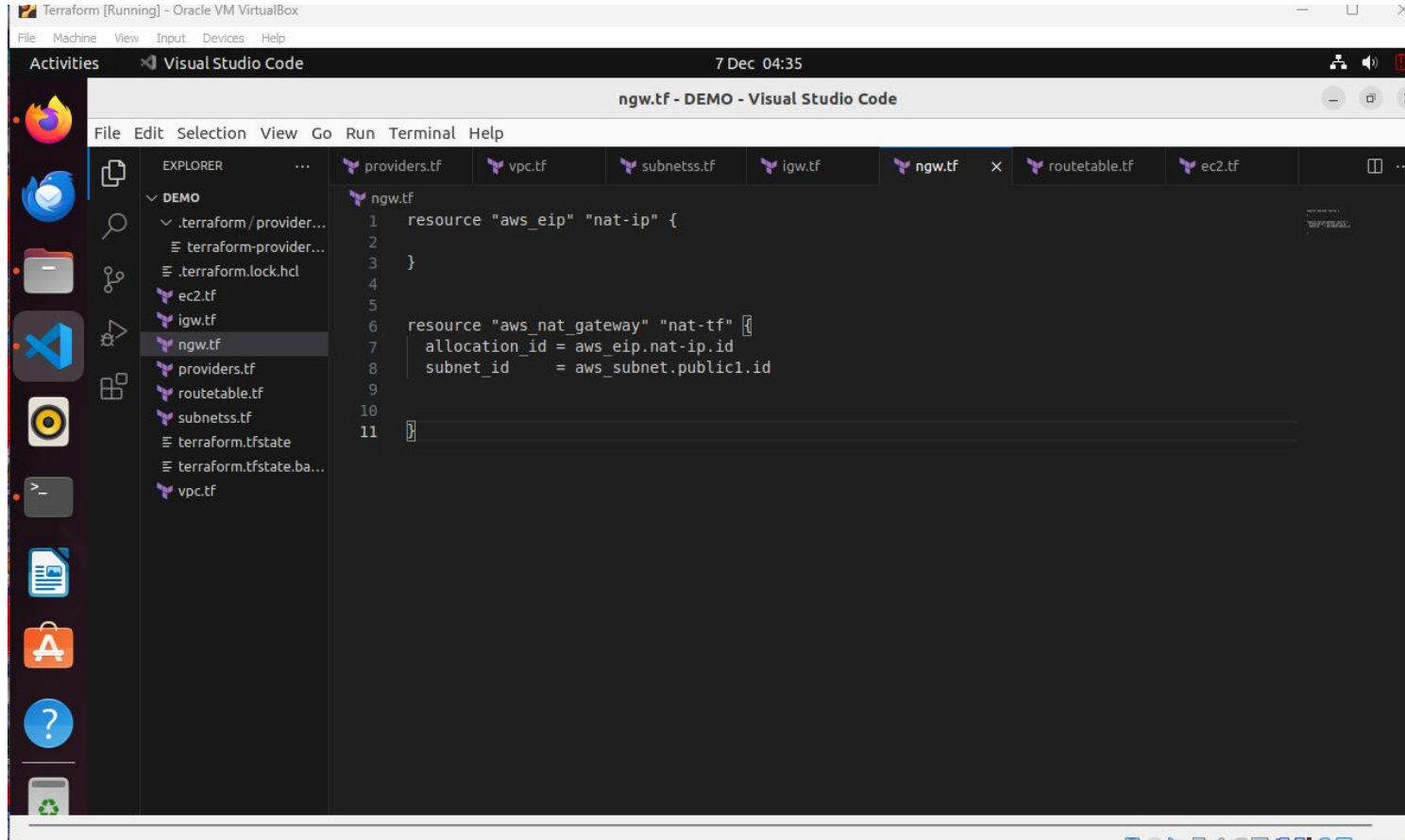
The screenshot shows a Visual Studio Code editor window titled "subnetss.tf - DEMO - Visual Studio Code". The Explorer sidebar on the left shows a project structure with a "DEMO" folder containing several Terraform files: ".terraform/provider...", ".terraform.lock.hcl", "ec2.tf", "igw.tf", "ngw.tf", "providers.tf", "routetable.tf", "subnetss.tf" (selected), "terraform.tfstate", "terraform.tfstate.ba...", and "vpc.tf". The main editor area displays the content of "subnetss.tf", which defines four AWS subnets. The first two are public subnets, and the last two are private subnets. The code is as follows:

```
1 resource "aws_subnet" "public1" {
2   vpc_id      = aws_vpc.terraform.id
3   cidr_block  = "10.0.1.0/24"
4   map_public_ip_on_launch="true"
5 }
6
7 resource "aws_subnet" "public2" {
8   vpc_id      = aws_vpc.terraform.id
9   cidr_block  = "10.0.2.0/24"
10  map_public_ip_on_launch="true"
11 }
12
13 resource "aws_subnet" "private1" {
14   vpc_id      = aws_vpc.terraform.id
15   cidr_block  = "10.0.3.0/24"
16 }
17
18
19 resource "aws_subnet" "private2" {
20   vpc_id      = aws_vpc.terraform.id
21   cidr_block  = "10.0.4.0/24"
22 }
23 }
```

4) Internet Gateway



5) Nat Gateway



Terraform [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Activities Visual Studio Code 7 Dec 04:35

ngw.tf - DEMO - Visual Studio Code

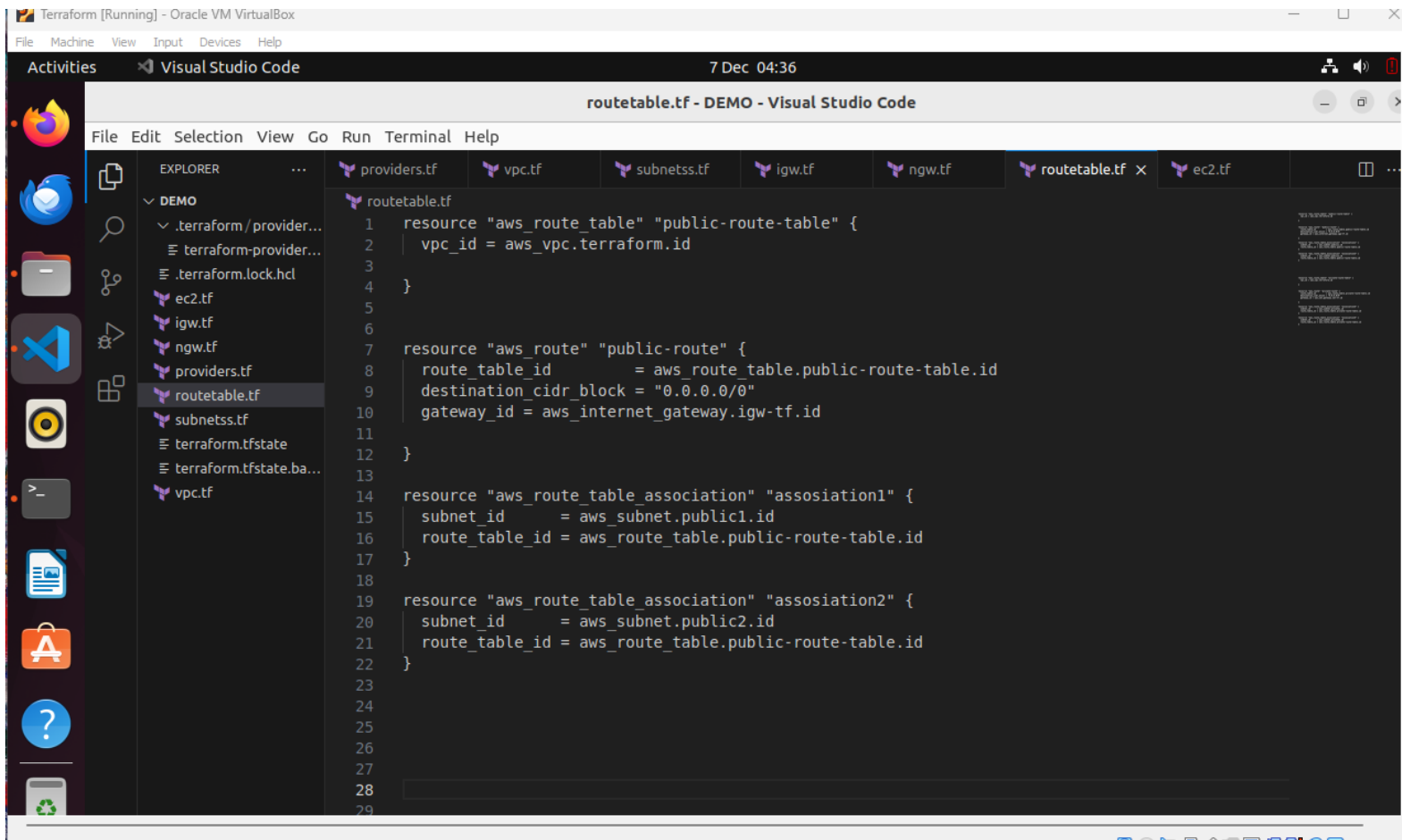
File Edit Selection View Go Run Terminal Help

EXPLORER

- DEMO
 - .terraform/provider...
 - terraform-provider...
 - .terraform.lock.hcl
 - ec2.tf
 - igw.tf
 - ngw.tf
 - providers.tf
 - routetable.tf
 - subnetss.tf
 - terraform.tfstate
 - terraform.tfstate.ba...
 - vpc.tf

```
ngw.tf
1 resource "aws_eip" "nat-ip" {
2
3 }
4
5
6 resource "aws_nat_gateway" "nat-tf" {
7   allocation_id = aws_eip.nat-ip.id
8   subnet_id     = aws_subnet.public1.id
9
10
11 }
```

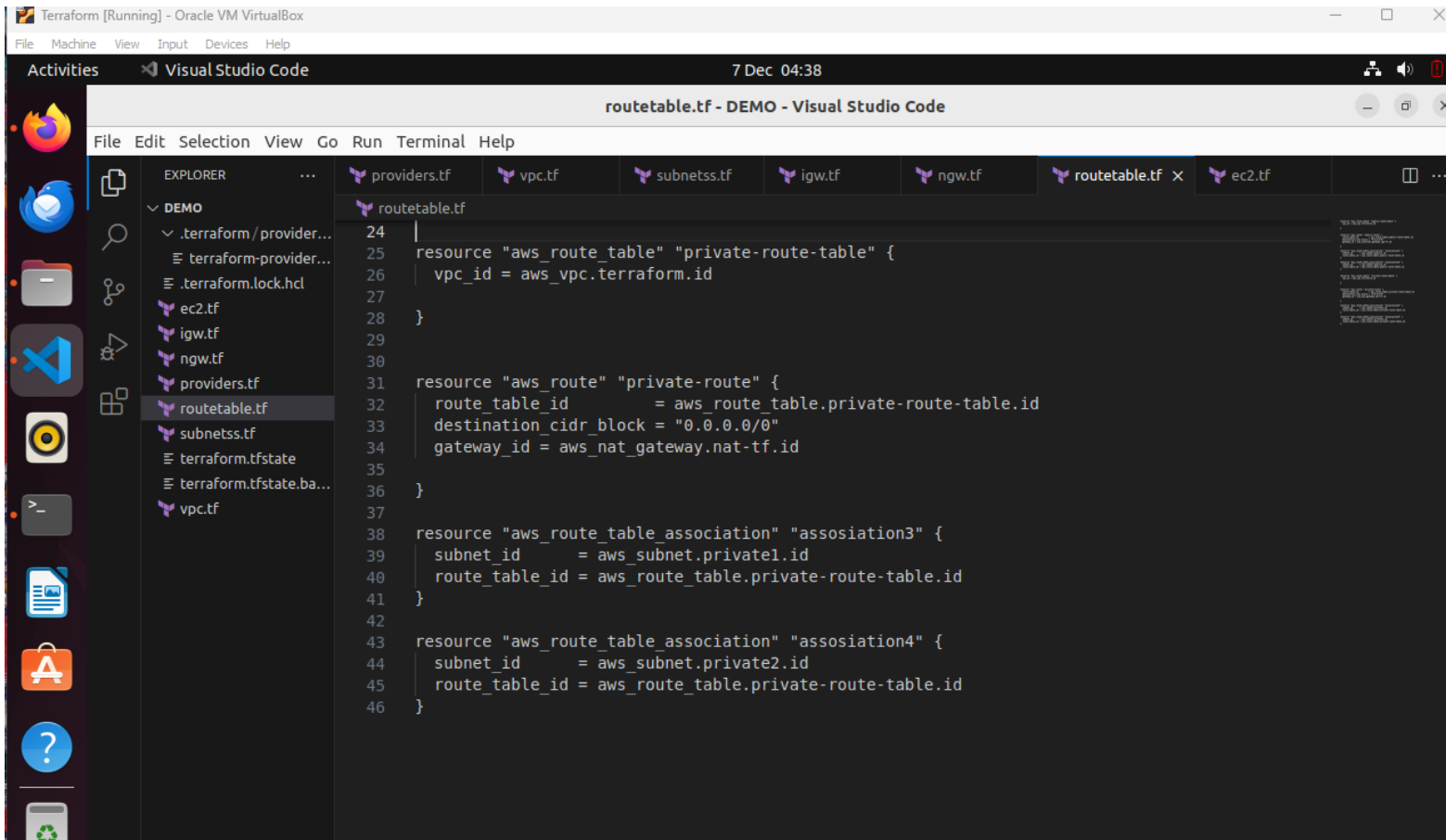

6) Public Route Table & Subnet Association



The screenshot shows a Visual Studio Code window titled "routetable.tf - DEMO - Visual Studio Code" with a dark theme. The Explorer sidebar on the left shows a project structure under "DEMO" with files like ".terraform/provider...", ".terraform.lock.hcl", "ec2.tf", "igw.tf", "ngw.tf", "providers.tf", "routetable.tf" (selected), "subnetss.tf", "terraform.tfstate", and "terraform.tfstate.ba...". The main editor displays the content of "routetable.tf" with the following Terraform code:

```
1 resource "aws_route_table" "public-route-table" {
2   vpc_id = aws_vpc.terraform.id
3 }
4
5
6
7 resource "aws_route" "public-route" {
8   route_table_id = aws_route_table.public-route-table.id
9   destination_cidr_block = "0.0.0.0/0"
10  gateway_id = aws_internet_gateway.igw-tf.id
11 }
12
13
14 resource "aws_route_table_association" "assosiation1" {
15   subnet_id = aws_subnet.public1.id
16   route_table_id = aws_route_table.public-route-table.id
17 }
18
19 resource "aws_route_table_association" "assosiation2" {
20   subnet_id = aws_subnet.public2.id
21   route_table_id = aws_route_table.public-route-table.id
22 }
23
24
25
26
27
28
29
```

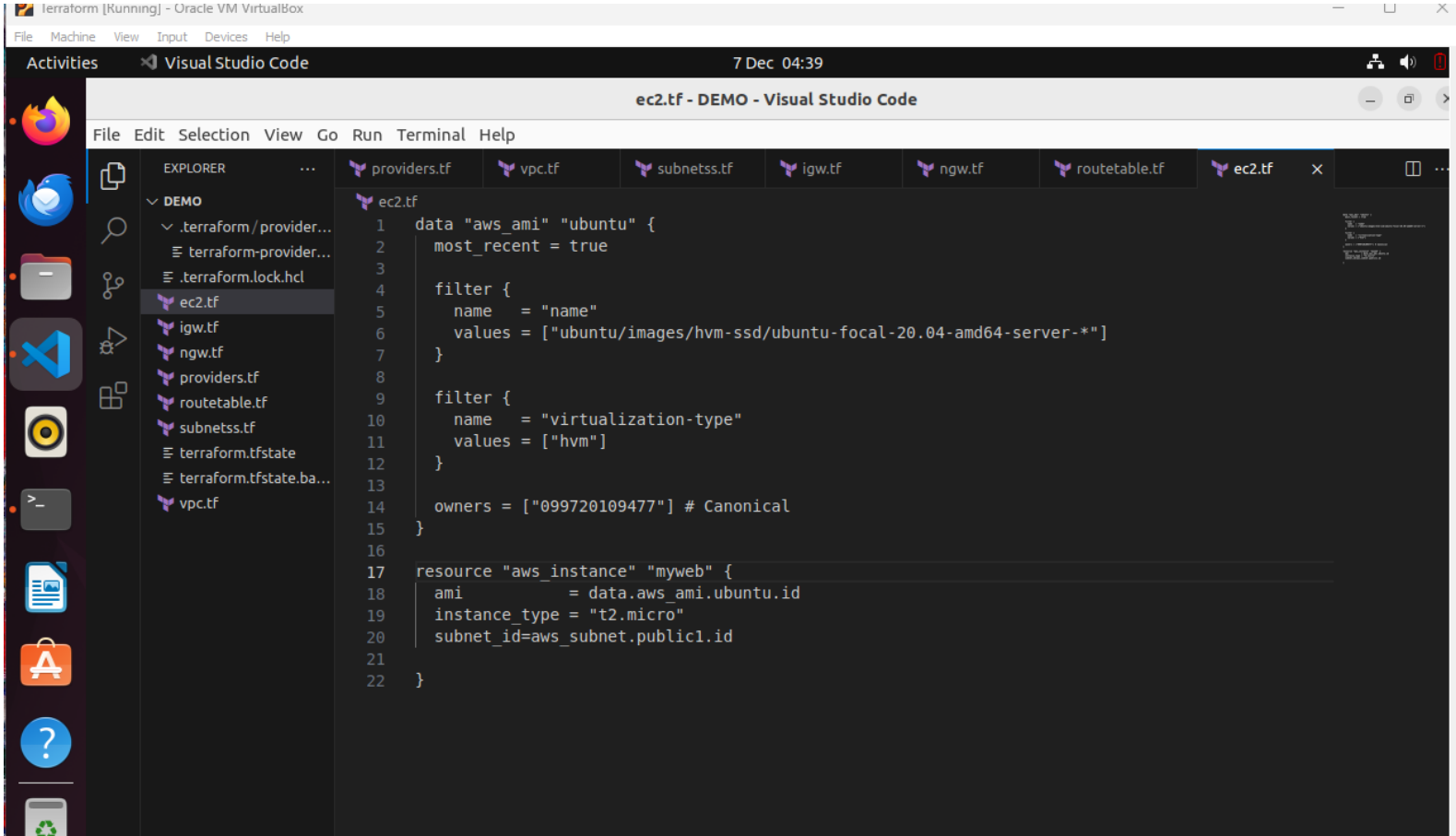
7) Private Route Table & Subnet Association



The screenshot displays the Visual Studio Code interface with a Terraform project. The Explorer sidebar on the left shows a directory structure under 'DEMO' including files like 'providers.tf', 'vpc.tf', 'subnetss.tf', 'igw.tf', 'ngw.tf', 'routetable.tf', 'ec2.tf', 'terraform.tfstate', and 'terraform.tfstate.ba...'. The 'routetable.tf' file is selected and open in the main editor. The code defines an AWS private route table, a private route, and two route table associations for private subnets.

```
24 |  
25 resource "aws_route_table" "private-route-table" {  
26   vpc_id = aws_vpc.terraform.id  
27 }  
28  
29  
30  
31 resource "aws_route" "private-route" {  
32   route_table_id      = aws_route_table.private-route-table.id  
33   destination_cidr_block = "0.0.0.0/0"  
34   gateway_id = aws_nat_gateway.nat-tf.id  
35 }  
36  
37  
38 resource "aws_route_table_association" "assosiation3" {  
39   subnet_id      = aws_subnet.private1.id  
40   route_table_id = aws_route_table.private-route-table.id  
41 }  
42  
43 resource "aws_route_table_association" "assosiation4" {  
44   subnet_id      = aws_subnet.private2.id  
45   route_table_id = aws_route_table.private-route-table.id  
46 }
```

8) EC2 instance



The screenshot shows a Visual Studio Code editor window titled "ec2.tf - DEMO - Visual Studio Code". The interface includes a sidebar with the Explorer view showing a project structure under "DEMO" with files like ".terraform/provider...", ".terraform.lock.hcl", "ec2.tf", "igw.tf", "ngw.tf", "providers.tf", "routetable.tf", "subnetss.tf", "terraform.tfstate", "terraform.tfstate.ba...", and "vpc.tf". The main editor area displays the content of "ec2.tf", which is a Terraform configuration for an AWS EC2 instance. The configuration includes a data source for the Ubuntu AMI and an instance resource.

```
1 data "aws_ami" "ubuntu" {
2   most_recent = true
3
4   filter {
5     name      = "name"
6     values    = ["ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-*"]
7   }
8
9   filter {
10    name      = "virtualization-type"
11    values    = ["hvm"]
12  }
13
14  owners      = ["099720109477"] # Canonical
15 }
16
17 resource "aws_instance" "myweb" {
18   ami           = data.aws_ami.ubuntu.id
19   instance_type = "t2.micro"
20   subnet_id     = aws_subnet.public1.id
21 }
22 }
```

Used Terraform commands

- 1) terraform init: To Prepare working directory for other commands
- 2) terraform validate: To Check whether the configuration is valid
- 3) terraform plan: To Show changes required by the current configuration
- 4) terraform apply: To Create or update infrastructure
- 5) terraform destroy: To Destroy previously-created infrastructure