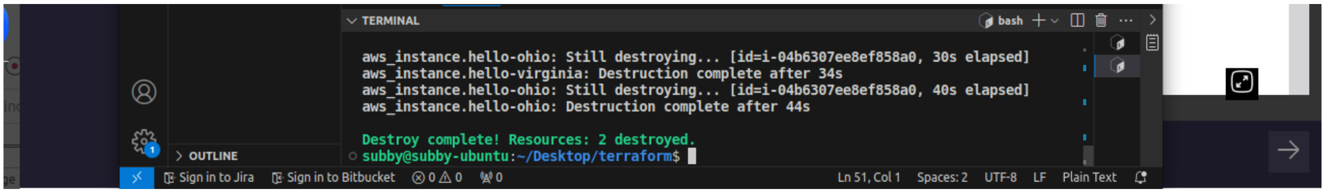
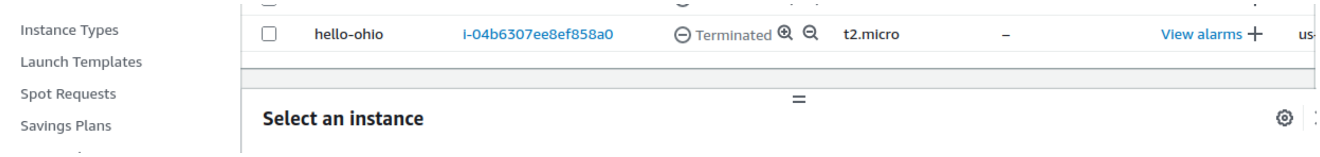
1. Destroy the previous deployments
2. Create a VPC with the required components using Terraform
3. Deploy an EC2 instance inside the VPC



```
aws_instance.hello-ohio: Still destroying... [id=i-04b6307ee8ef858a0, 30s elapsed]
aws_instance.hello-virginia: Destruction complete after 34s
aws_instance.hello-ohio: Still destroying... [id=i-04b6307ee8ef858a0, 40s elapsed]
aws_instance.hello-ohio: Destruction complete after 44s

Destroy complete! Resources: 2 destroyed.
subby@subby-ubuntu:~/Desktop/terraform$
```



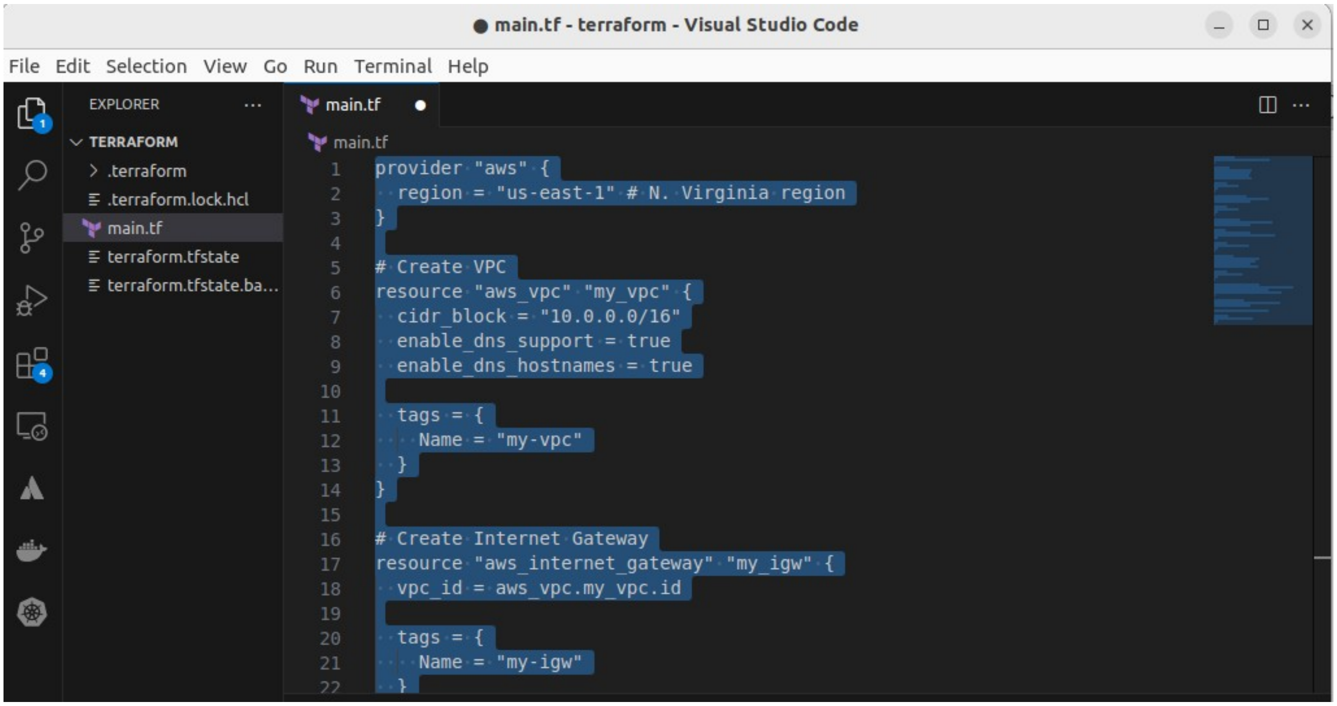
hello-ohio i-04b6307ee8ef858a0 Terminated t2.micro – View alarms us-

Select an instance

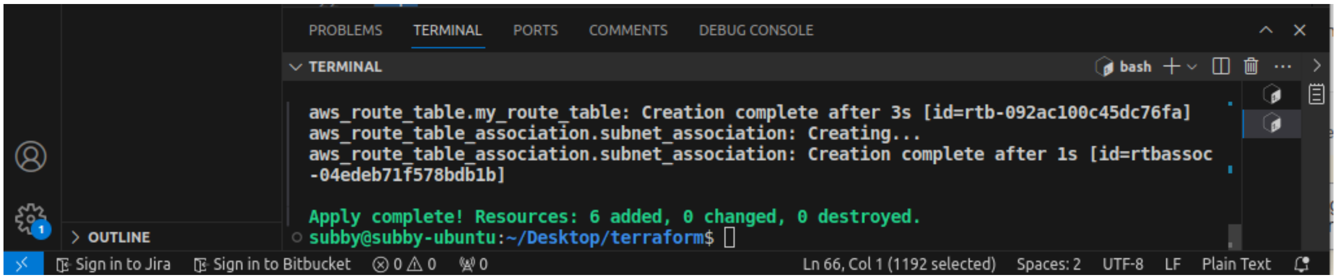
VPC, subnets, route tables, internet gateway, and security groups.

A VPC with CIDR block 10.0.0.0/16 is created.
An Internet Gateway is attached to the VPC for internet access.
A Route Table is defined with a route to the Internet Gateway.
A Subnet 10.0.1.0/24 is created in availability zone us-east-1a.
The Route Table is associated with the Subnet.
A Security Group is created for controlling inbound and outbound traffic to instances within the VPC.

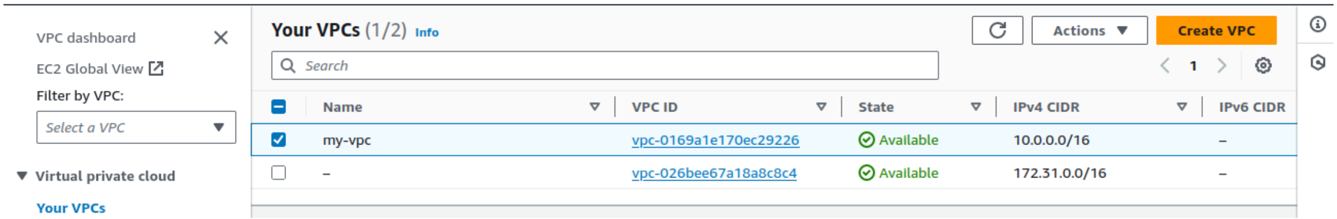
```
provider "aws" {
  region = "us-east-1" # N. Virginia region
}

# Create VPC
resource "aws_vpc" "my_vpc" {
  cidr_block = "10.0.0.0/16"
  enable_dns_support = true
  enable_dns_hostnames = true

  tags = {
    Name = "my-vpc"
  }
}

# Create Internet Gateway
resource "aws_internet_gateway" "my_igw" {
  vpc_id = aws_vpc.my_vpc.id

  tags = {
    Name = "my-igw"
  }
}

# Create Route Table
resource "aws_route_table" "my_route_table" {
  vpc_id = aws_vpc.my_vpc.id

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

  tags = {
    Name = "my-route-table"
  }
}

# Create Subnet
resource "aws_subnet" "my_subnet" {
  vpc_id     = aws_vpc.my_vpc.id
  cidr_block = "10.0.1.0/24"
  availability_zone = "us-east-1a"

  tags = {
    Name = "my-subnet"
  }
}

# Associate Route Table with Subnet
resource "aws_route_table_association" "subnet_association" {
  subnet_id      = aws_subnet.my_subnet.id
  route_table_id = aws_route_table.my_route_table.id
}

# Create Security Group
resource "aws_security_group" "my_security_group" {
  vpc_id = aws_vpc.my_vpc.id

  // Define your security group rules here

  tags = {t
    Name = "my-security-group"
  }
}
```



```
provider "aws" {
  region = "us-east-1" # N. Virginia region
}

# Create VPC
resource "aws_vpc" "my_vpc" {
  cidr_block = "10.0.0.0/16"
  enable_dns_support = true
  enable_dns_hostnames = true

  tags = {
    Name = "my-vpc"
  }
}

# Create Internet Gateway
resource "aws_internet_gateway" "my_igw" {
  vpc_id = aws_vpc.my_vpc.id

  tags = {
    Name = "my-igw"
  }
```



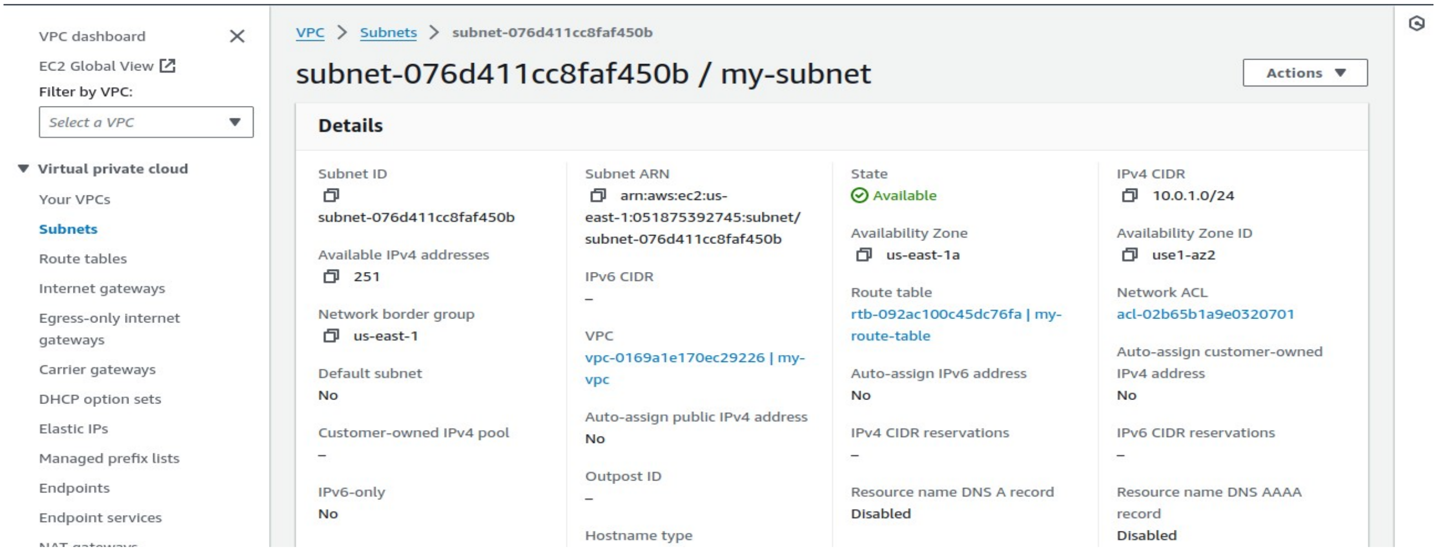
```
aws_route_table.my_route_table: Creation complete after 3s [id=rtb-092ac100c45dc76fa]
aws_route_table_association.subnet_association: Creating...
aws_route_table_association.subnet_association: Creation complete after 1s [id=rtbassoc
-04edeb71f578bdb1b]

Apply complete! Resources: 6 added, 0 changed, 0 destroyed.
subby@subby-ubuntu:~/Desktop/terraform$ 
```



Your VPCs (1/2) Info

	Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR
☑	my-vpc	vpc-0169a1e170ec29226	✓ Available	10.0.0.0/16	–
□	–	vpc-026bee67a18a8c8c4	✓ Available	172.31.0.0/16	–

SUBNET



VPC > Subnets > subnet-076d411cc8faf450b

subnet-076d411cc8faf450b / my-subnet

Details

Subnet ID
subnet-076d411cc8faf450b

Available IPv4 addresses
251

Network border group
us-east-1

Default subnet
No

Customer-owned IPv4 pool
–

IPv6-only
No

Subnet ARN
arn:aws:ec2:us-east-1:051875392745:subnet/subnet-076d411cc8faf450b

IPv6 CIDR
–

VPC
vpc-0169a1e170ec29226 | my-vpc

Auto-assign public IPv4 address
No

Outpost ID
–

Hostname type

State
✓ Available

Availability Zone
us-east-1a

Route table
rtb-092ac100c45dc76fa | my-route-table

Auto-assign IPv6 address
No

IPv4 CIDR reservations
–

Resource name DNS A record
Disabled

IPv4 CIDR
10.0.1.0/24

Availability Zone ID
use1-az2

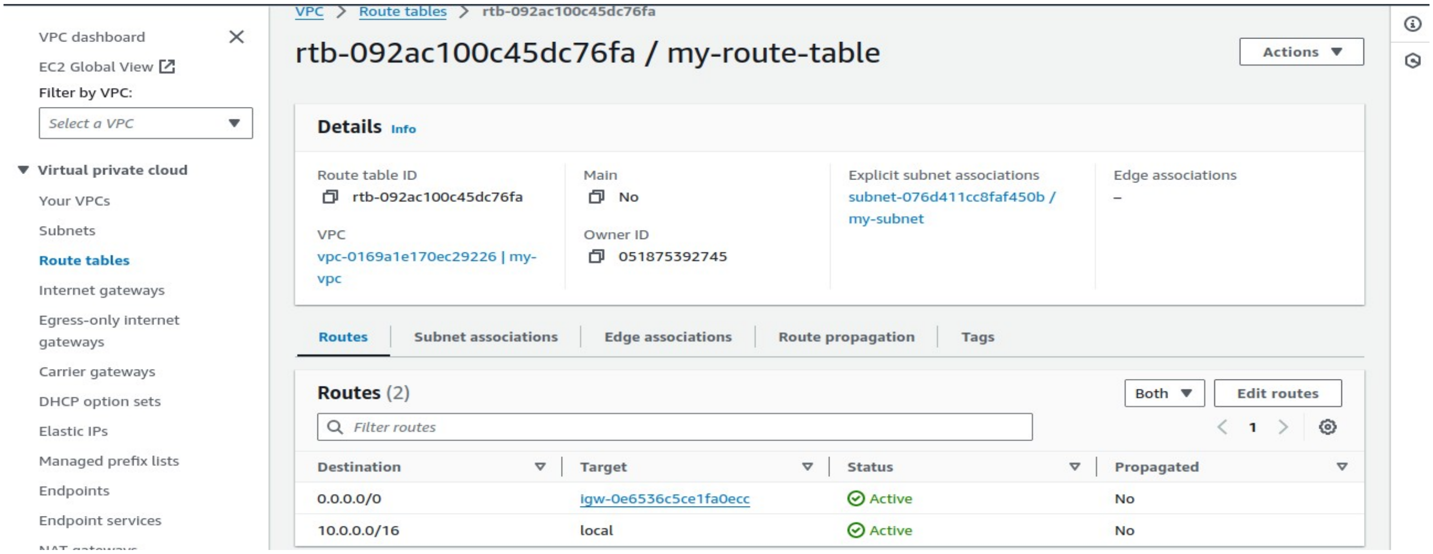
Network ACL
acl-02b65b1a9e0320701

Auto-assign customer-owned IPv4 address
No

IPv6 CIDR reservations
–

Resource name DNS AAAA record
Disabled

ROUTE-TABLE



VPC > Route tables > rtb-092ac100c45dc76fa

rtb-092ac100c45dc76fa / my-route-table

Details Info

Route table ID
rtb-092ac100c45dc76fa

VPC
vpc-0169a1e170ec29226 | my-vpc

Main
No

Owner ID
051875392745

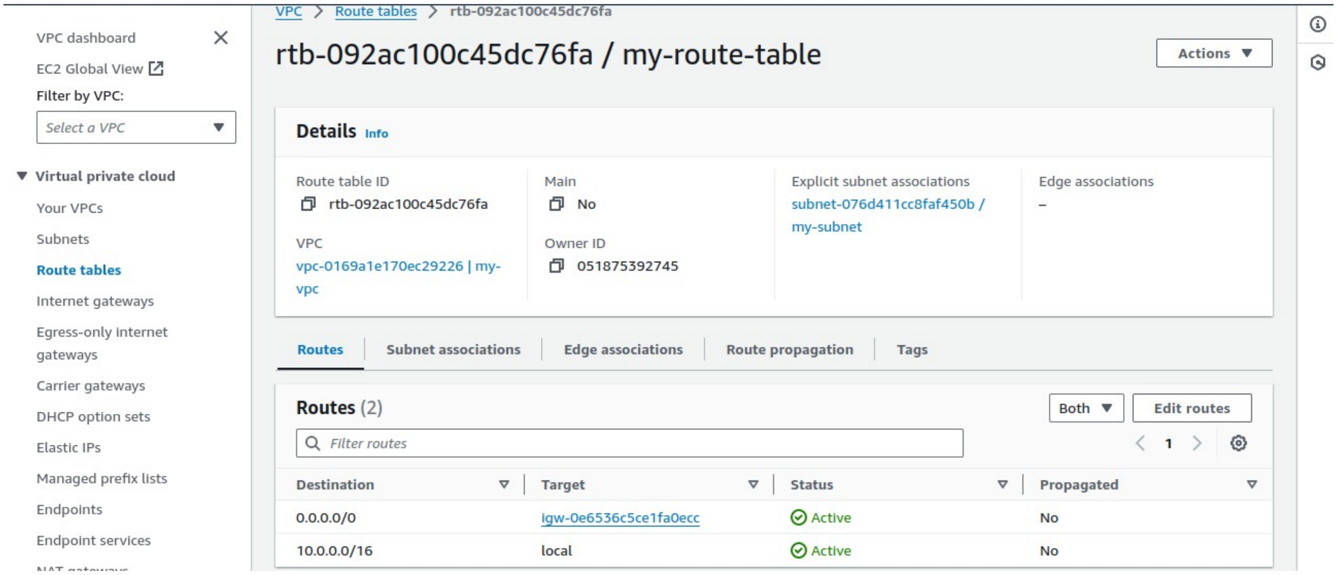
Explicit subnet associations
subnet-076d411cc8faf450b / my-subnet

Edge associations
–

Routes (2)

Destination	Target	Status	Propagated
0.0.0.0/0	igw-0e6536c5ce1fa0ecc	✓ Active	No
10.0.0.0/16	local	✓ Active	No

SUBNET ASSOCIATION



VPC > Route tables > rtb-092ac100c45dc76fa

rtb-092ac100c45dc76fa / my-route-table

Details Info

Route table ID
rtb-092ac100c45dc76fa

VPC
vpc-0169a1e170ec29226 | my-vpc

Main
No

Owner ID
051875392745

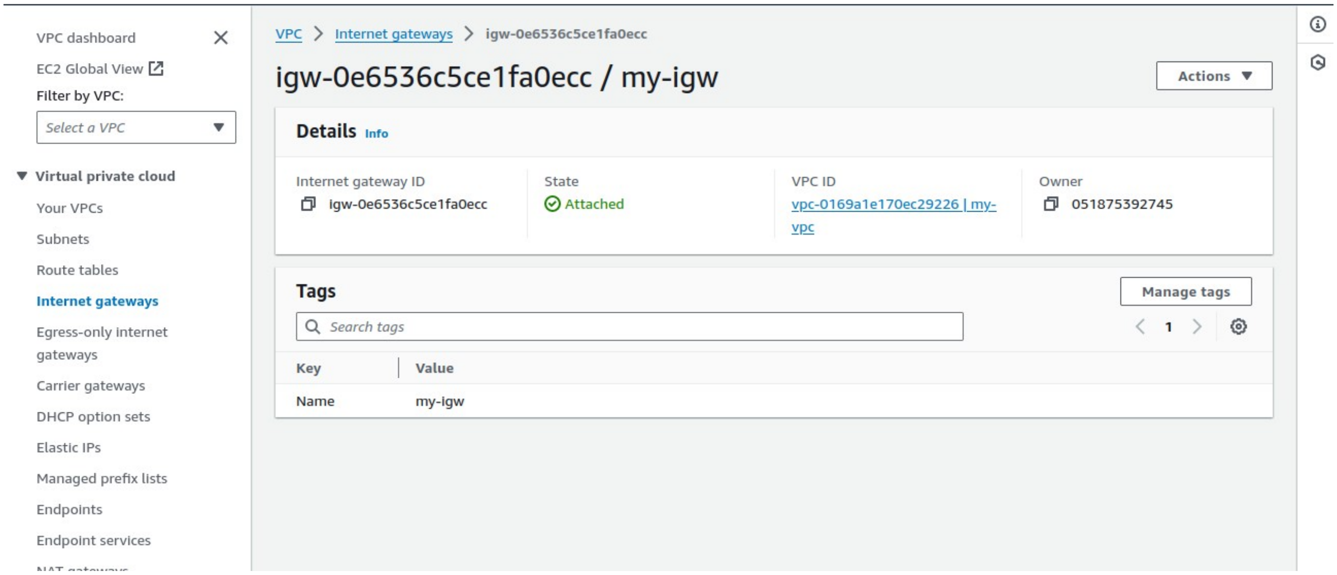
Explicit subnet associations
subnet-076d411cc8faf450b / my-subnet

Edge associations
–

Routes (2)

Destination	Target	Status	Propagated
0.0.0.0/0	igw-0e6536c5ce1fa0ecc	✓ Active	No
10.0.0.0/16	local	✓ Active	No

IGW



igw-0e6536c5ce1fa0ecc / my-igw

Details

Internet gateway ID: igw-0e6536c5ce1fa0ecc

State: Attached

VPC ID: vpc-0169a1e170ec29226 | my-vpc

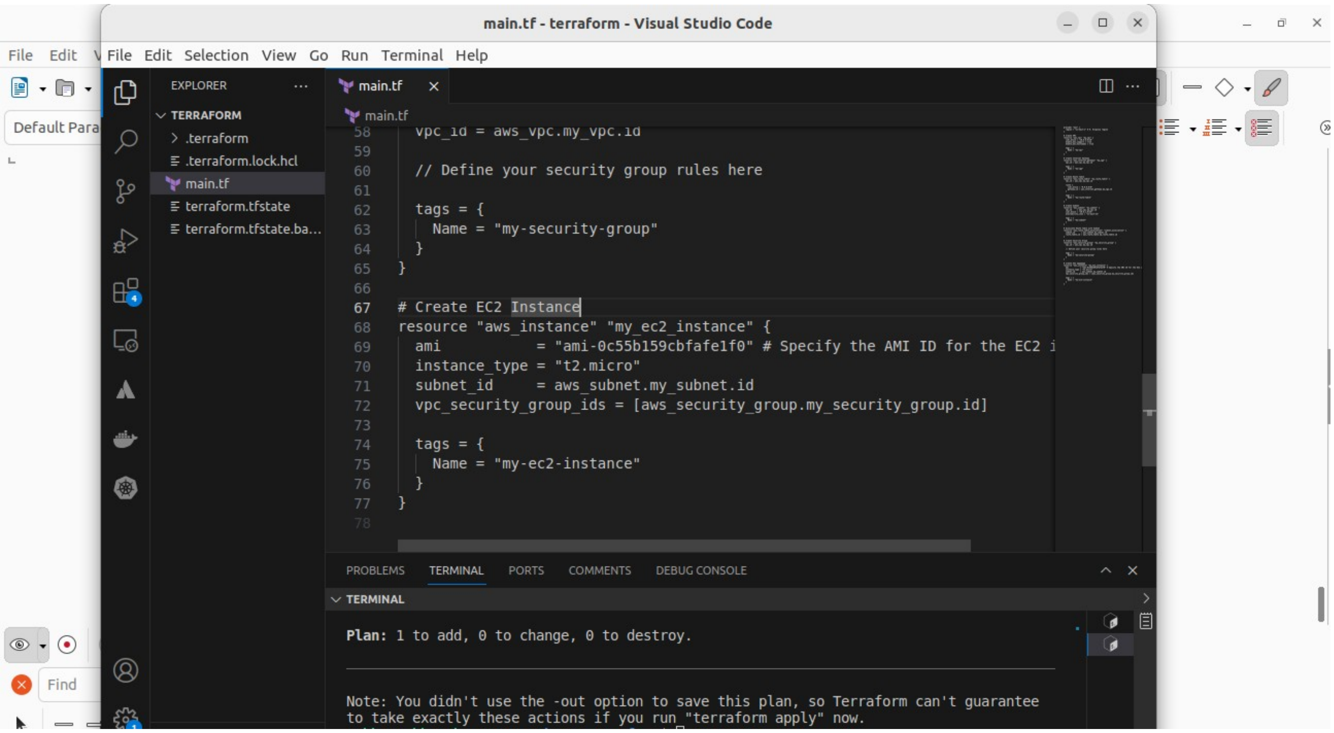
Owner: 051875392745

Tags

Key	Value
Name	my-igw

Deploy an EC2 instance inside the VPC

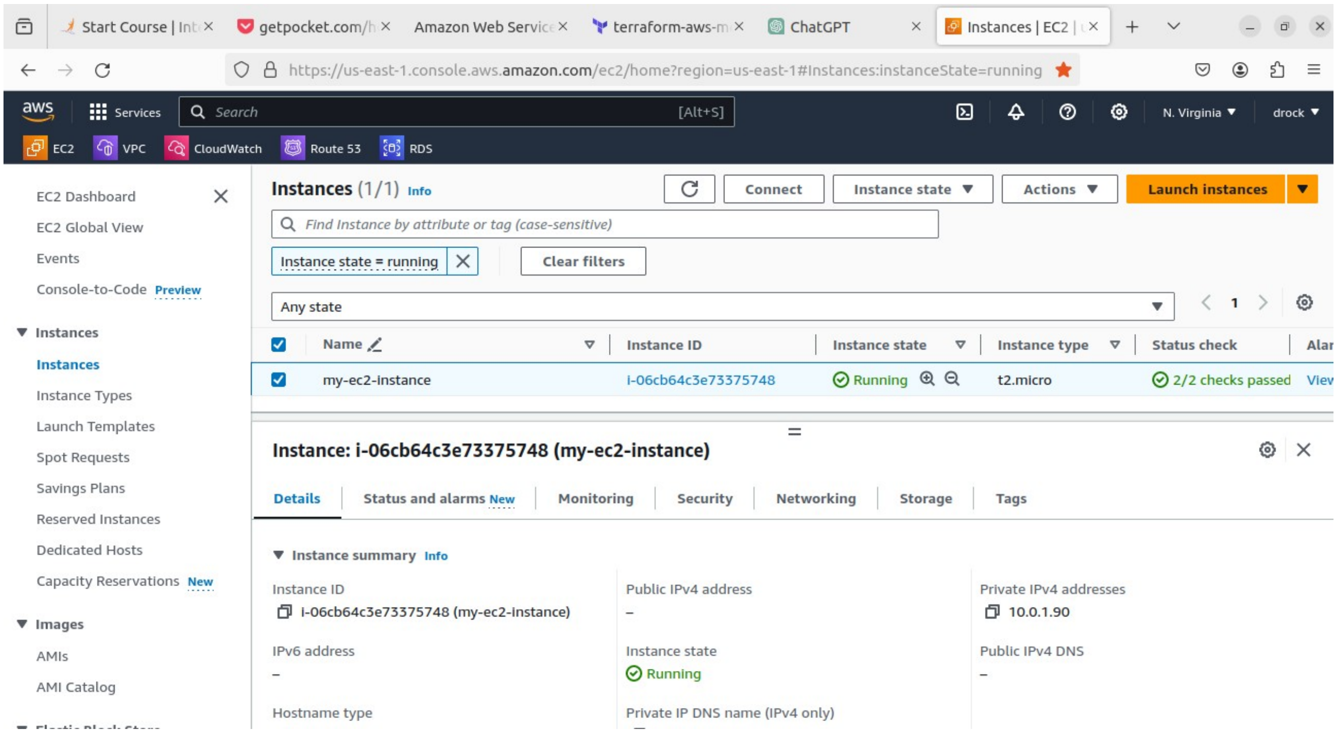
```
provider "aws" {
  region = "us-east-1" # N. Virginia region
}

# Create VPC
resource "aws_vpc" "my_vpc" {
  cidr_block = "10.0.0.0/16"
  enable_dns_support = true
  enable_dns_hostnames = true

  tags = {
    Name = "my-vpc"
  }
}

# Create Internet Gateway
resource "aws_internet_gateway" "my_igw" {
  vpc_id = aws_vpc.my_vpc.id

  tags = {
    Name = "my-igw"
  }
}

# Create Route Table
resource "aws_route_table" "my_route_table" {
  vpc_id = aws_vpc.my_vpc.id

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

  tags = {
    Name = "my-route-table"
  }
}

# Create Subnet
resource "aws_subnet" "my_subnet" {
  vpc_id     = aws_vpc.my_vpc.id
  cidr_block = "10.0.1.0/24"
  availability_zone = "us-east-1a"

  tags = {
    Name = "my-subnet"
  }
}

# Associate Route Table with Subnet
resource "aws_route_table_association" "subnet_association" {
  subnet_id      = aws_subnet.my_subnet.id
  route_table_id = aws_route_table.my_route_table.id
}

# Create Security Group
resource "aws_security_group" "my_security_group" {
  vpc_id = aws_vpc.my_vpc.id

  // Define your security group rules here

  tags = {
    Name = "my-security-group"
  }
}

# Data block to fetch most recent Amazon Linux 2 AMI
data "aws_ami" "virginia_ami" {
  most_recent = true

  filter {
    name   = "name"
    values = ["amzn2-ami-hvm-*"]
  }

  filter {
    name   = "virtualization-type"
    values = ["hvm"]
  }
}

# Create EC2 Instance with the fetched AMI
resource "aws_instance" "my_ec2_instance" {
  ami           = data.aws_ami.virginia_ami.id
  instance_type = "t2.micro"
  subnet_id     = aws_subnet.my_subnet.id
  vpc_security_group_ids = [aws_security_group.my_security_group.id]

  tags = {
    Name = "my-ec2-instance"
  }
}
```


main.tf - terraform - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
TERRAFORM
.terraform
.terraform.lock.hcl
main.tf
terraform.tfstate
terraform.tfstate.ba...
PROBLEMS TERMINAL PORTS COMMENTS DEBUG CONSOLE
TERMINAL

```
  vpc_id = aws_vpc.my_vpc.id

  // Define your security group rules here

  tags = {
    Name = "my-security-group"
  }
}

# Create EC2 Instance
resource "aws_instance" "my_ec2_instance" {
  ami           = "ami-0c55b159cbfafe1f0" # Specify the AMI ID for the EC2 i
  instance_type = "t2.micro"
  subnet_id     = aws_subnet.my_subnet.id
  vpc_security_group_ids = [aws_security_group.my_security_group.id]

  tags = {
    Name = "my-ec2-instance"
  }
}
```

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

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee
to take exactly these actions if you run "terraform apply" now.
```

NO PUBLIC IPV4



Instances (1/1) Info

Instance state = running

Any state

☑	Name	Instance ID	Instance state	Instance type	Status check	Alar
☑	my-ec2-instance	i-06cb64c3e73375748	Running	t2.micro	2/2 checks passed	View

Instance: i-06cb64c3e73375748 (my-ec2-instance)

Instance summary Info

Instance ID
i-06cb64c3e73375748 (my-ec2-instance)

Public IPv4 address
–

Private IPv4 addresses
10.0.1.90

IPv6 address
–

Instance state
Running

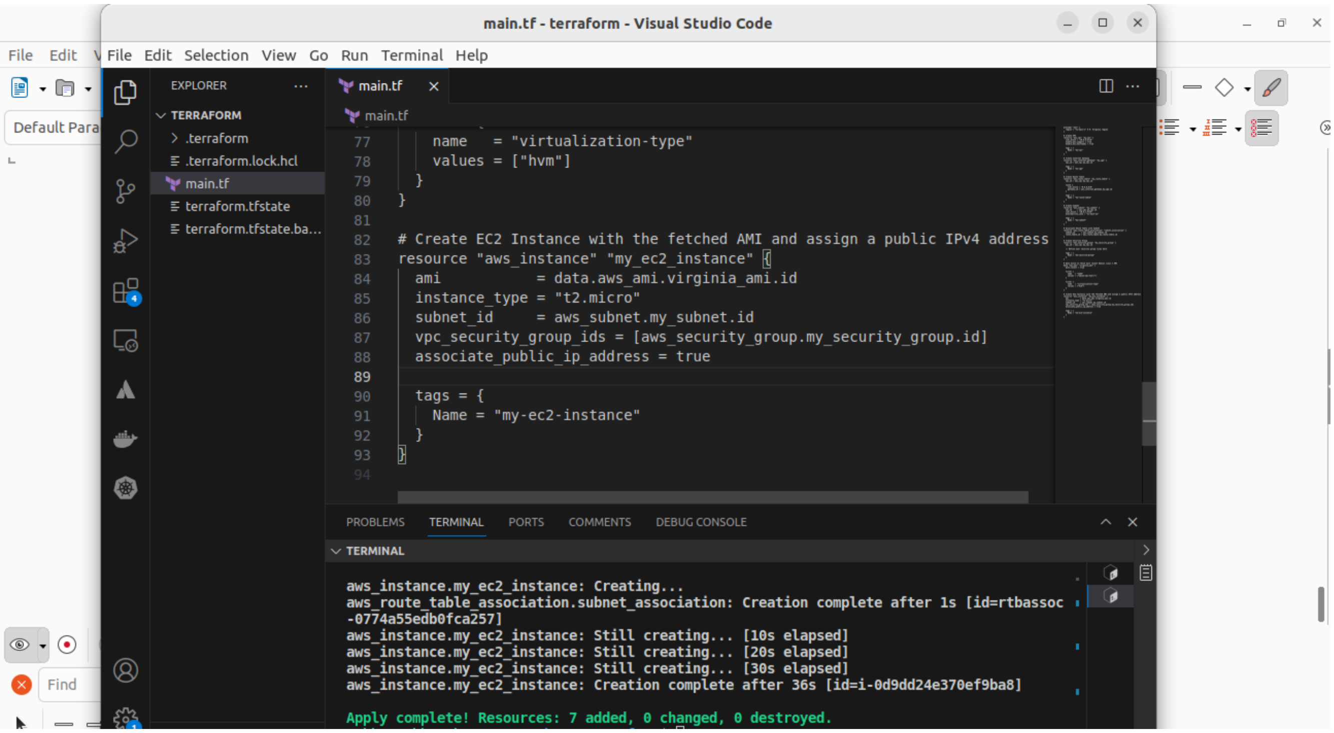
Public IPv4 DNS
–

Hostname type

Private IP DNS name (IPv4 only)

EC2 instance with a public IPv4 address

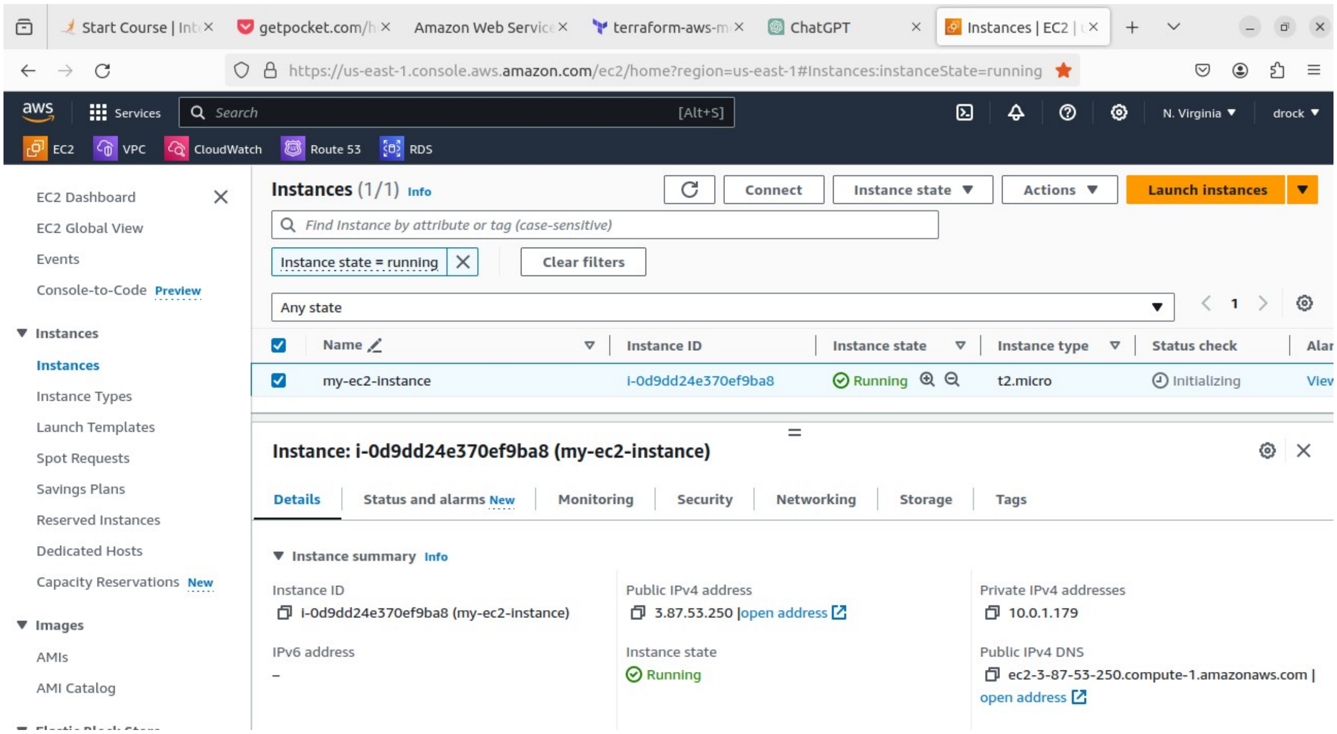
```
provider "aws" {
  region = "us-east-1" # N. Virginia region
}

# Create VPC
resource "aws_vpc" "my_vpc" {
  cidr_block = "10.0.0.0/16"
  enable_dns_support = true
  enable_dns_hostnames = true

  tags = {
    Name = "my-vpc"
  }
}

# Create Internet Gateway
resource "aws_internet_gateway" "my_igw" {
  vpc_id = aws_vpc.my_vpc.id

  tags = {
    Name = "my-igw"
  }
}

# Create Route Table
resource "aws_route_table" "my_route_table" {
  vpc_id = aws_vpc.my_vpc.id

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

  tags = {
    Name = "my-route-table"
  }
}

# Create Subnet
resource "aws_subnet" "my_subnet" {
  vpc_id     = aws_vpc.my_vpc.id
  cidr_block = "10.0.1.0/24"
  availability_zone = "us-east-1a"

  tags = {
    Name = "my-subnet"
  }
}

# Associate Route Table with Subnet
resource "aws_route_table_association" "subnet_association" {
  subnet_id      = aws_subnet.my_subnet.id
  route_table_id = aws_route_table.my_route_table.id
}

# Create Security Group
resource "aws_security_group" "my_security_group" {
  vpc_id = aws_vpc.my_vpc.id

  // Define your security group rules here

  tags = {
    Name = "my-security-group"
  }
}

# Data block to fetch most recent Amazon Linux 2 AMI
data "aws_ami" "virginia_ami" {
  most_recent = true

  filter {
    name   = "name"
    values = ["amzn2-ami-hvm-*"]
  }

  filter {
    name   = "virtualization-type"
    values = ["hvm"]
  }
}

# Create EC2 Instance with the fetched AMI and assign a public IPv4 address
resource "aws_instance" "my_ec2_instance" {
  ami           = data.aws_ami.virginia_ami.id
  instance_type = "t2.micro"
  subnet_id     = aws_subnet.my_subnet.id
  vpc_security_group_ids = [aws_security_group.my_security_group.id]
  associate_public_ip_address = true

  tags = {
    Name = "my-ec2-instance"
  }
}
```



main.tf - terraform - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
TERRAFORM
.terraform
.terraform.lock.hcl
main.tf
terraform.tfstate
terraform.tfstate.ba...
PROBLEMS TERMINAL PORTS COMMENTS DEBUG CONSOLE
Default Para
Find

```
    name   = "virtualization-type"
    values = ["hvm"]
  }
}

# Create EC2 Instance with the fetched AMI and assign a public IPv4 address
resource "aws_instance" "my_ec2_instance" {
  ami           = data.aws_ami.virginia_ami.id
  instance_type = "t2.micro"
  subnet_id     = aws_subnet.my_subnet.id
  vpc_security_group_ids = [aws_security_group.my_security_group.id]
  associate_public_ip_address = true

  tags = {
    Name = "my-ec2-instance"
  }
}
```

```
aws_instance.my_ec2_instance: Creating...
aws_route_table_association.subnet_association: Creation complete after 1s [id=rtbassoc
-0774a55edb0fca257]
aws_instance.my_ec2_instance: Still creating... [10s elapsed]
aws_instance.my_ec2_instance: Still creating... [20s elapsed]
aws_instance.my_ec2_instance: Still creating... [30s elapsed]
aws_instance.my_ec2_instance: Creation complete after 36s [id=i-0d9dd24e370ef9ba8]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.
```



Instances (1/1) Info

Instance state = running

Any state

☑	Name	Instance ID	Instance state	Instance type	Status check	Alar
☑	my-ec2-instance	i-0d9dd24e370ef9ba8	Running	t2.micro	Initializing	View

Instance: i-0d9dd24e370ef9ba8 (my-ec2-instance)

Instance summary Info

Instance ID
i-0d9dd24e370ef9ba8 (my-ec2-instance)

Public IPv4 address
3.87.53.250 |open address

Private IPv4 addresses
10.0.1.179

IPv6 address
–

Instance state
Running

Public IPv4 DNS
ec2-3-87-53-250.compute-1.amazonaws.com | open address