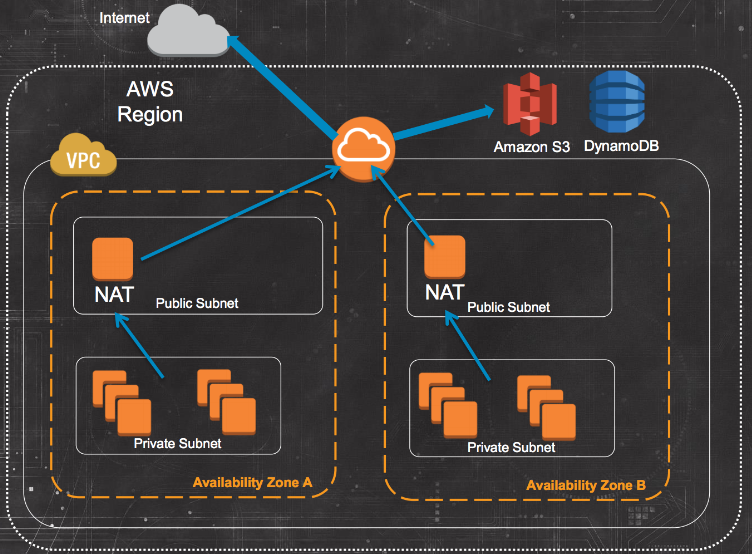
**TERRAFORM MULTI-FILES**

**AWS VPC-Subnet(Public/Private)-Internet Gateway-NAT Gateway**



**Terraform files**

|  |
| --- |
| **Mymain.tf**  provider "aws"{  access\_key = "AKIAS7QWBE2WN5N5WTC4"  secret\_key="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"  region="us-east-2"  } |

|  |
| --- |
| **vpc\_demo.tf**  resource “aws\_vpc” “vpc\_demo” {  cidr\_block = var.cidr  instance\_tenancy = var.instance\_tenancy  enable\_dns\_hostnames = var.enable\_dns\_hostnames  enable\_dns\_support = var.enable\_dns\_support  enable\_classiclink = var.enable\_classiclink  tags = {  Name = “var.tags”  }  } |

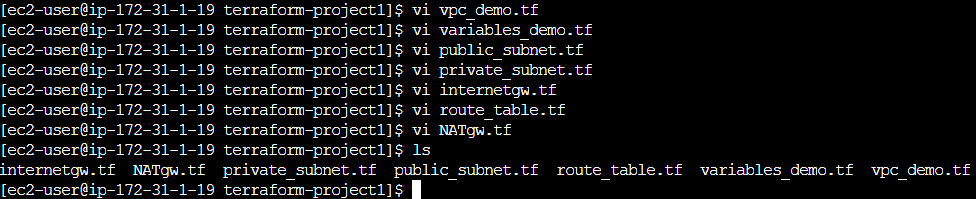
|  |
| --- |
| **Variables\_demo.tf**  variable “cidr” {  description = “the cidr block for the VPC. default value is a valid cidr, but not acceptable by aws and should be overridden”  type = string  default = “10.0.0.0/16”  }  variable “instance\_tenancy” {  description = “A tenancy option for instances launched into the VPC”  type = string  default = “default”  }  variable “enable\_dns\_hostnames” {  description = “should be true to enable DNS hostnames in the VPC”  type = bool  default = true  }  variable “enable\_dns\_support” {  description = “should be true to enable DNS support in the VPC”  type = bool  default = true  }  variable “enable\_classiclink” {  description = “should be true to enable classiclink for the VPC. Only valid in regions and accounts that support EC2Classic.”  type = bool  default = false  } |

|  |
| --- |
| **public\_subnet.tf**  resource “aws\_subnet” “public\_1” {  vpc\_id = aws\_vpc.vpc\_demo.id  map\_public\_ip\_on\_launch = true  cidr\_block = “10.0.1.0/24”  tags = {  Name = “public\_1-demo”  }  }  resource “aws\_subnet” “public\_2” {  vpc\_id = aws\_vpc.vpc\_demo.id  map\_public\_ip\_on\_launch = true  cidr\_block = “10.0.2.0/24”  tags = {  Name = “public\_2-demo”  } |
| **private\_subnet.tf**  resource “aws\_subnet” “private\_1” {  vpc\_id = aws\_vpc.vpc\_demo.id  map\_public\_ip\_on\_launch = false  cidr\_block = “10.0.3.0/24”  tags = {  Name = “private\_1-demo”  }  }  resource “aws\_subnet” “private\_2” {  vpc\_id = aws\_vpc.vpc\_demo.id  map\_public\_ip\_on\_launch = false  cidr\_block = “10.0.4.0/24”  tags = {  Name = “private\_2-demo”  }  } |

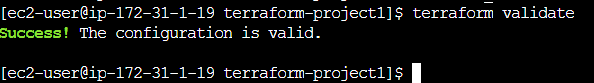
|  |
| --- |
| **Internetgw.tf**  resource “aws\_internet\_gateway” “gw” {  vpc\_id = “${aws\_vpc.vpc\_demo.id}”  tags ={  Name = “internet-gateway-demo”  }  } |

|  |
| --- |
| **route\_table.tf**  resource “aws\_route\_table” “route-public” {  vpc\_id = “${aws\_vpc.vpc\_demo.id}”  route {  cidr\_block = “10.0.0.0/0”  gateway\_id = “${aws\_internet\_gateway.gw.id}”  }  tags = {  Name = public-route-table.demo  }  }  resource “aws\_route\_table\_association” “public\_1” {  subnet\_id = “${aws\_subnet.public\_1.id}”  route\_table\_id = “${aws\_route\_table.route-public.id}”  }  resource “aws\_route\_table\_association” “public\_2” {  subnet\_id = “${aws\_subnet.public\_2.id}”  route\_table\_id = “${aws\_route\_table.route-public.id}”  } |
| **NATgateway.tf**  resource “aws\_eip” “nat” {  vpc = true  }  resource “aws\_nat\_gateway” “nat\_gw” {  allocation\_id = “${aws.eip.nat.id}”  subnet\_id = “{aws\_subnet.public\_1.id}”  depends\_on =[“aws\_internet\_gateway.gw”]  }  Resource “aws\_route\_table” “route-private” {  vpc\_id = “${aws\_vpc.vpc\_demo.id}”  route {  cidr\_block = “10.0.0.0/0”  gateway\_id = “${aws\_internet\_gateway.gw.id}”  }  tags = {  Name = private-route-table.demo  }  }  resource “aws\_route\_table\_association” “private\_1” {  subnet\_id = “${aws\_subnet.private\_1.id}”  route\_table\_id = “${aws\_route\_table.route-private.id}”  }  resource “aws\_route\_table\_association” “private\_2” {  subnet\_id = “${aws\_subnet.private\_2.id}”  route\_table\_id = “${aws\_route\_table.route-private.id}”  } |

Following files has been created.

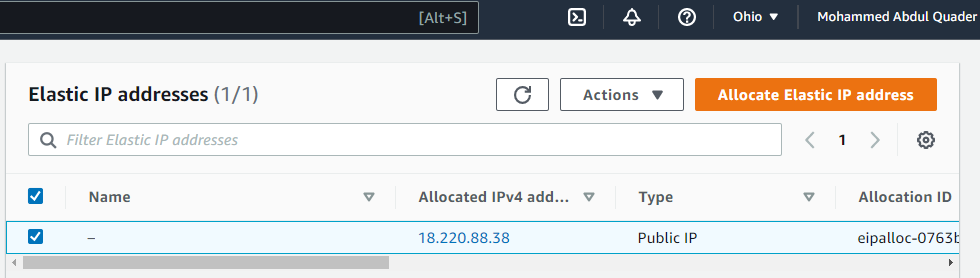


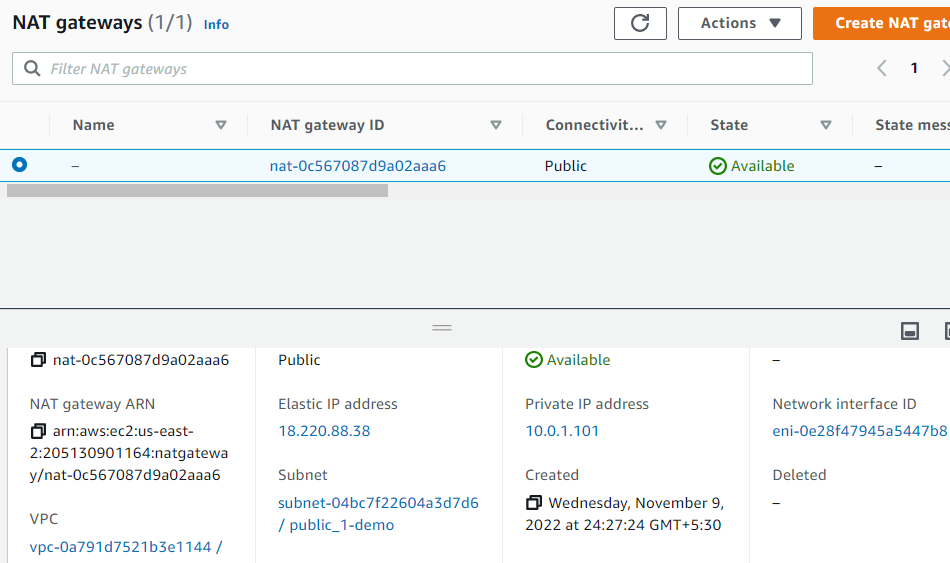
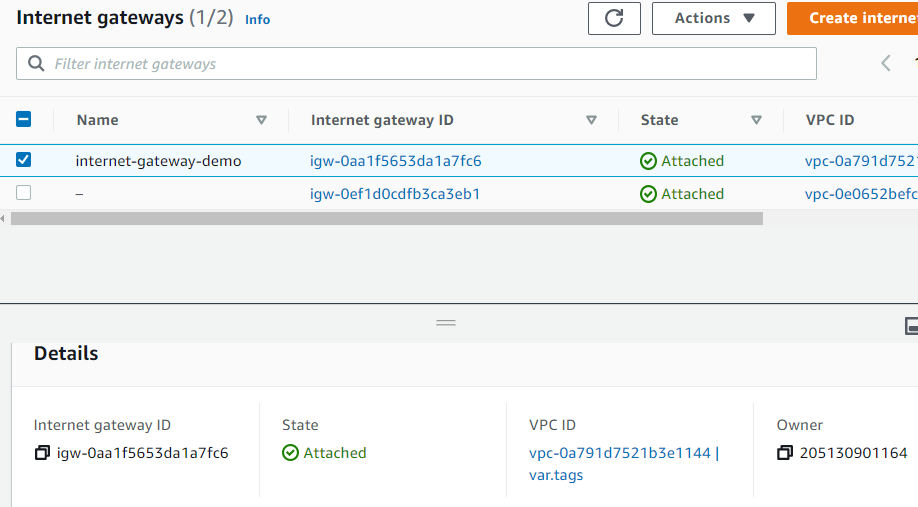
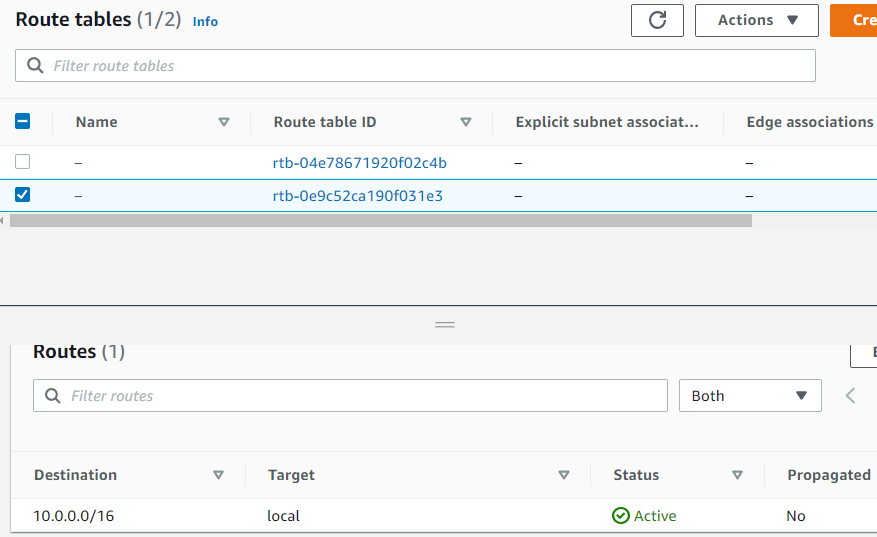
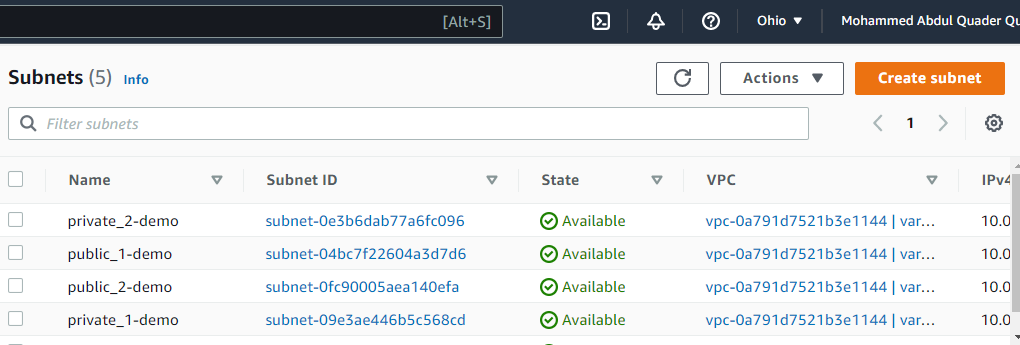
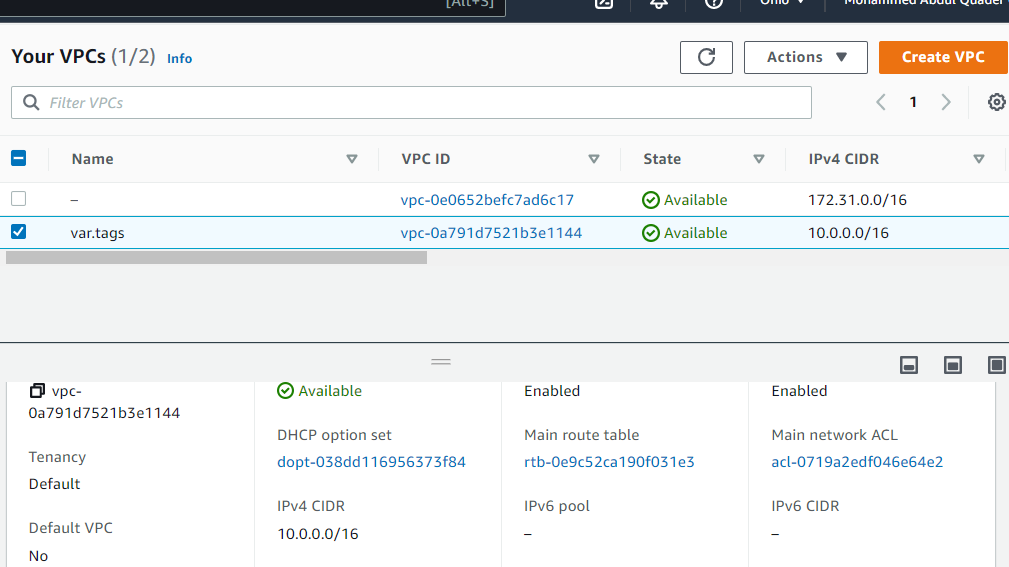
Run **$terraform validate** command to validate the syntax



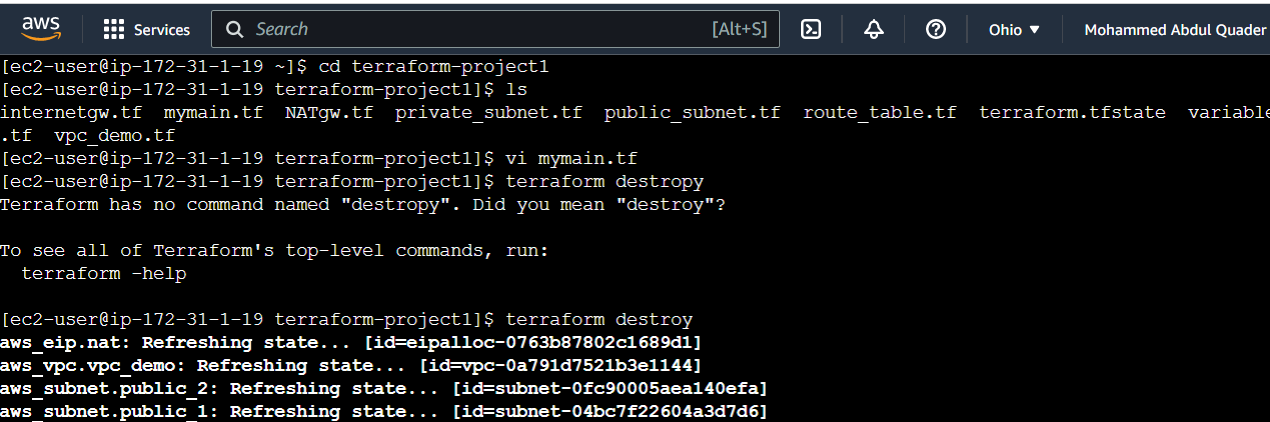
Run **$terraform plan** command

Run **$terraform apply** command and check the result





Run **$terraform destroy** command to destroy the infrastructure.

****

