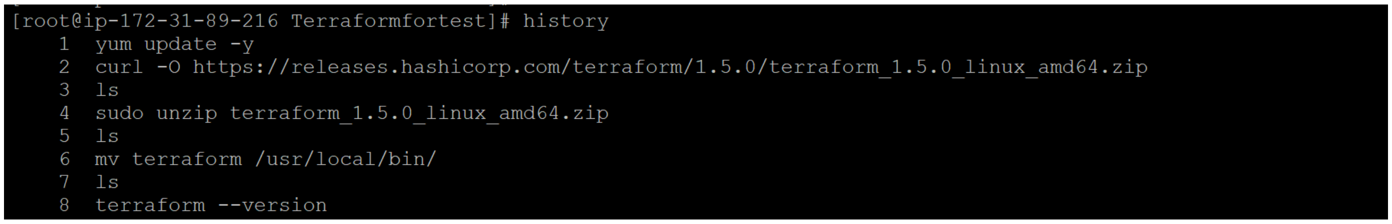
Terraform documentation for spinning Nginx Server

Kiran Reddy Bokkala:

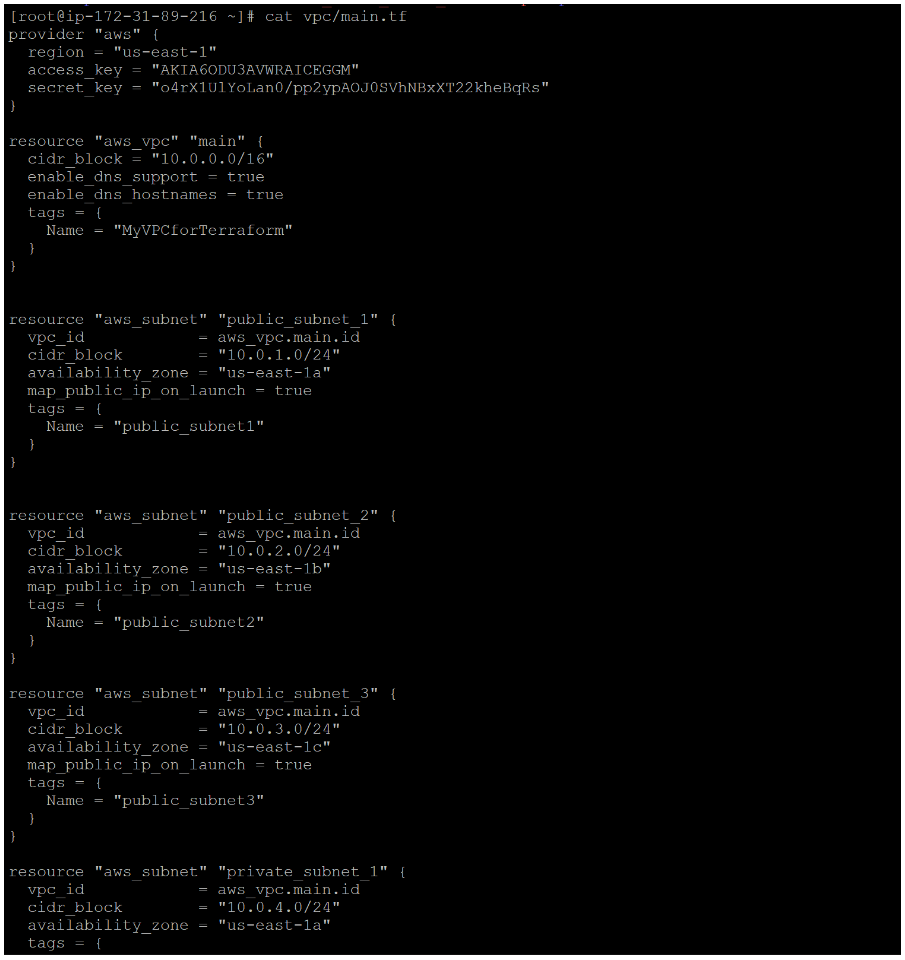
1. Created the EC2 virtual machine and installed the Terraform in it as shown below.



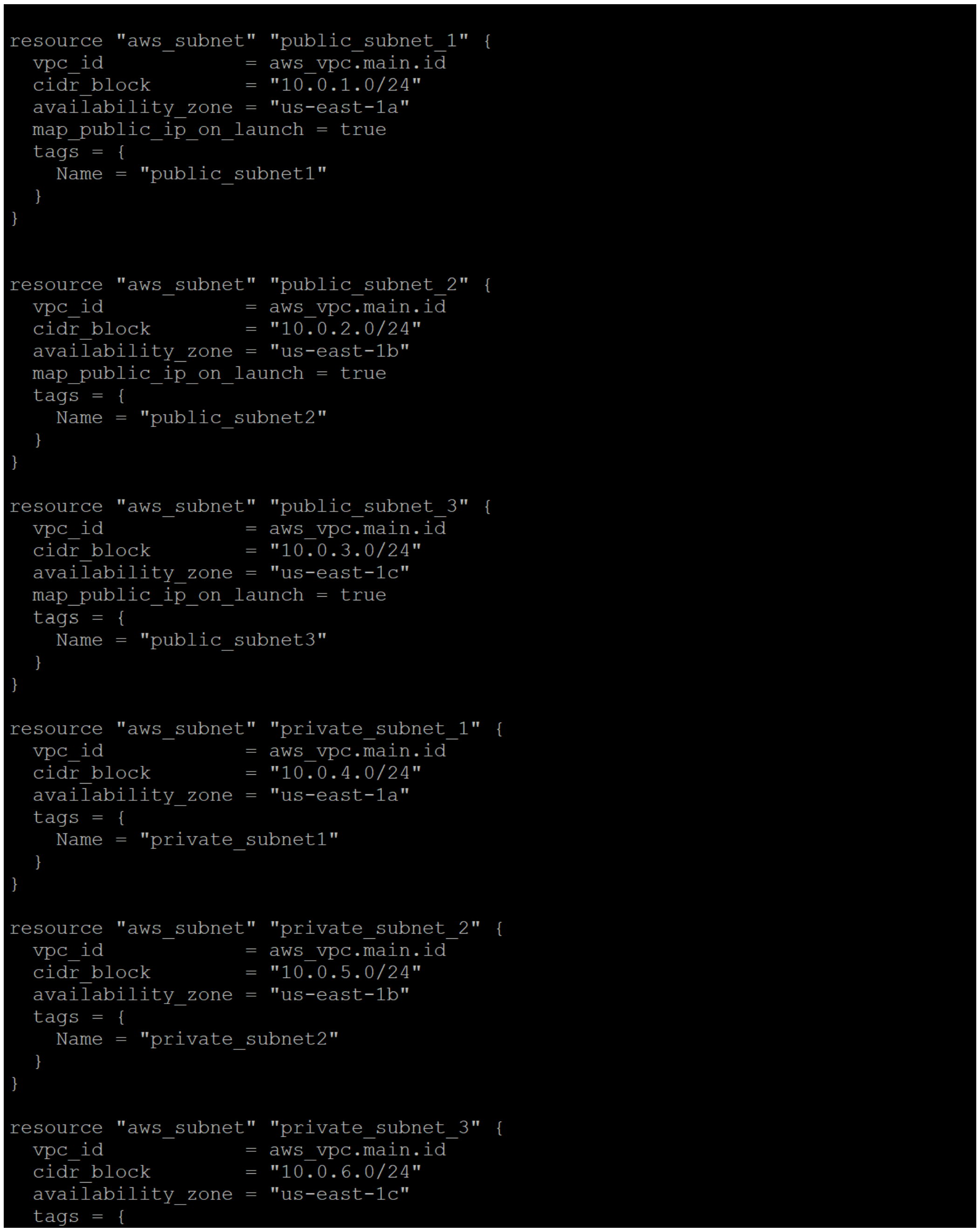
```
[root@ip-172-31-89-216 Terraformfortest]# history
    1  yum update -y
    2  curl -O https://releases.hashicorp.com/terraform/1.5.0/terraform_1.5.0_linux_amd64.zip
    3  ls
    4  sudo unzip terraform_1.5.0_linux_amd64.zip
    5  ls
    6  mv terraform /usr/local/bin/
    7  ls
    8  terraform --version
```

2. Created a directory VPC and EC2

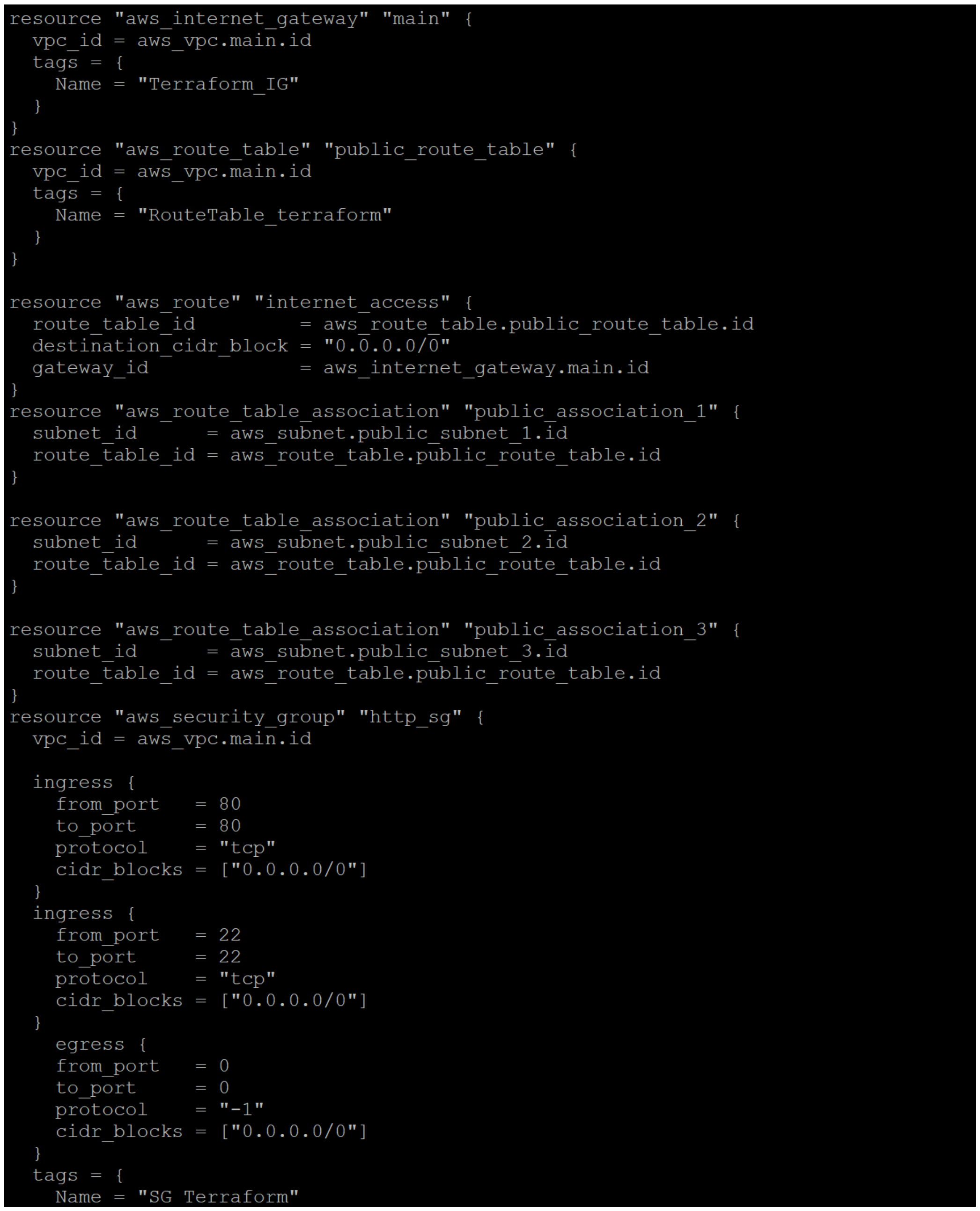
In the VPC directory created a main.tf file where it has all the code related to creating VPC, subnets, internet gateways, routes, security groups



```
[root@ip-172-31-89-216 ~]# cat vpc/main.tf
provider "aws" {
  region = "us-east-1"
  access_key = "AKIA6ODU3AVWRAICEGGM"
  secret_key = "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"
}

resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16"
  enable_dns_support = true
  enable_dns_hostnames = true
  tags = {
    Name = "MyVPCforTerraform"
  }
}


resource "aws_subnet" "public_subnet_1" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.1.0/24"
  availability_zone = "us-east-1a"
  map_public_ip_on_launch = true
  tags = {
    Name = "public_subnet1"
  }
}


resource "aws_subnet" "public_subnet_2" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.2.0/24"
  availability_zone = "us-east-1b"
  map_public_ip_on_launch = true
  tags = {
    Name = "public_subnet2"
  }
}

resource "aws_subnet" "public_subnet_3" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.3.0/24"
  availability_zone = "us-east-1c"
  map_public_ip_on_launch = true
  tags = {
    Name = "public_subnet3"
  }
}

resource "aws_subnet" "private_subnet_1" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.4.0/24"
  availability_zone = "us-east-1a"
  tags = {
```

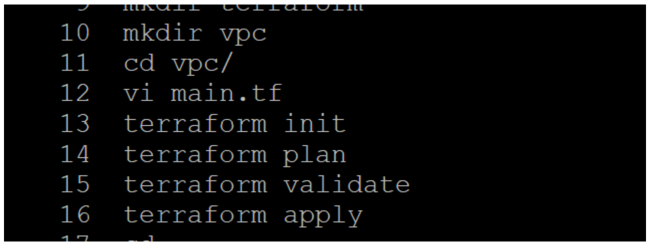


```
resource "aws_subnet" "public_subnet_1" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.1.0/24"
  availability_zone = "us-east-1a"
  map_public_ip_on_launch = true
  tags = {
    Name = "public_subnet1"
  }
}


resource "aws_subnet" "public_subnet_2" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.2.0/24"
  availability_zone = "us-east-1b"
  map_public_ip_on_launch = true
  tags = {
    Name = "public_subnet2"
  }
}

resource "aws_subnet" "public_subnet_3" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.3.0/24"
  availability_zone = "us-east-1c"
  map_public_ip_on_launch = true
  tags = {
    Name = "public_subnet3"
  }
}

resource "aws_subnet" "private_subnet_1" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.4.0/24"
  availability_zone = "us-east-1a"
  tags = {
    Name = "private_subnet1"
  }
}

resource "aws_subnet" "private_subnet_2" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.5.0/24"
  availability_zone = "us-east-1b"
  tags = {
    Name = "private_subnet2"
  }
}

resource "aws_subnet" "private_subnet_3" {
  vpc_id            = aws_vpc.main.id
  cidr_block        = "10.0.6.0/24"
  availability_zone = "us-east-1c"
  tags = {
```



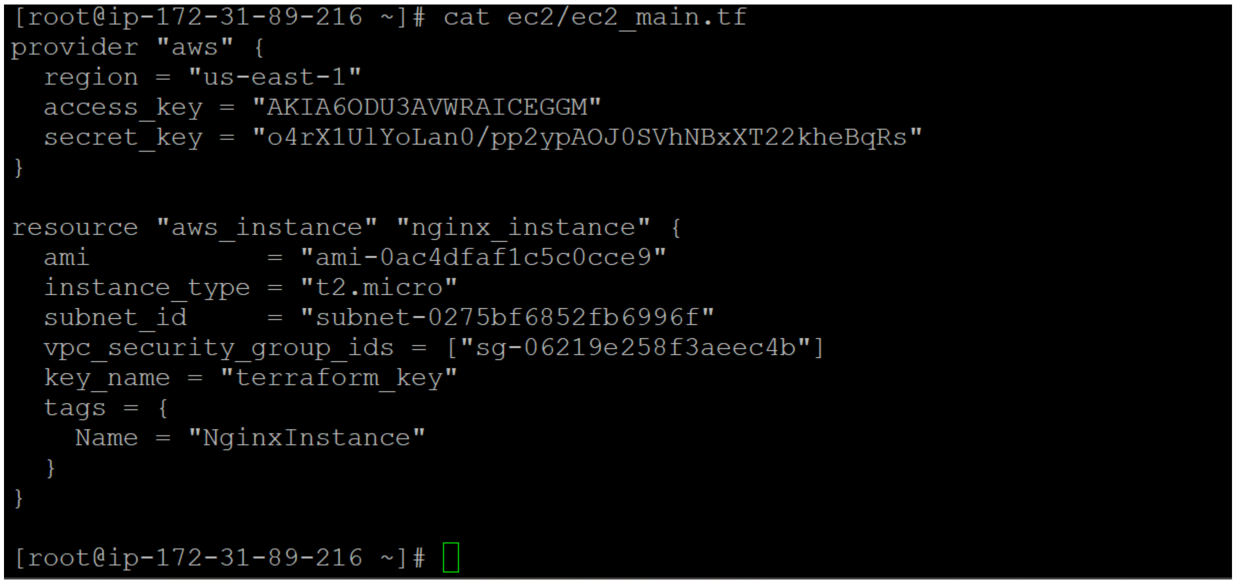
```
resource "aws_internet_gateway" "main" {
  vpc_id = aws_vpc.main.id
  tags = {
    Name = "Terraform_IG"
  }
}
resource "aws_route_table" "public_route_table" {
  vpc_id = aws_vpc.main.id
  tags = {
    Name = "RouteTable_terraform"
  }
}

resource "aws_route" "internet_access" {
  route_table_id         = aws_route_table.public_route_table.id
  destination_cidr_block = "0.0.0.0/0"
  gateway_id             = aws_internet_gateway.main.id
}
resource "aws_route_table_association" "public_association_1" {
  subnet_id      = aws_subnet.public_subnet_1.id
  route_table_id = aws_route_table.public_route_table.id
}

resource "aws_route_table_association" "public_association_2" {
  subnet_id      = aws_subnet.public_subnet_2.id
  route_table_id = aws_route_table.public_route_table.id
}

resource "aws_route_table_association" "public_association_3" {
  subnet_id      = aws_subnet.public_subnet_3.id
  route_table_id = aws_route_table.public_route_table.id
}
resource "aws_security_group" "http_sg" {
  vpc_id = aws_vpc.main.id

  ingress {
    from_port   = 80
    to_port     = 80
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
    egress {
    from_port   = 0
    to_port     = 0
    protocol    = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
  tags = {
    Name = "SG_Terraform"
```

3. Then executed the main.tf code as shown below for provisioning the Network.



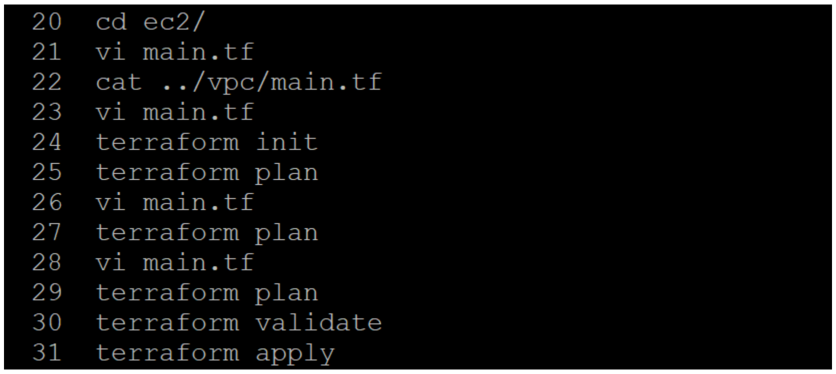
```
   10  mkdir vpc
   11  cd vpc/
   12  vi main.tf
   13  terraform init
   14  terraform plan
   15  terraform validate
   16  terraform apply
```

4. Now created the terraform code for provisioning Nginx EC2 server in ec2 folder.



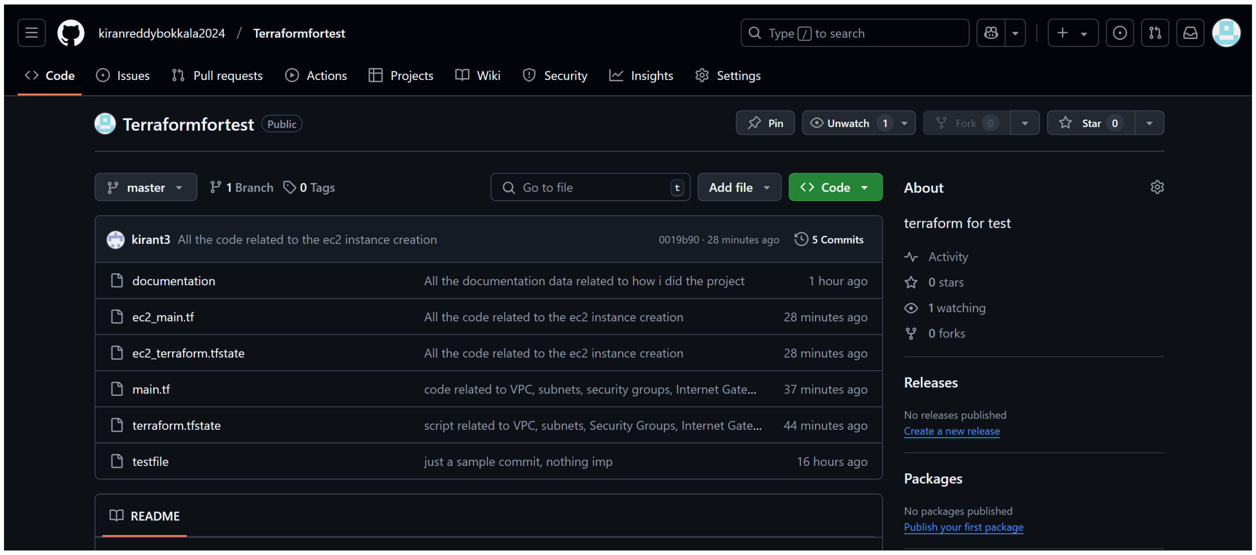
```
[root@ip-172-31-89-216 ~]# cat ec2/ec2_main.tf
provider "aws" {
  region = "us-east-1"
  access_key = "AKIA6ODU3AVWRAICEGGM"
  secret_key = "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"
}

resource "aws_instance" "nginx_instance" {
  ami             = "ami-0ac4dfaf1c5c0cce9"
  instance_type = "t2.micro"
  subnet_id     = "subnet-0275bf6852fb6996f"
  vpc_security_group_ids = ["sg-06219e258f3aeec4b"]
  key_name = "terraform_key"
  tags = {
    Name = "NginxInstance"
  }
}

[root@ip-172-31-89-216 ~]# 
```

And executed the code as shown below.



```
 20  cd ec2/
 21  vi main.tf
 22  cat ../vpc/main.tf
 23  vi main.tf
 24  terraform init
 25  terraform plan
 26  vi main.tf
 27  terraform plan
 28  vi main.tf
 29  terraform plan
 30  terraform validate
 31  terraform apply
```

5. Now created a GitHub account with a public repository named Terraformfortest as shown below.



kiranreddybokkala2024 / Terraformfortest

Code · Issues · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

Terraformfortest Public

Pin · Unwatch 1 · Fork 0 · Star 0

master · 1 Branch · 0 Tags · Go to file · Add file · Code

kirant3 All the code related to the ec2 instance creation · 0019b90 · 28 minutes ago · 5 Commits

documentation	All the documentation data related to how i did the project	1 hour ago
ec2_main.tf	All the code related to the ec2 instance creation	28 minutes ago
ec2_terraform.tfstate	All the code related to the ec2 instance creation	28 minutes ago
main.tf	code related to VPC, subnets, security groups, Internet Gate...	37 minutes ago
terraform.tfstate	script related to VPC, subnets, Security Groups, Internet Gate...	44 minutes ago
testfile	just a sample commit, nothing imp	16 hours ago

README

About

terraform for test

Activity · 0 stars · 1 watching · 0 forks

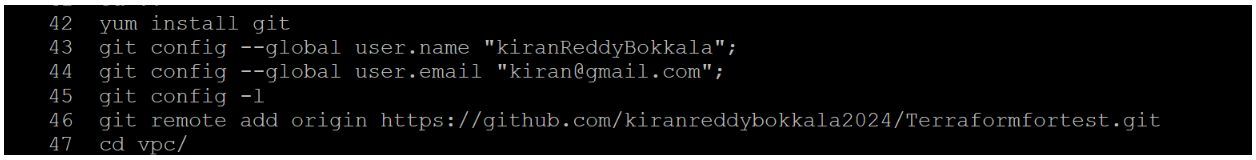
Releases

No releases published
Create a new release

Packages

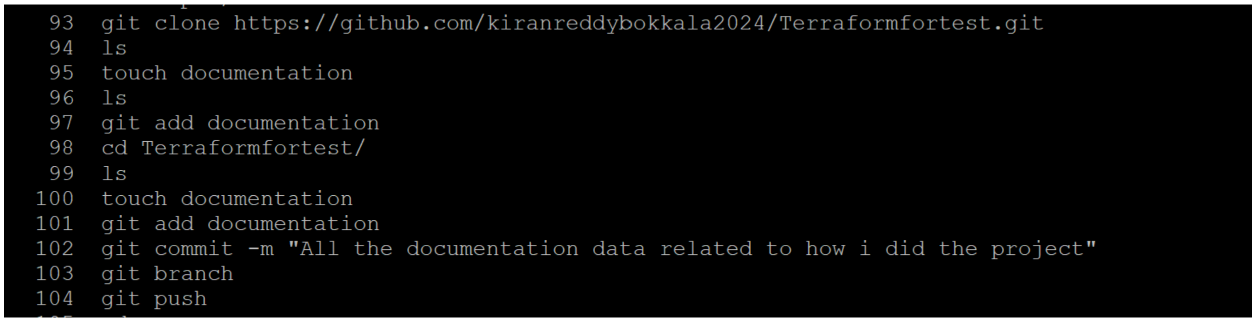
No packages published
Publish your first package

6. Installed Git in the terraform server and cloned the data first.



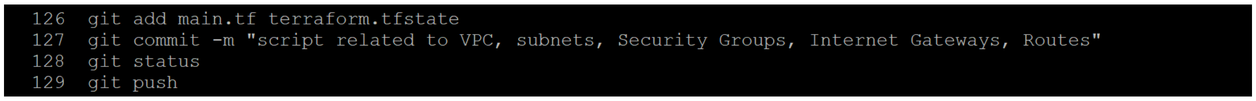
```
yum install git
git config --global user.name "kiranReddyBokkala";
git config --global user.email "kiran@gmail.com";
git config -l
git remote add origin https://github.com/kiranreddybokkala2024/Terraformfortest.git
cd vpc/
```

7. Now just created a documentation file for steps and pushed it to the GitHub account.



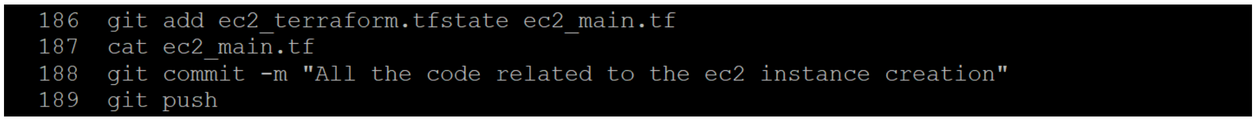
```
git clone https://github.com/kiranreddybokkala2024/Terraformfortest.git
ls
touch documentation
ls
git add documentation
cd Terraformfortest/
ls
touch documentation
git add documentation
git commit -m "All the documentation data related to how i did the project"
git branch
git push
```

8. Now pushed the data related to VPC into GitHub.



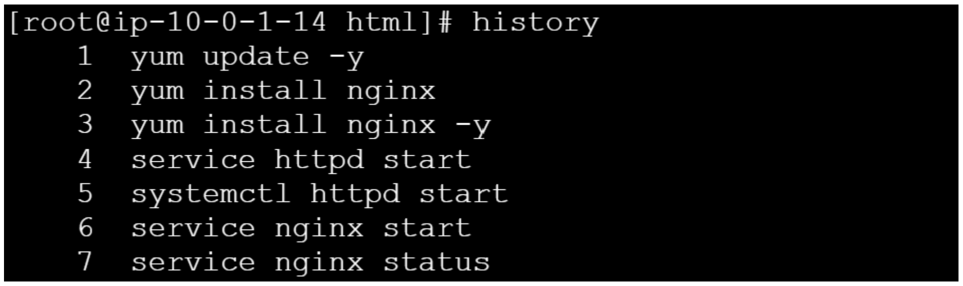
```
git add main.tf terraform.tfstate
git commit -m "script related to VPC, subnets, Security Groups, Internet Gateways, Routes"
git status
git push
```

9. Now pushed the data related to EC2 Nginx instance into GitHub account.



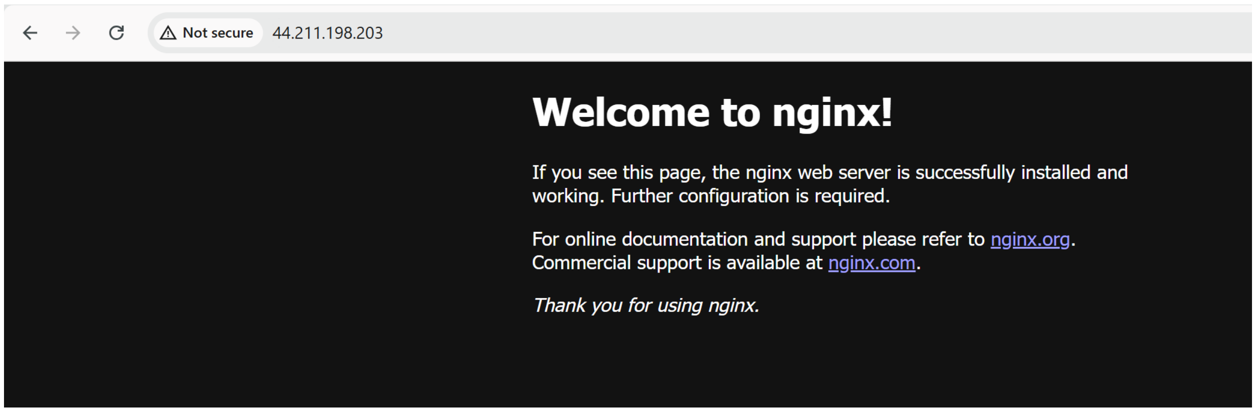
```
  186  git add ec2_terraform.tfstate ec2_main.tf
  187  cat ec2_main.tf
  188  git commit -m "All the code related to the ec2 instance creation"
  189  git push
```

10. As mentioned, Later after deploying the Nginx server using Terraform ,Logged into the Nginx server and executed below commands to install Nginx server.



```
[root@ip-10-0-1-14 html]# history
    1  yum update -y
    2  yum install nginx
    3  yum install nginx -y
    4  service httpd start
    5  systemctl httpd start
    6  service nginx start
    7  service nginx status
```

Default page:



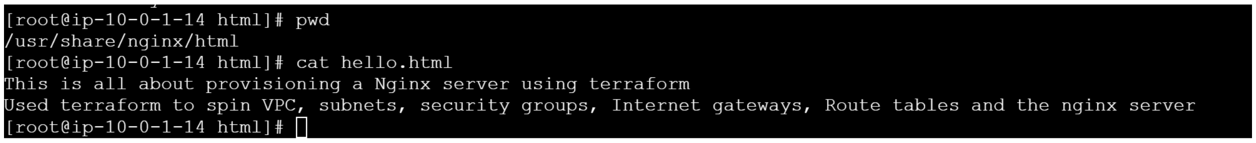
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Custom page



```
[root@ip-10-0-1-14 html]# pwd
/usr/share/nginx/html
[root@ip-10-0-1-14 html]# cat hello.html
This is all about provisioning a Nginx server using terraform
Used terraform to spin VPC, subnets, security groups, Internet gateways, Route tables and the nginx server
[root@ip-10-0-1-14 html]# 
```

i-01bafe1e32f6acbf1 (NginxInstance)

PublicIPs: 44.211.198.203   PrivateIPs: 10.0.1.14