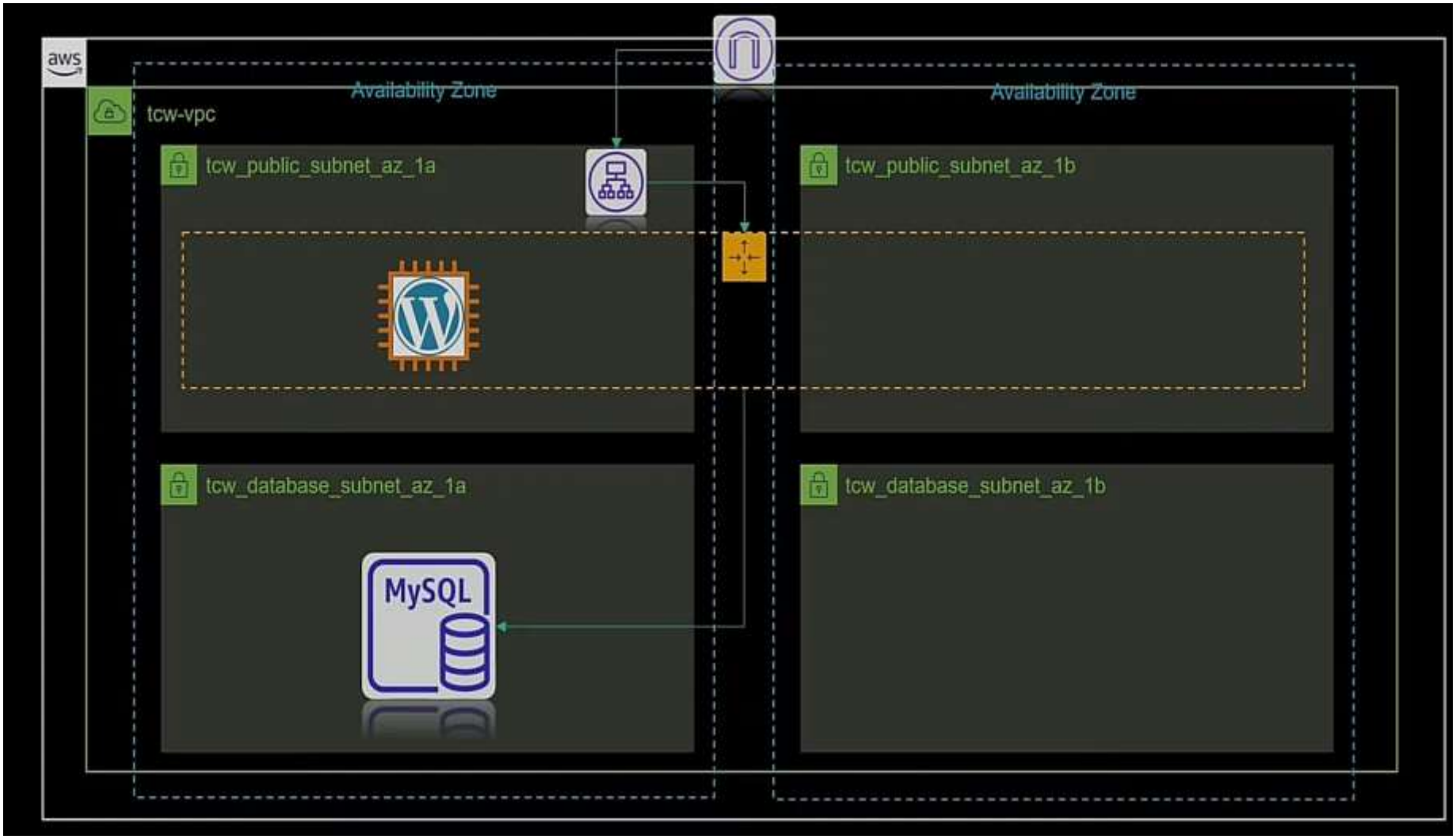
Project

This terraform projects creates Vpc, LoadBalancer, Autoscaling, RDS databse with mysql

#EC2 instance with wordpress installed and configured

- Architecture of project look like this:--



Prerequisite

- AWS account
- Terraform
- Amazon cli if having window

Continue---------

Route Table for Application

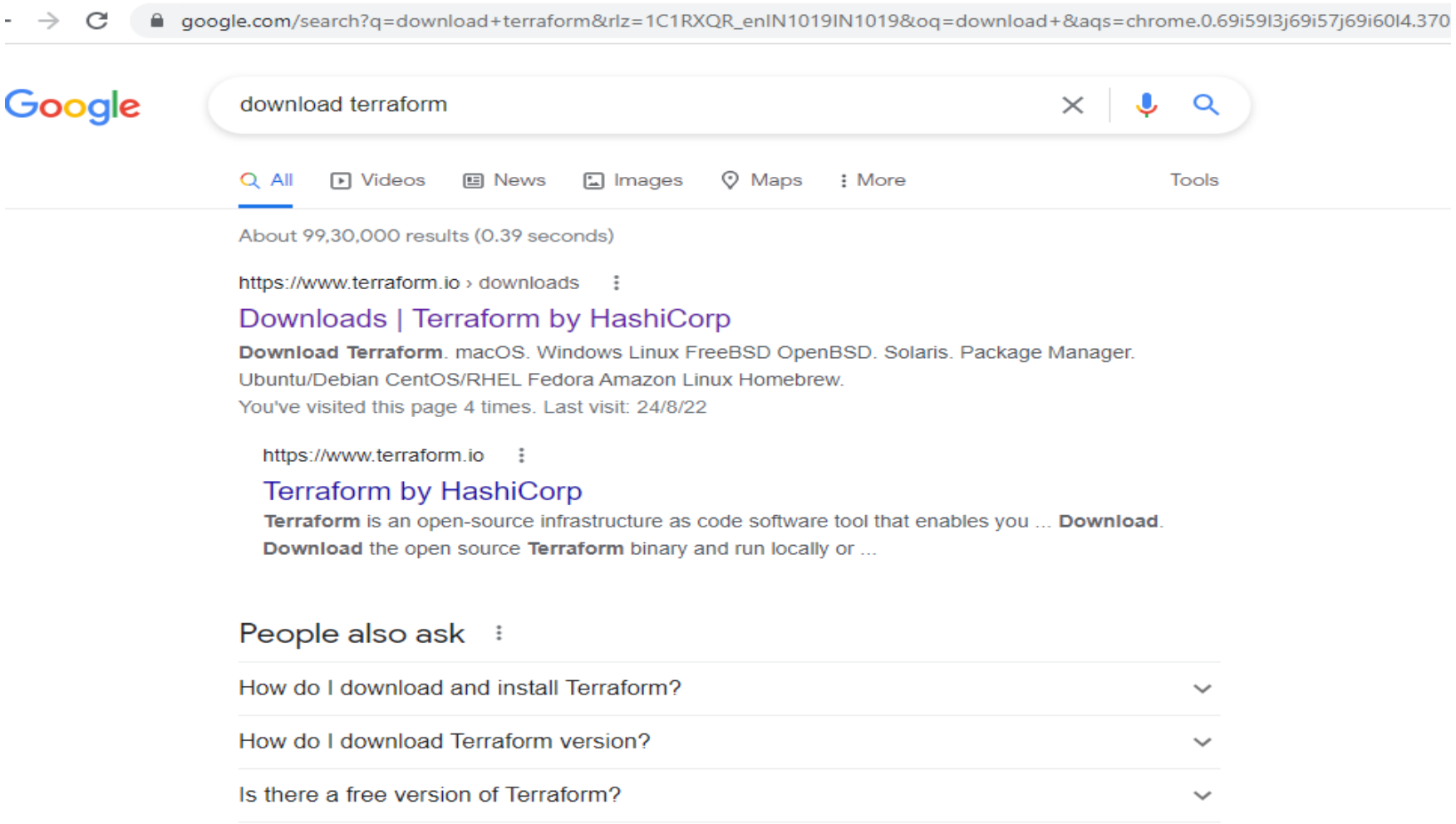
ROUTE	CIDR_BLOCK
Local	192.168.0.0/16
Internet_Gateway	0.0.0.0/0

Route Table for Database

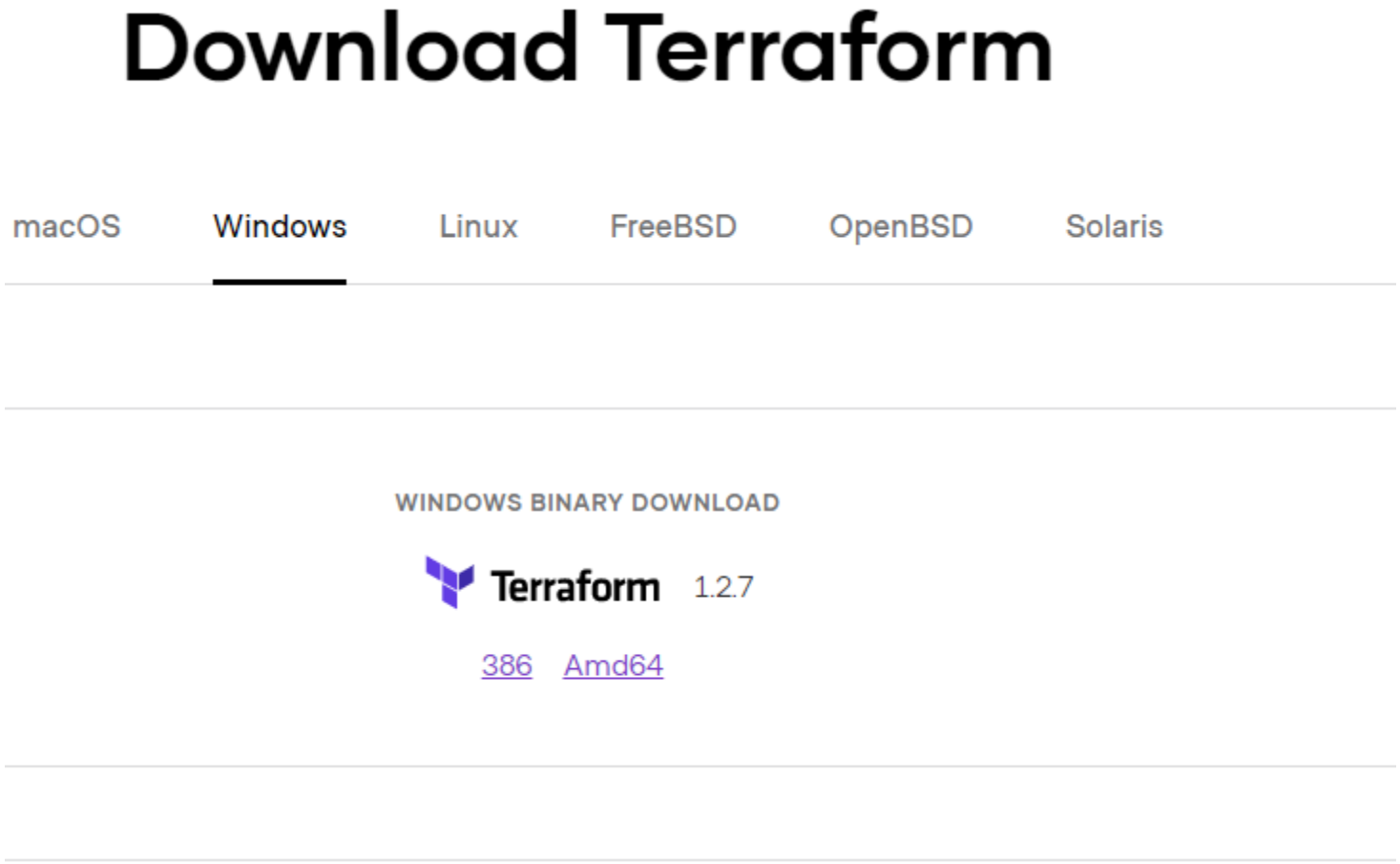
ROUTE	CIDR_BLOCK
Local	192.168.0.0/16

Downloading & Installing Prerequisite

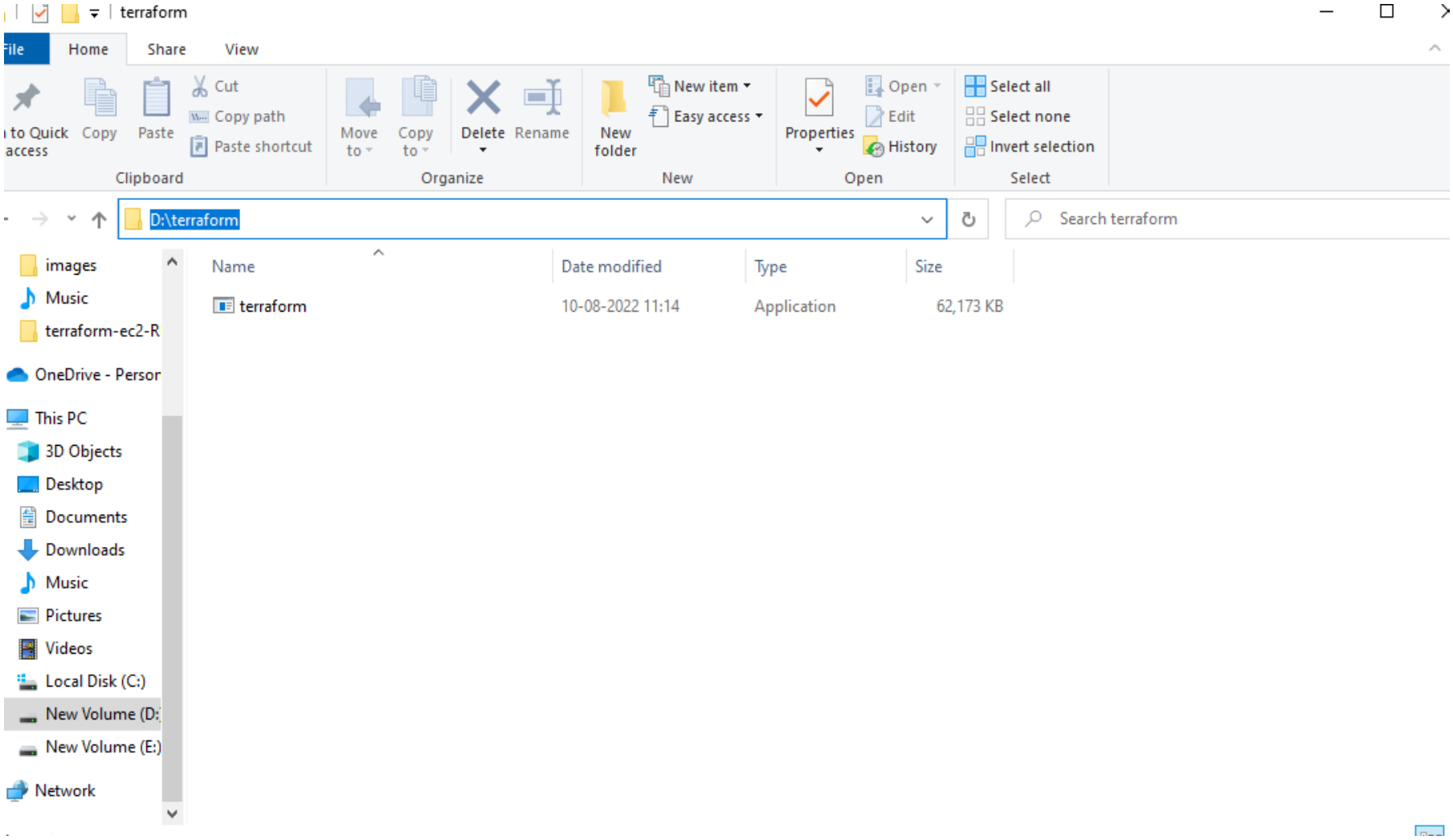
- Go to google and type download terraform



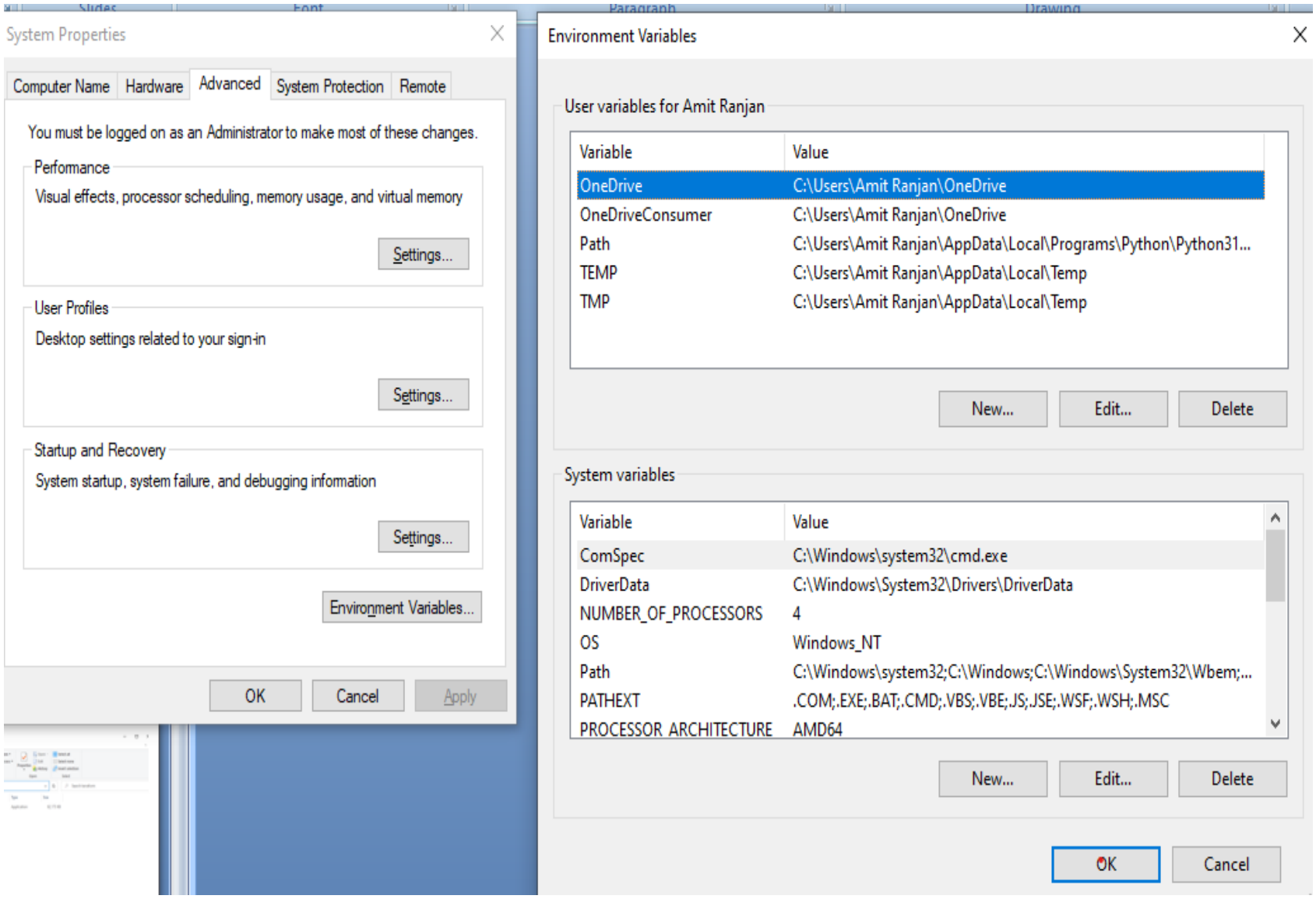
Click to Download|Terraform by HasiCorp



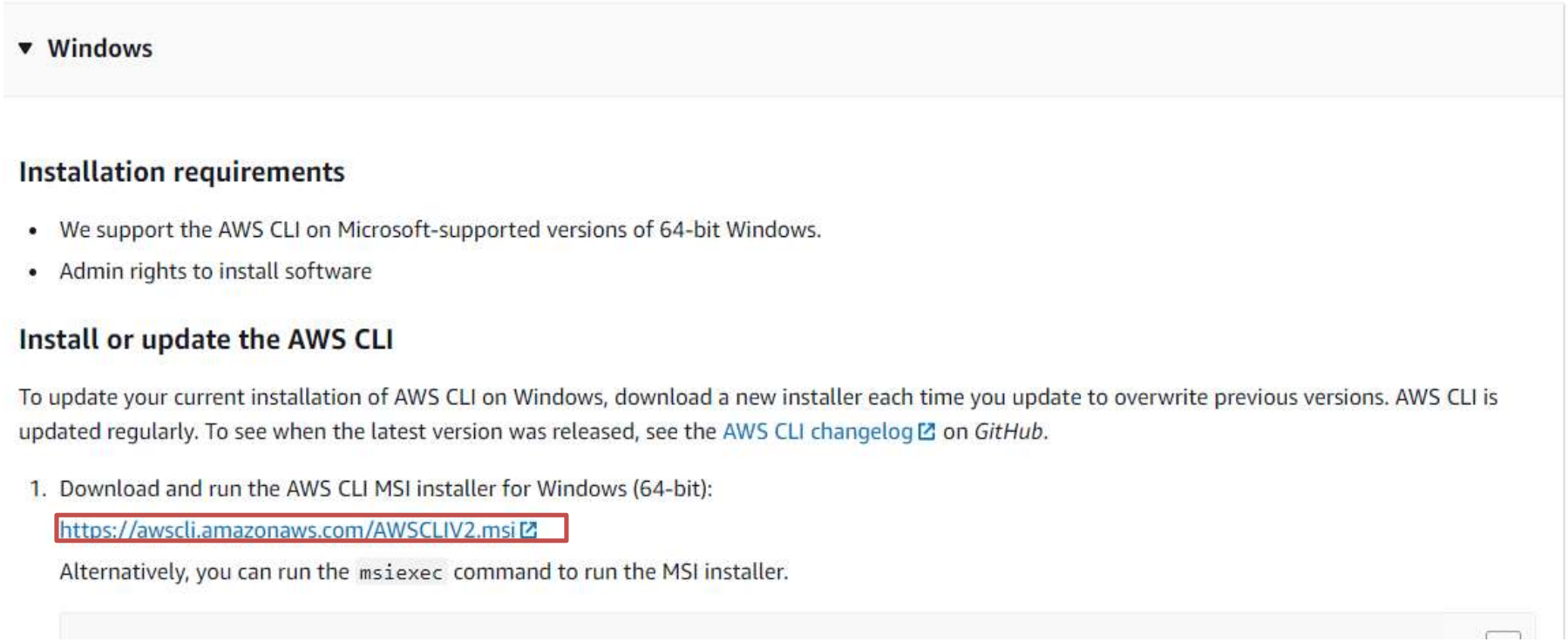
Copy the path where terraform is located



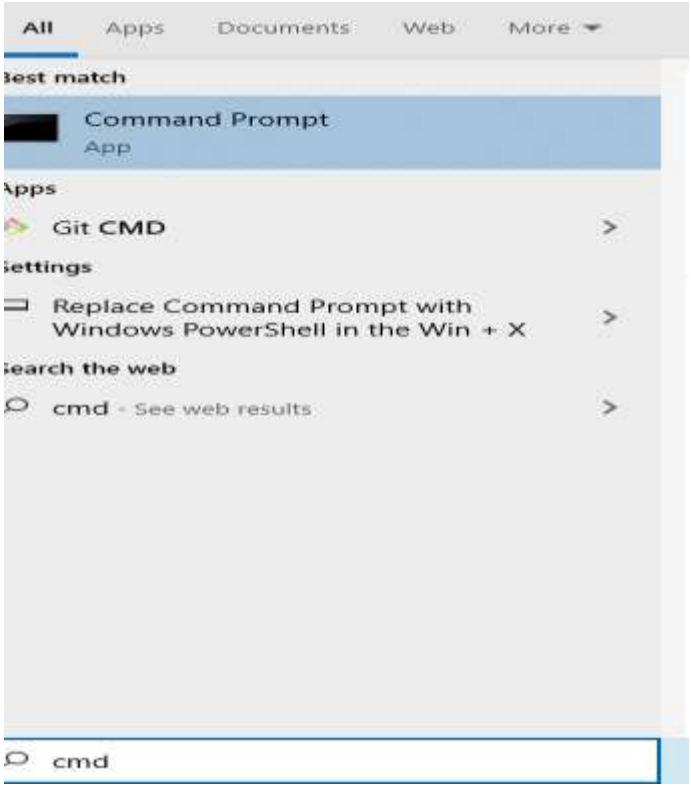
•Paste it to environment variable

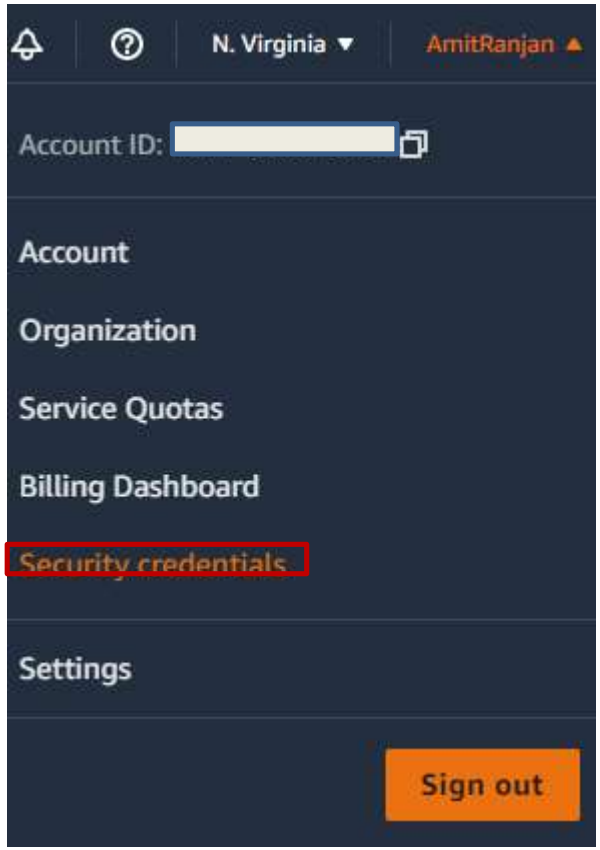


- Download Aws cli
- Url:-- https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html
- Click on the https://awscli.amazonaws.com/AWSCLIV2.msi as shown below .
- After downloading click on install

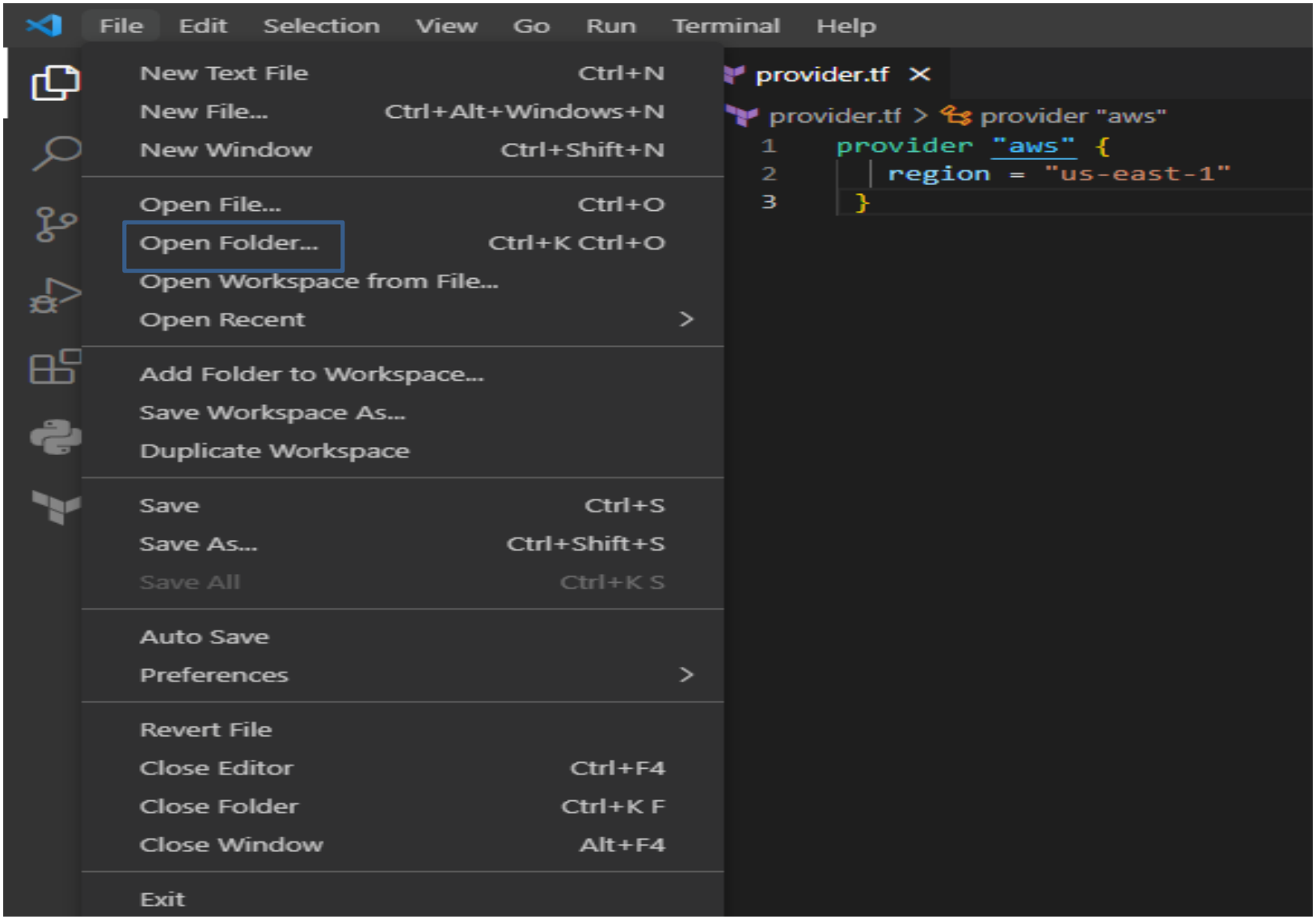


•After installing aws cli go to cmd and write aws configure and it will ask for access and secret key put public and secret key of aws console and go to security credential and click on Access keys (access key ID and secret access key) and then click on **Create New Access Key** and download the csv file

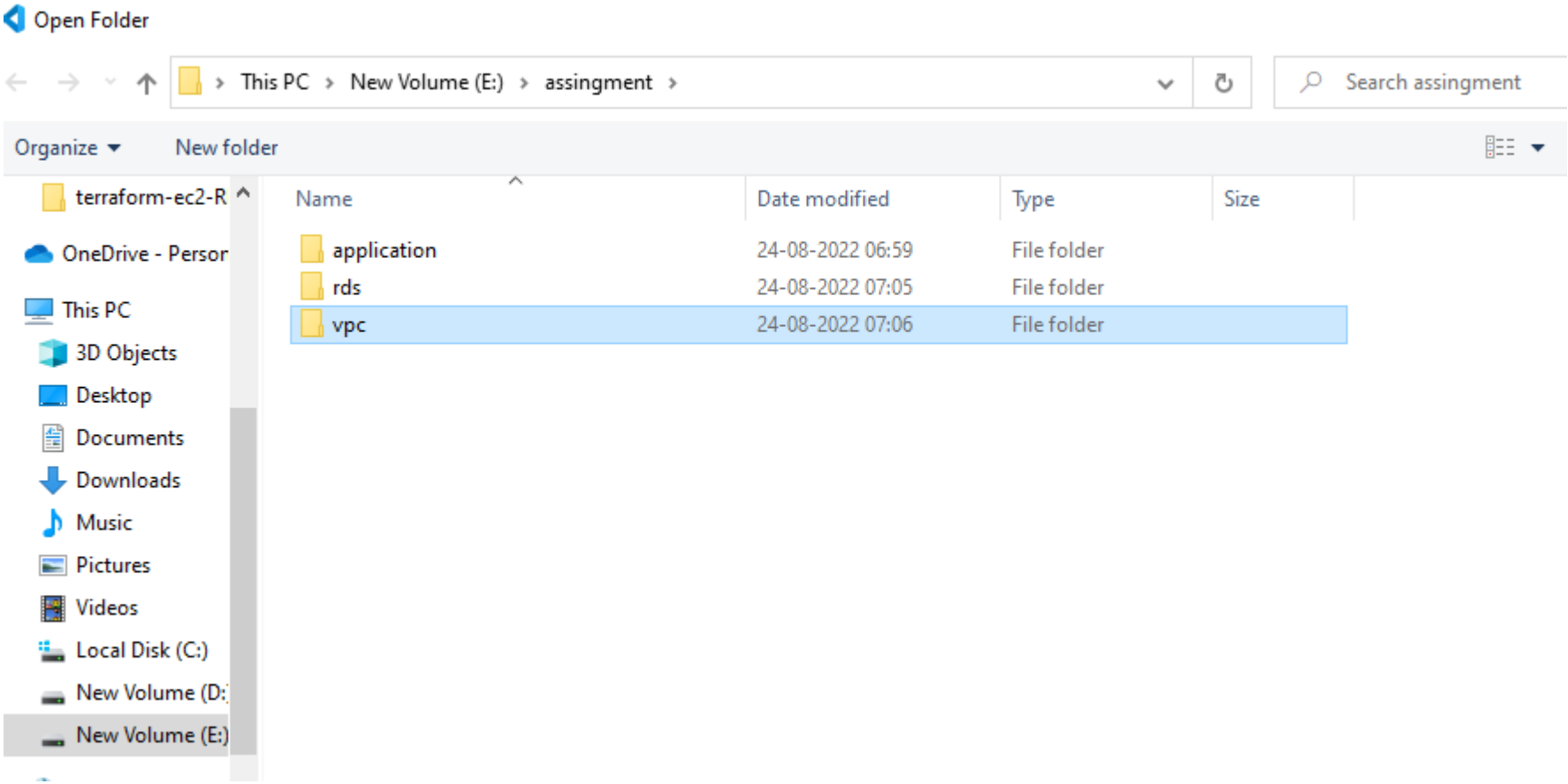




• Open vscode and go to

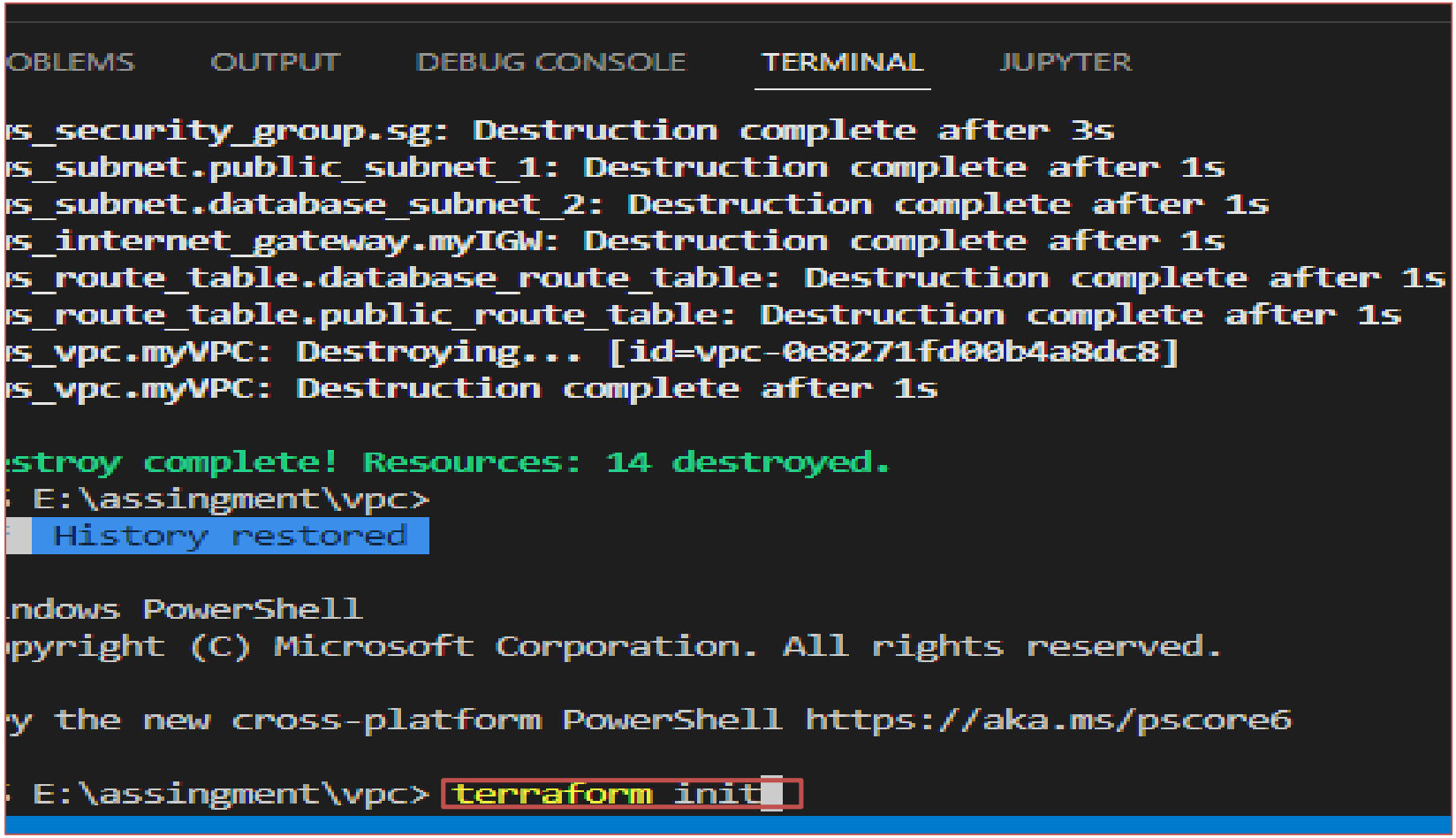


•Open vpc folder and click select folder on the right bottom side



•Now click on terminal in the vs code

•Commands you have to use in this project



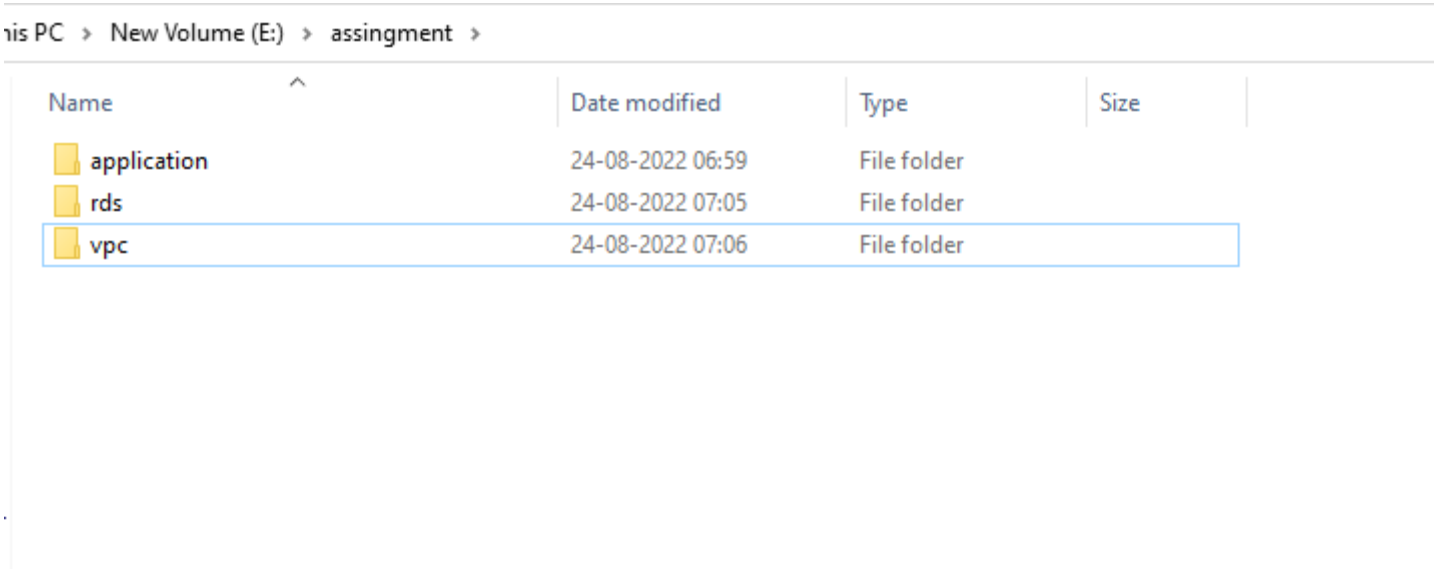
•Now click terraform plan and then terraform apply

```
PS D:\terraform project\Assingment\vpc_creation> terraform plan
data.aws_availability_zones.available: Reading...
data.aws_availability_zones.available: Read complete after 0s [id=us-east-1]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicat
  + create

Terraform will perform the following actions:

  # aws_internet_gateway.igw will be created
  + resource "aws_internet_gateway" "igw" {
      + arn      = (known after apply)
      + id       = (known after apply)
      + owner_id = (known after apply)
      + tags     = {
          + "Name" = "tcw_igw"
        }
      + tags_all = {
          + "Name" = "tcw_igw"
        }
      + vpc_id   = (known after apply)
    }

  # aws_route.public_internet_gateway will be created
  + resource "aws_route" "public_internet_gateway" {
      + destination_cidr_block = "0.0.0.0/0"
```

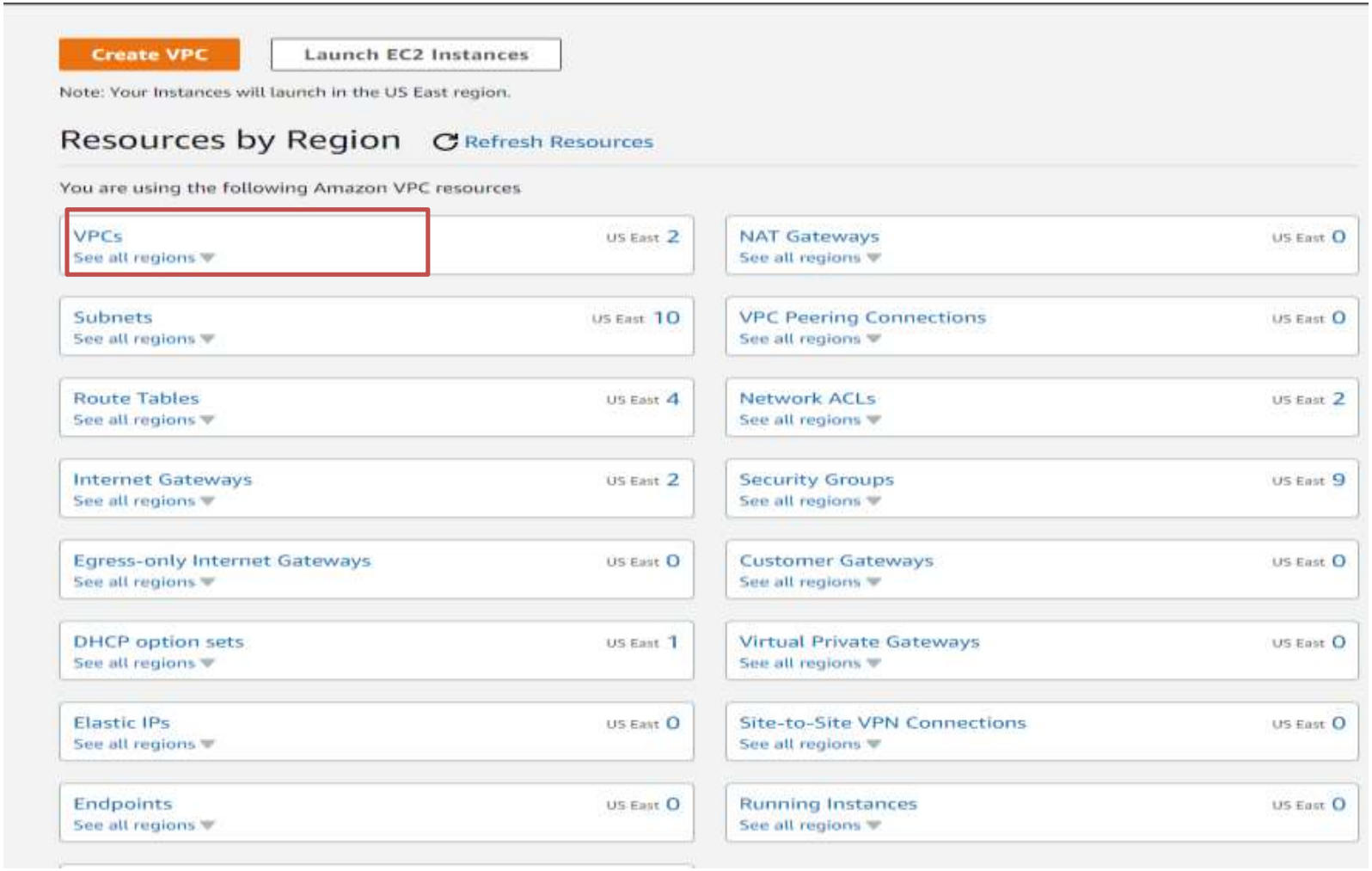
```
Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform a
PS E:\assingment\application> terraform apply
data.template_file.user_data: Reading...
data.template_file.user_data: Read complete after 0s [id=67cb15a07cc39847133978765474b839b7d51932047b23beb4cd83ea4ef3a74f]
data.aws_subnets.available_db_subnet: Reading...
data.aws_security_group.tcw_sg: Reading...
data.aws_vpc.vpc_available: Reading...
data.aws_ami.amazon_linux_2: Reading...
data.aws_subnets.available_db_subnet: Read complete after 1s [id=us-east-1]
data.aws_security_group.tcw_sg: Read complete after 1s [id=sg-07669d2f51270c167]
data.aws_ami.amazon_linux_2: Read complete after 1s [id=ami-02538f8925e3aa27a]
data.aws_vpc.vpc_available: Read complete after 2s [id=vpc-0e8271fd00b4a8dc8]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:
```

•Same you have to do with the Rds creation and application all the variable are there in the variable.tf

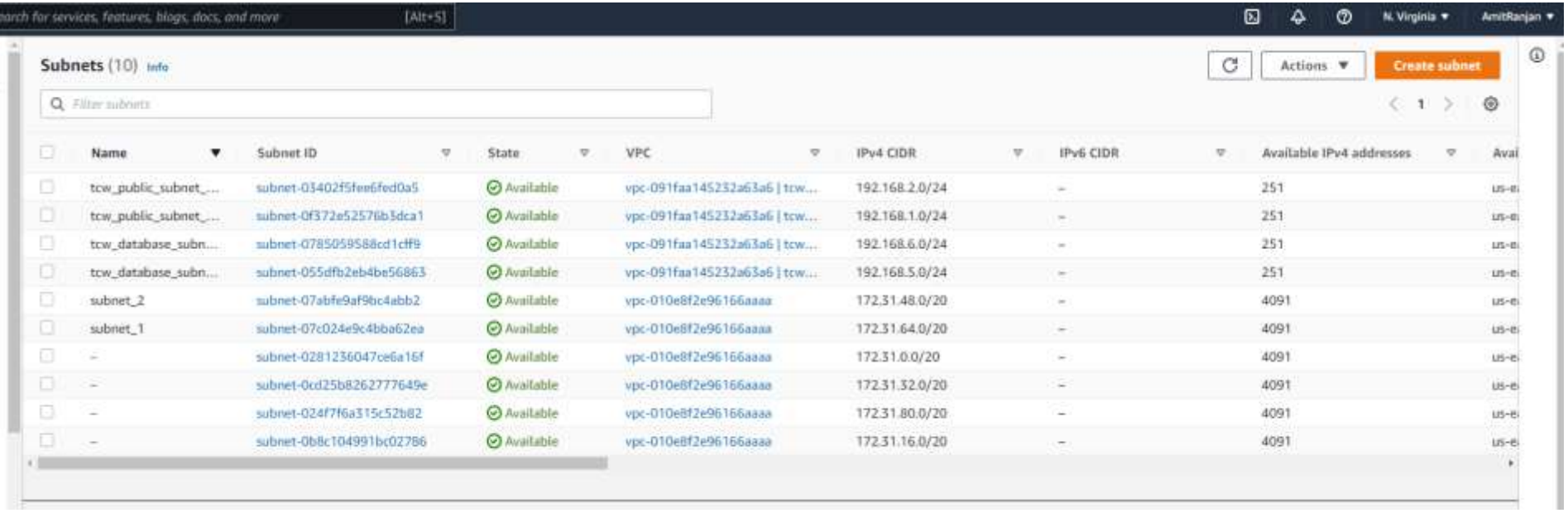
•Command which you have to use i;e:-- terraform init , terraform plan, terraform apply



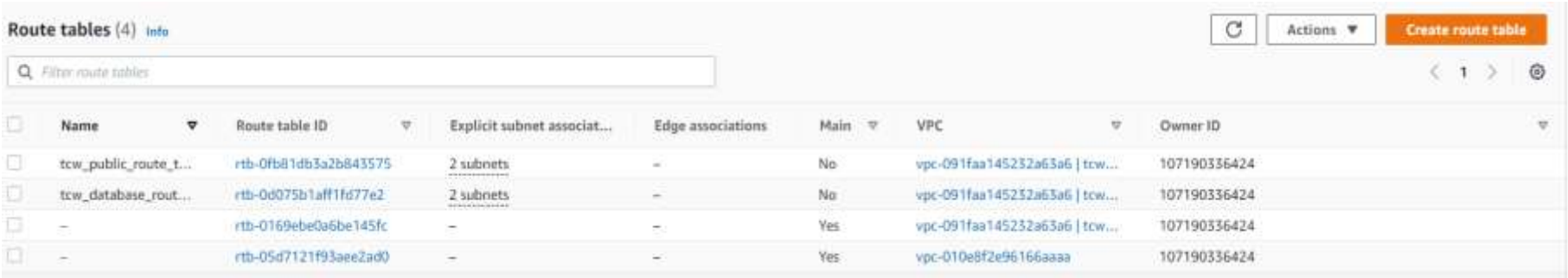
•Ater creation of vpc , Rds go to aws console and check once whether it is created or not as shown on the image below :--



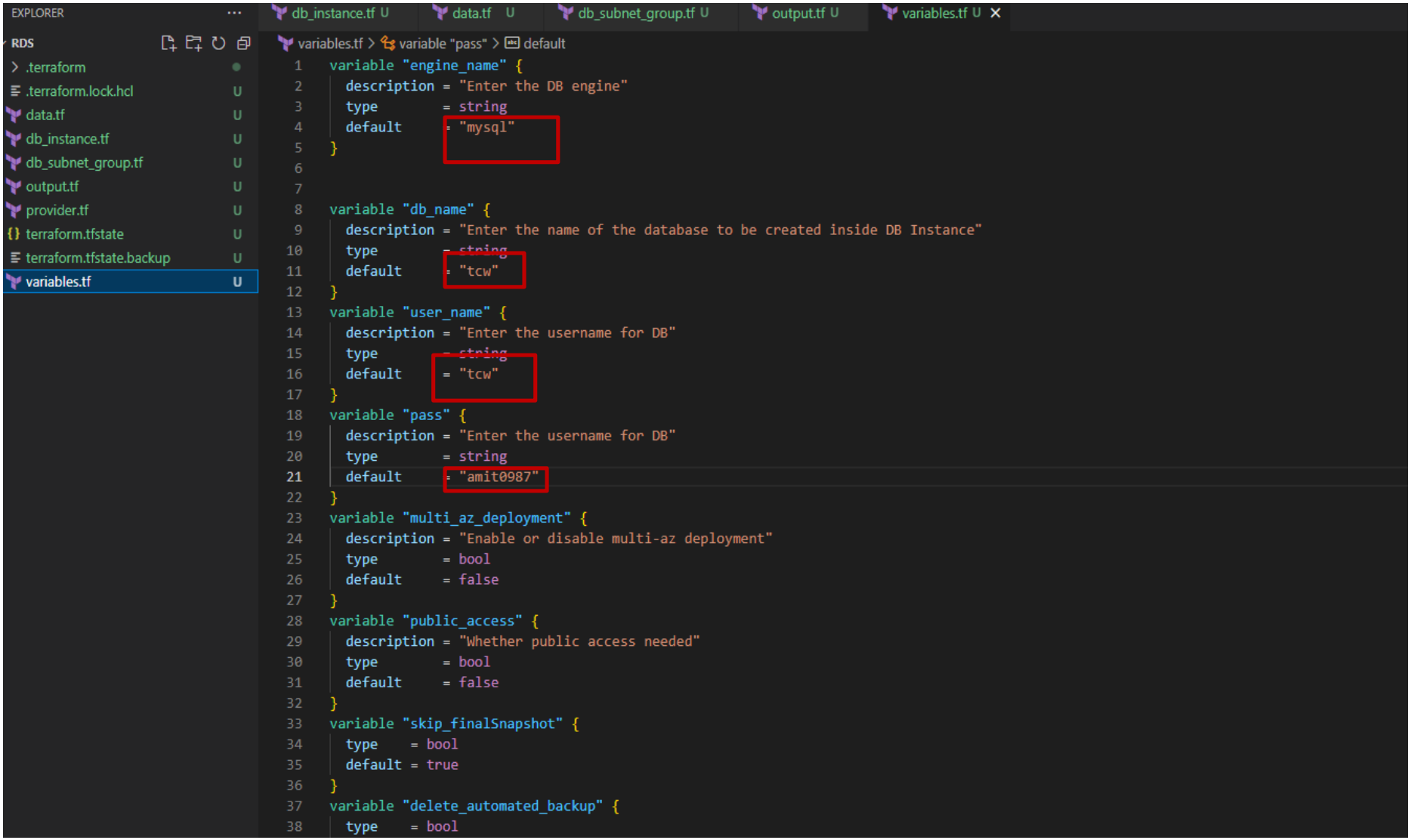
It will be like this as shown on this image:--



•Internet gateway and route table will be like this as shown below

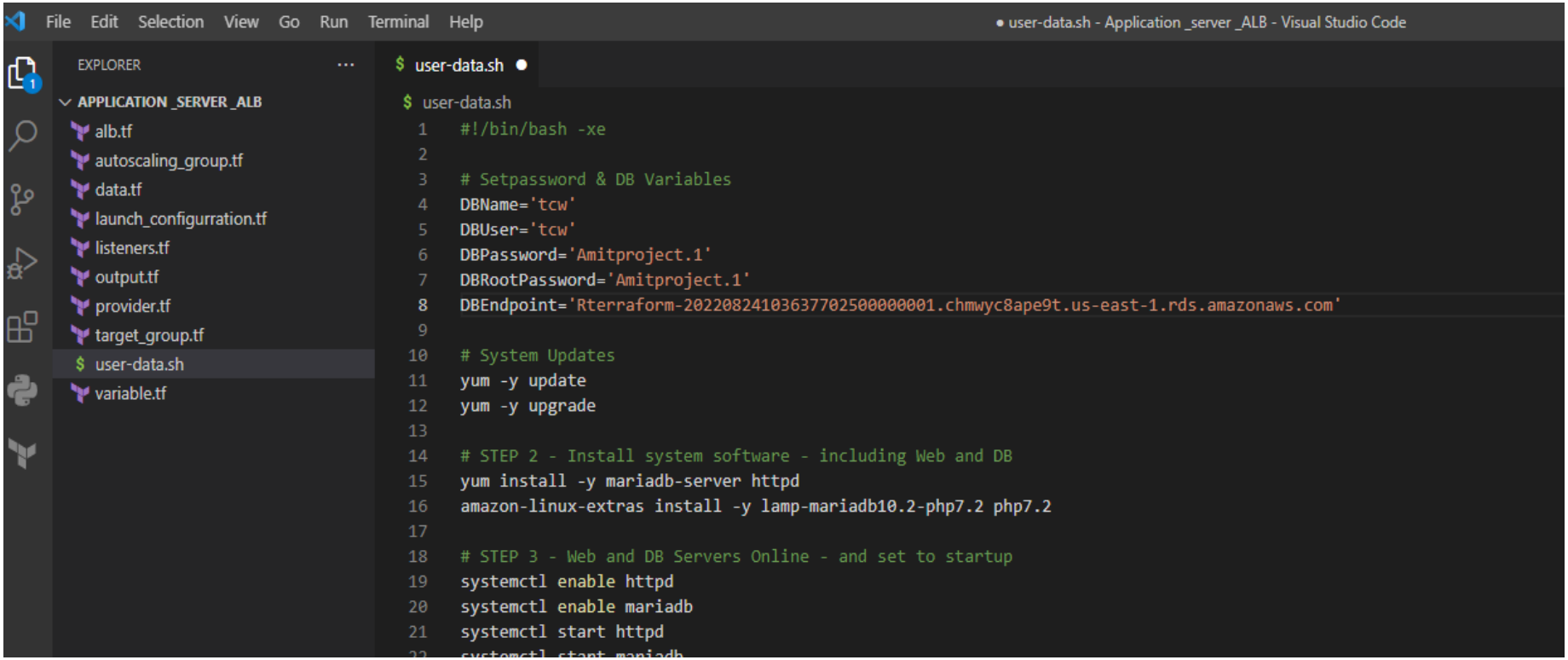


•To change user engine, name and password of the database go to dbs folder under there there is a variable.tf all the variable like database name password are there:-

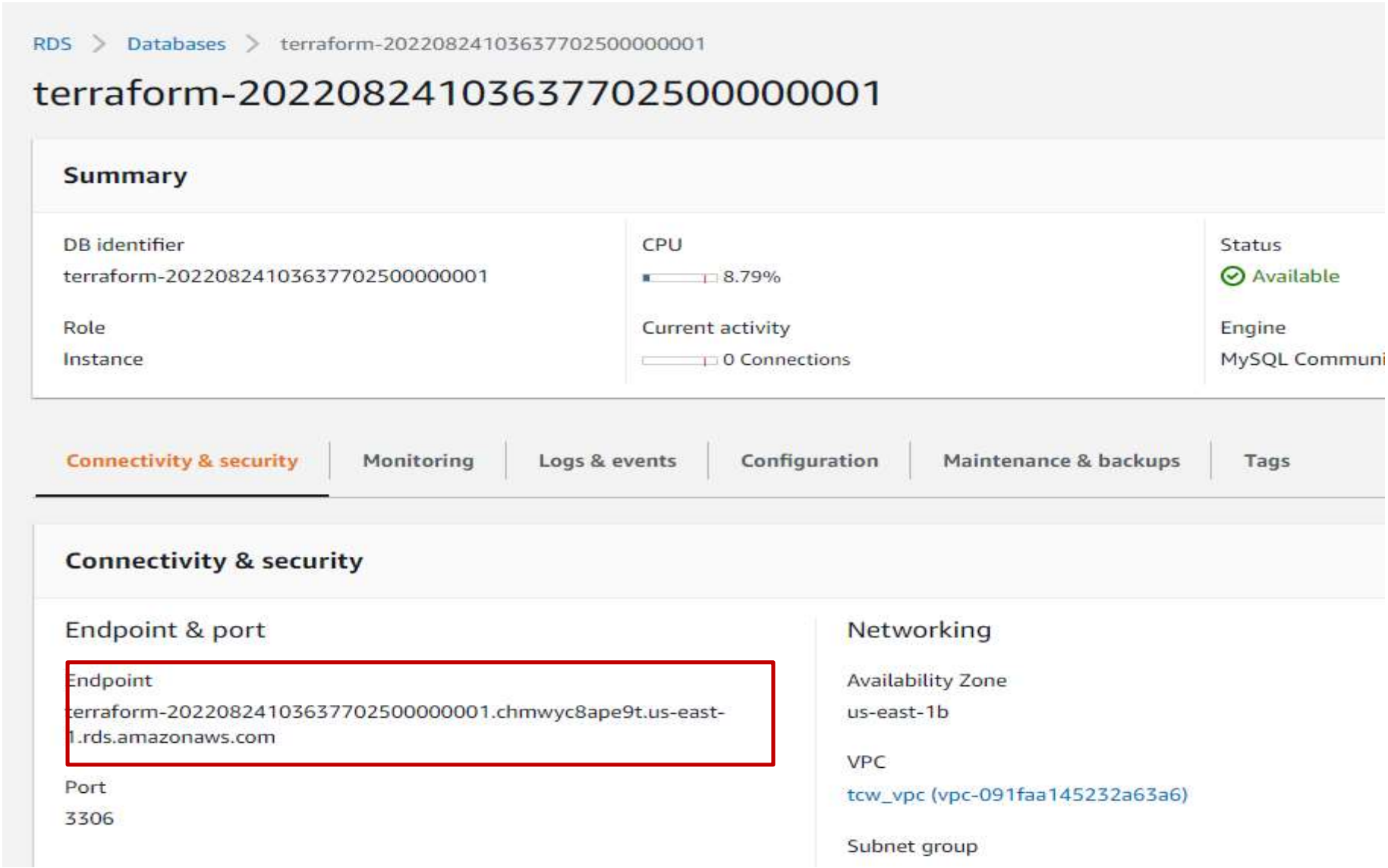


•After creating database lastly come to the application folder

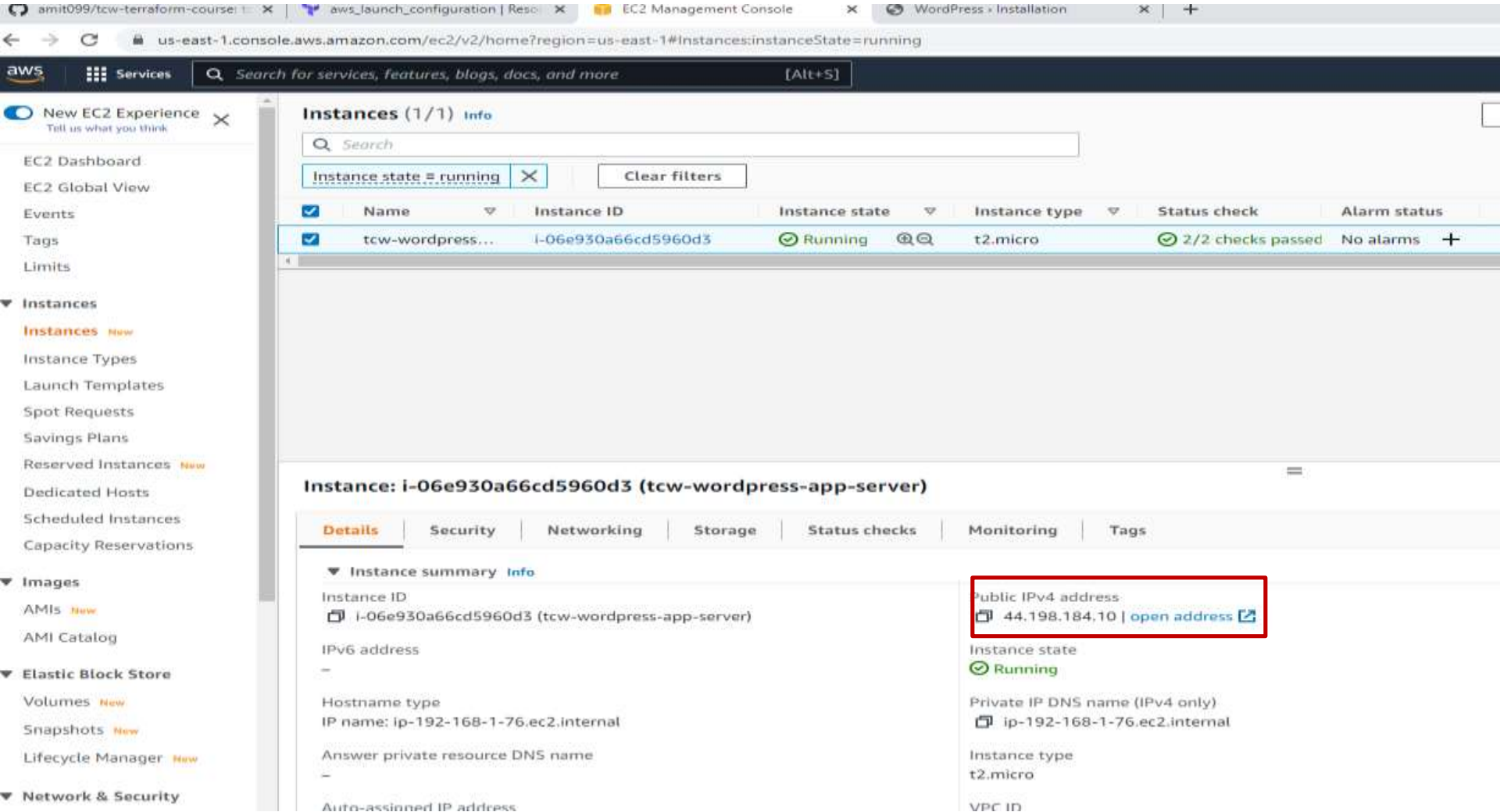
•Now go to user-data.sh if you modified the database name, username and pass you have to put here the modified Dbname, Dbuser, DBPassword and for DBEndpoint you have to go to aws console search RDS themn Copy the endpoint which is provided there



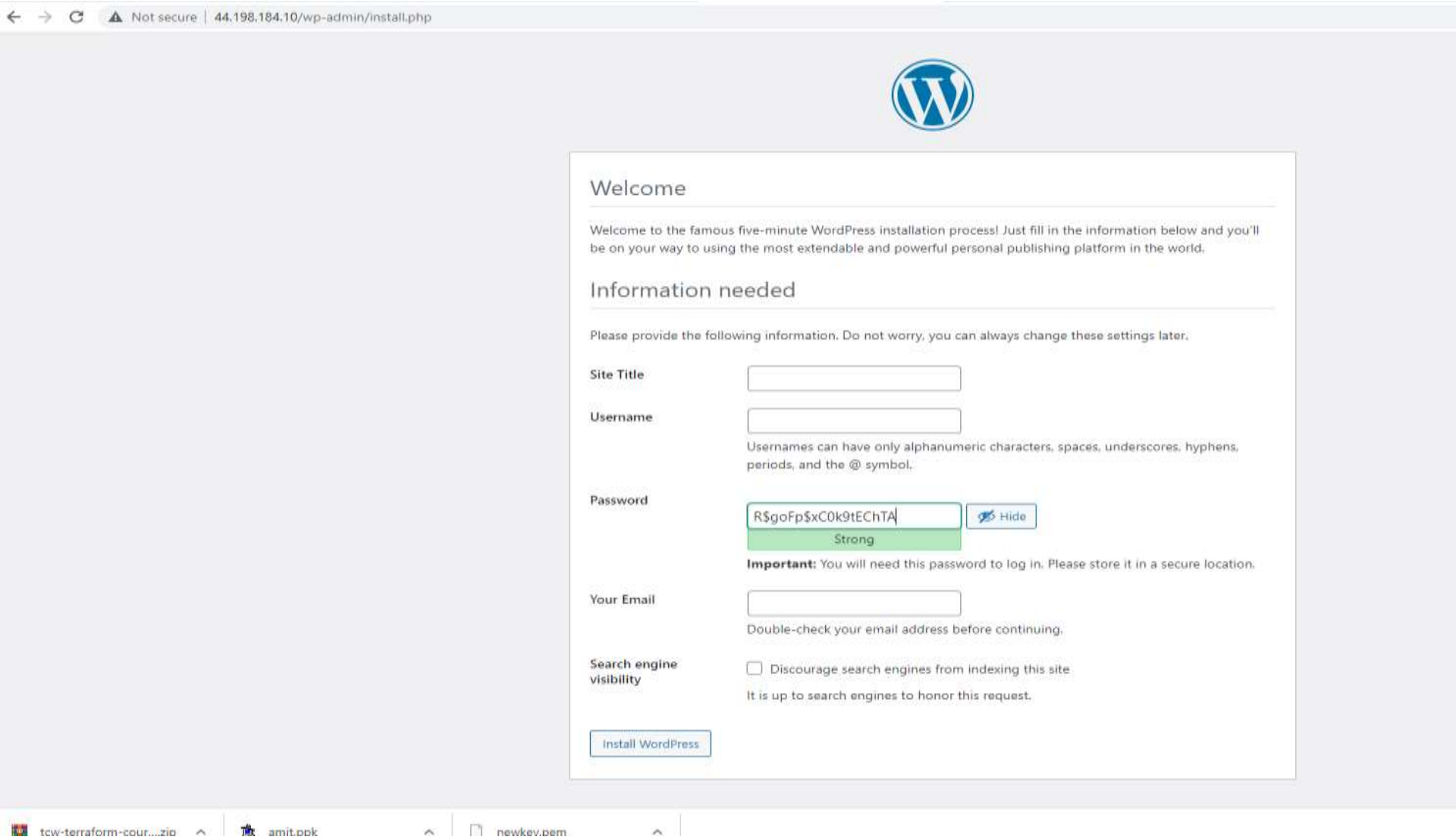
•Endpoint will be like this as shown below



- After doing all this step in application which is in this folder type terraform apply
- And to remove simply type terraform destroy whatherever you want to remove whether it is vpc, rds, or application server
- After apply go to the Aws console copy the public ip and put it on the browser address bar and click ok



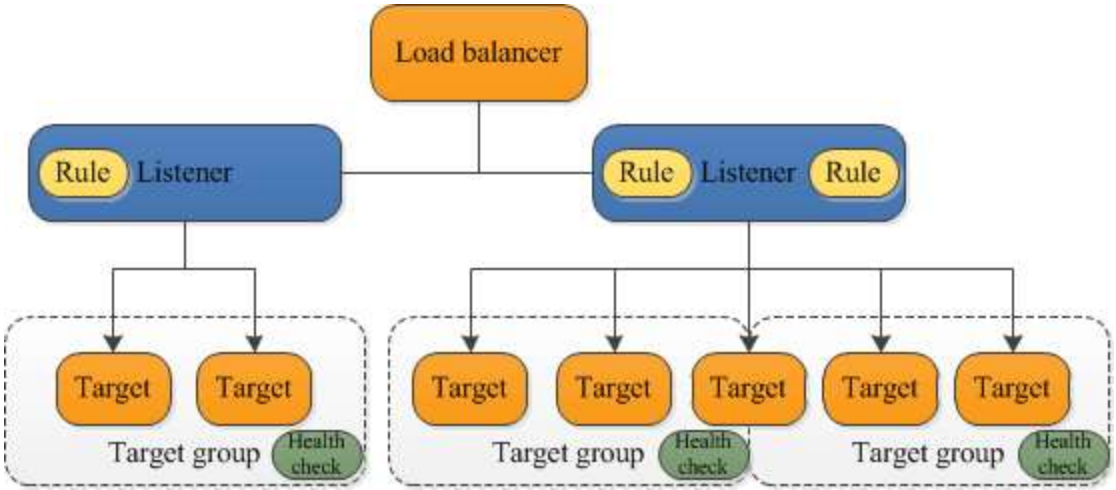
HURRAY! All Done it will look like this:---



Advantages:--

Of having Application load balancer and Autoscaling are as follows:--

The load balancer distributes incoming application traffic across multiple targets, such as EC2 instances, in multiple Availability Zones. This increases the availability of your application. You add one or more listeners to your load balancer.

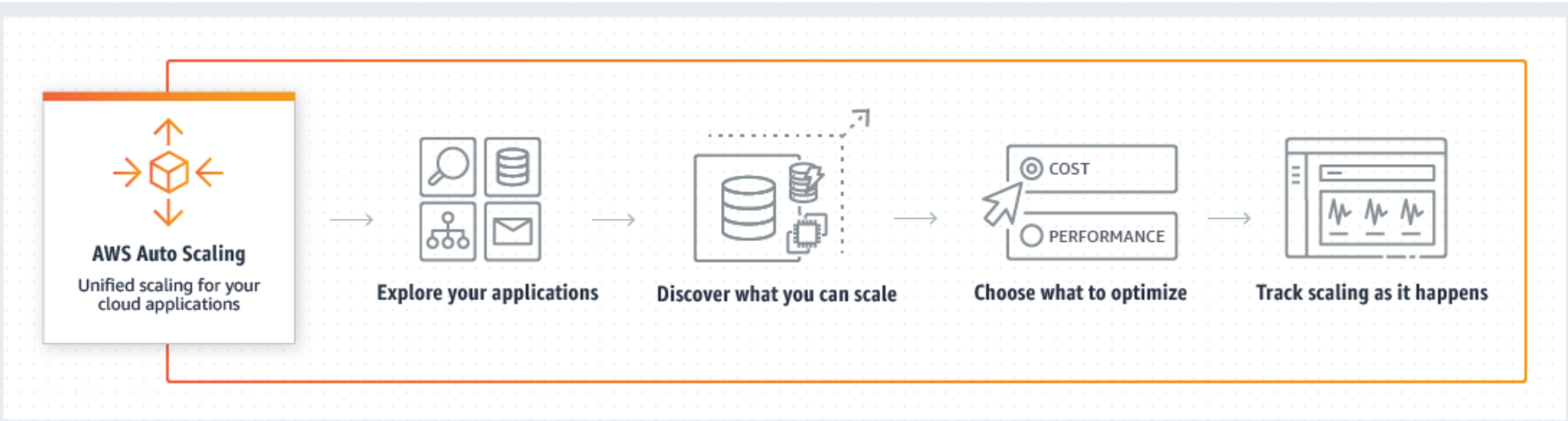


Autoscaling:-

•SETUP SCALING QUICKLY

•AUTOMATICALLY MAINTAIN PERFORMANCE

•AWS Auto Scaling monitors your applications and automatically adjusts capacity to maintain steady, predictable performance at the lowest possible cost. Using AWS Auto Scaling, it's easy to setup application scaling for multiple resources across multiple services in minutes. The service provides a simple, powerful user interface that lets you build scaling plans for resources including Amazon ec2 instances and Spot Fleets, Amazon ECs tasks, Amazon DynamoDB tables and indexes, and Amazon Aurora Replicas. AWS Auto Scaling makes scaling simple with recommendations that allow you to optimize performance, costs, or balance between them. If you're already using Amazon EC2 Auto Scaling to dynamically scale your Amazon EC2 instances, you can now combine it with AWS Auto Scaling to scale additional resources for other AWS services. With AWS Auto Scaling, your applications always have the right resources at the right time.



THANK YOU

For any further clarifications/patch assistance, please contact:- 9113446714

email:- amit.ranjan.akr@gmail.com