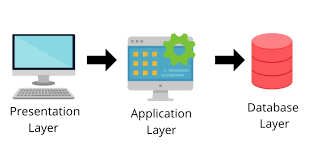
**PROJECT – 1**

**DEPLOYMENT OF 3 TIER ARCHITECTURE IN AWS BY USING TERRAFORM**

**What is 3 tier ARCHITECTURE?**

A three-tier architecture is a software architecture pattern commonly used in the development of web applications. It divides

an application into three interconnected components or tiers, each responsible for specific functionality. The three tiers are

Presentation Tier (Client Tier):

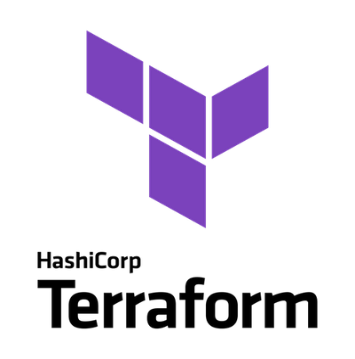
* Also known as the user interface (UI) or front-end tier.
* Responsible for presenting information to the user and receiving user inputs.
* Typically includes web browsers or client applications that interact with the application's interface.

Application Tier (Logic Tier or Middle Tier):

* Contains the business logic or application logic that processes and manages data.
* Responsible for processing user requests, executing business rules, and interacting with the data tier.
* Often implemented using server-side technologies, such as application servers, web servers, or serverless functions.

Data Tier (Data Storage or Backend Tier):

* Manages and stores the data used by the application.
* Includes databases, data storage systems, and other data-related components.
* Responsible for retrieving and storing data in response to requests from the application tier.

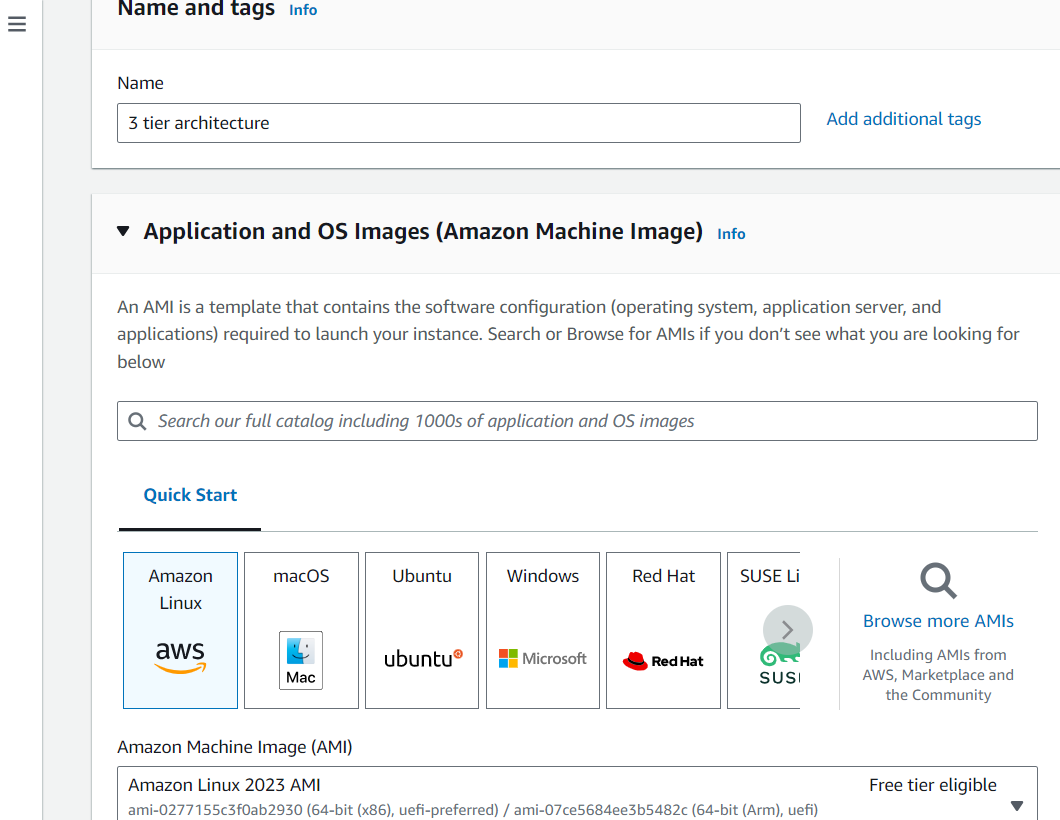
**WHAT IS TERRAFORM?**

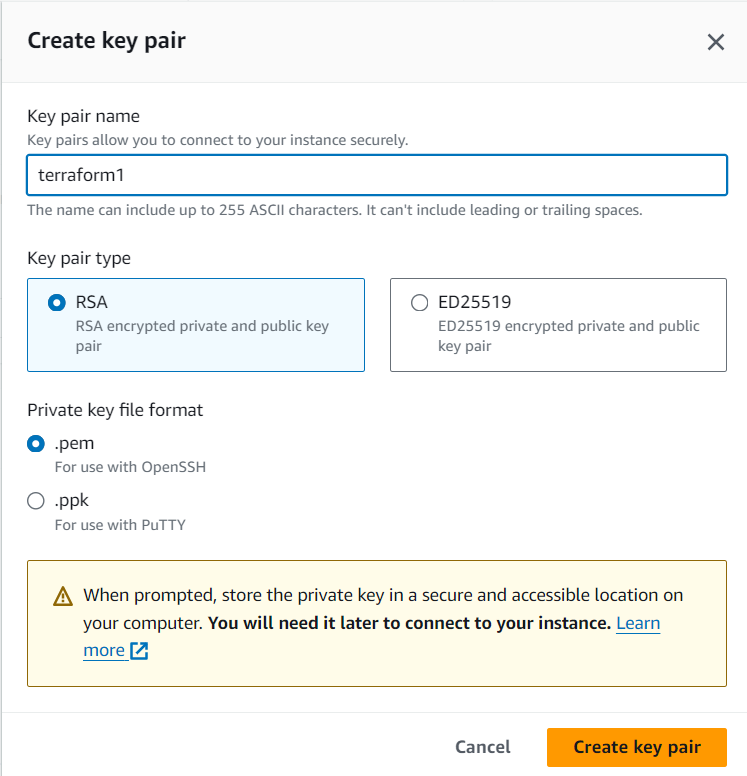
Terraform is an open-source infrastructure as code (IAC) tool developed by HASHICORP. It allows users to define and provision infrastructure using a declarative configuration language. With Terraform, you can describe your infrastructure components and their dependencies in code, and then use that code to create, modify, and manage infrastructure resources across various cloud providers, on-premises data centers, and other infrastructure platforms.

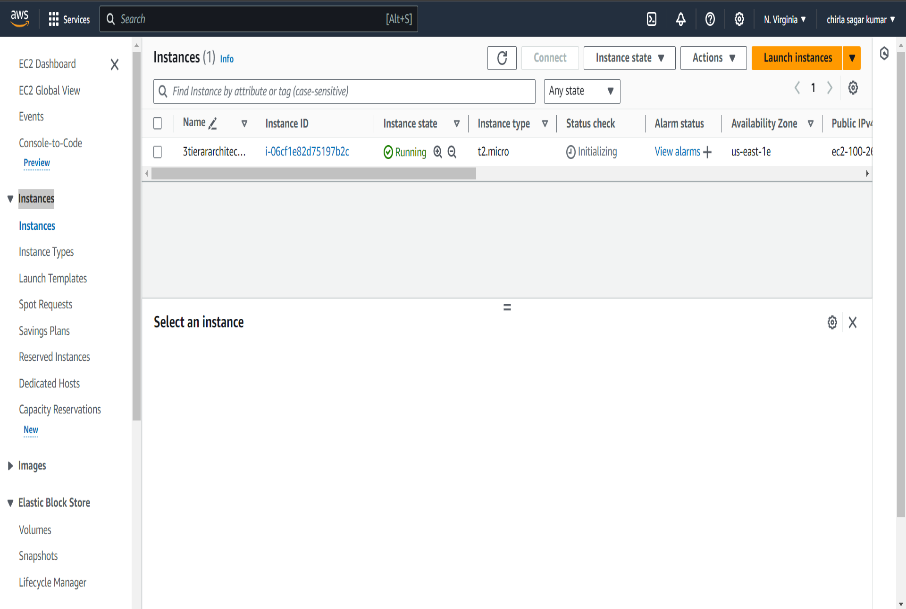
**Contents:**

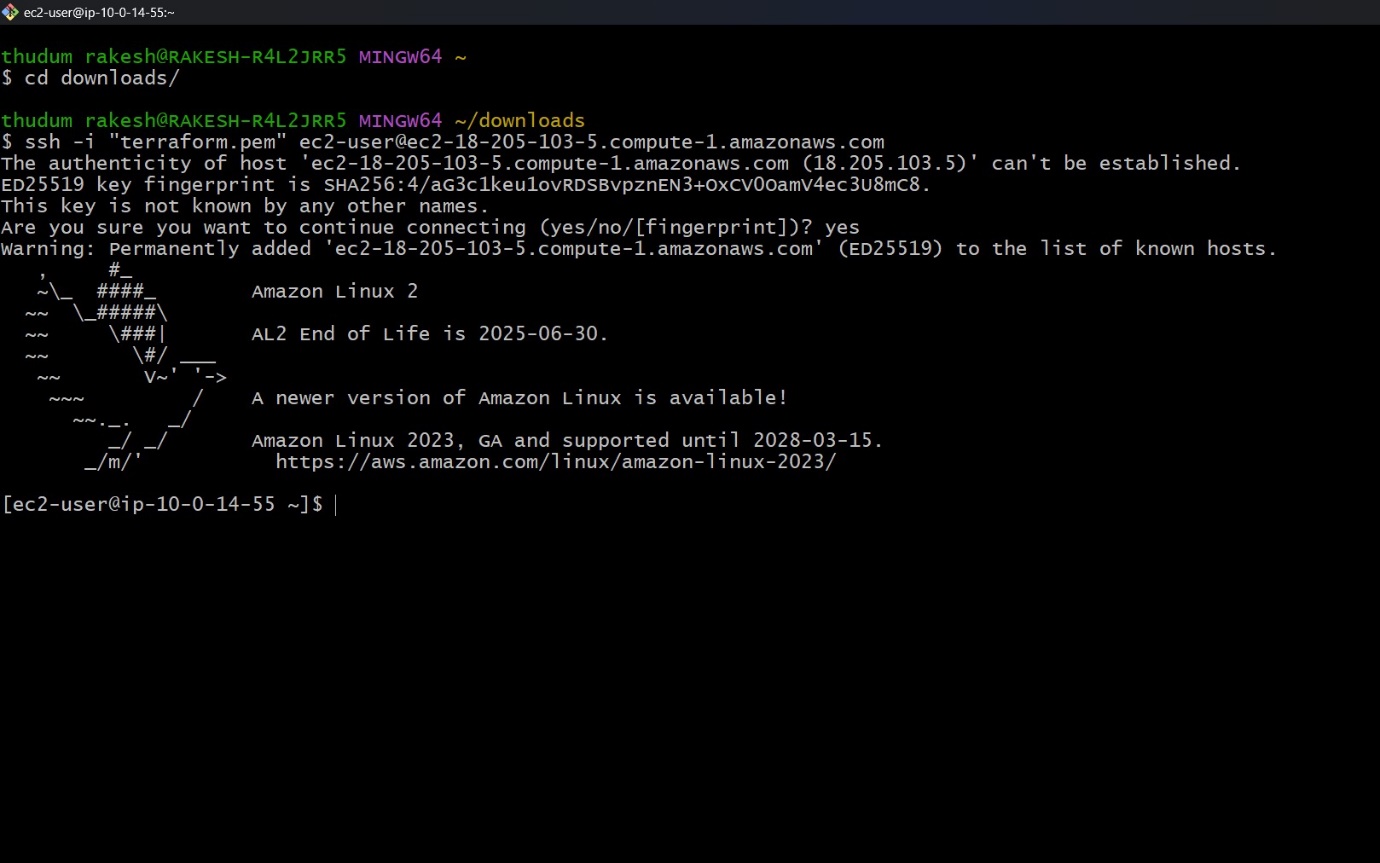
* Create a file for the VPC
* Create a file for the Subnet.
* Create a file for the Internet gateway.
* Create a file for the Route table.
* Create a file for EC2 instance.
* Create a file for security group for front-end tier.
* Create a file for security group for the data base tier.
* Create a file for application load balancer.
* Create a file for RDS instance.
* Create a file for output.
* Create a file for variable.
* Create a file for user data.
* Verify the resources.

To deployment of 3 tier architecture using terraform in AWS we need to create a new EC2 instance.

* Creation of ec2 instance:
* Login to AWS console and search ec2 in the search bar
* Click on launch instance and give the name to the instance and also give the required configuration
* Give the amazon Linux server
* Create a new key pair
* Use free tier eligible
* Next click on launch instance

****

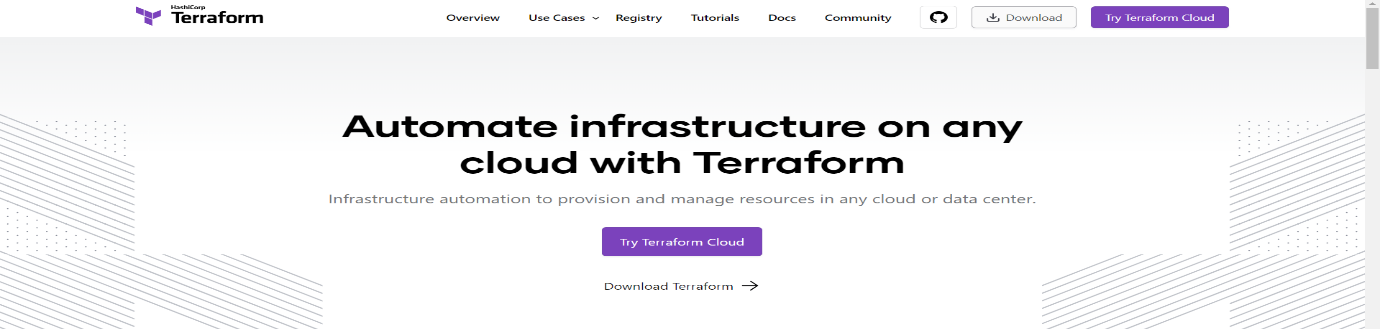
****

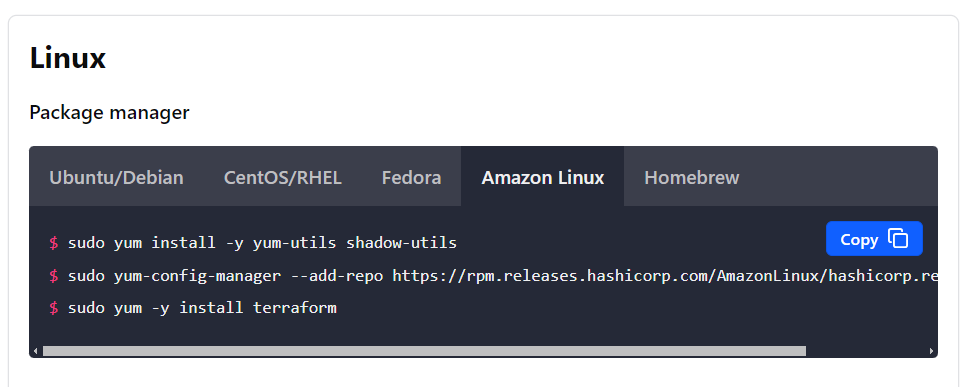
* After creation of new EC2 instance connect it with the ssh client
* Open your git bash
* Type cd Downloads\ and after that paste your ssh client before we copied in new ec2 instance
* Now we successfully launched our ec2 instance in the Linux server as shown in the figure bellow
* 

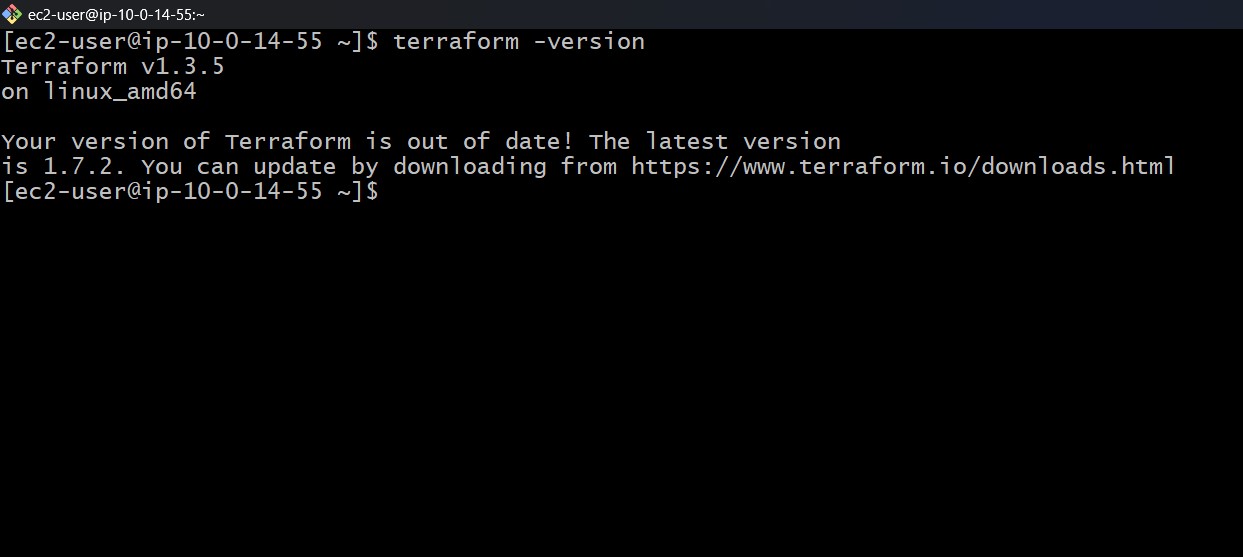
Now we have to download the terraform in the server

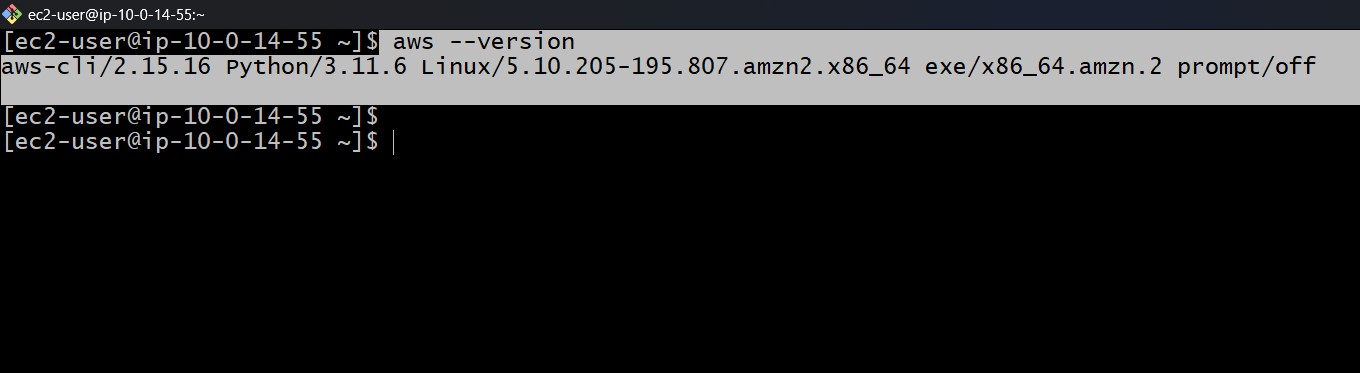
HOW TO DOWNLOAD THE TERRAFORM.?

* Go to the chrome browser search for terraform
* Go to the 1st link and download terraform in amazon Linux server as shown in fig bellow
* Copy that link and paste in the server then the terraform will download



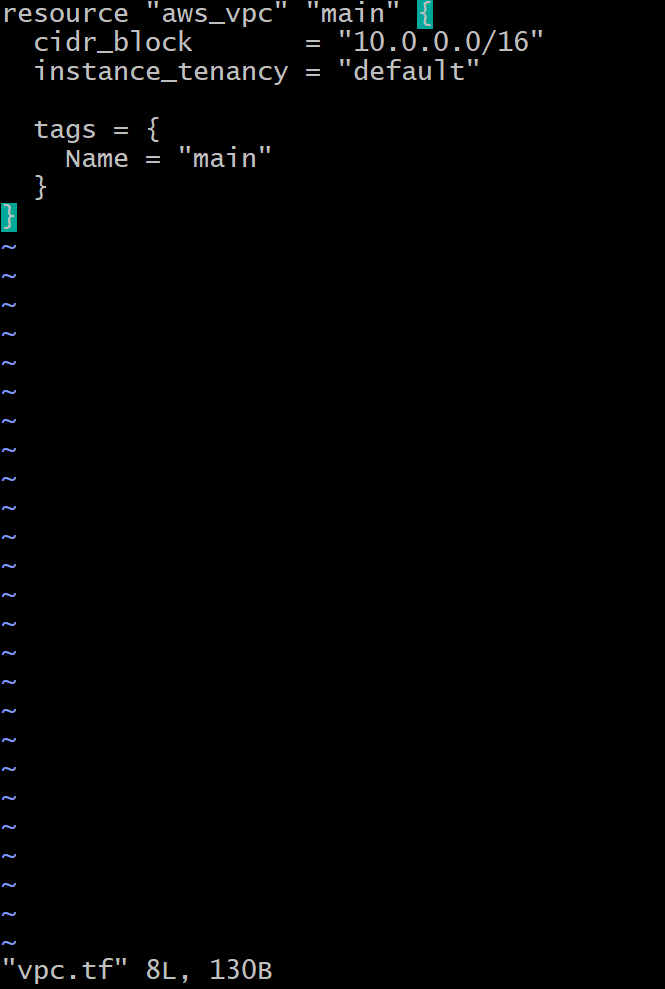


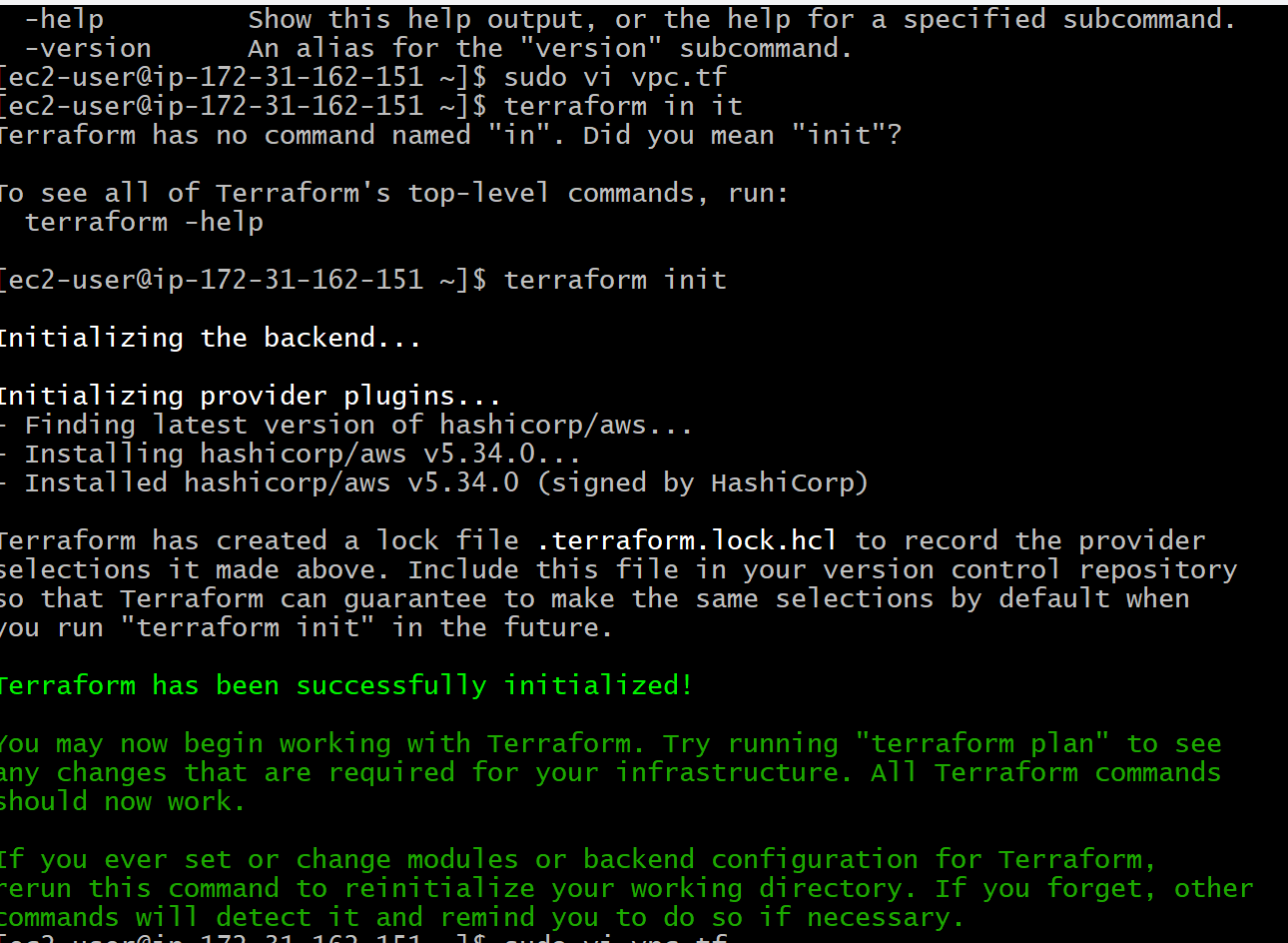




**Step -1 create a file for the v p c**

* Create a file VPC with the command “vi vpc.tf”
* Now you will enter into the file and write a script as show in fig bellow
* After that we need to initialize terraform with the command “terraform in it”
* Next we have to give a terraform plan command
* Now its shows error as shown in fig below because access denied so we need to provide access key and secret key by creating provider file

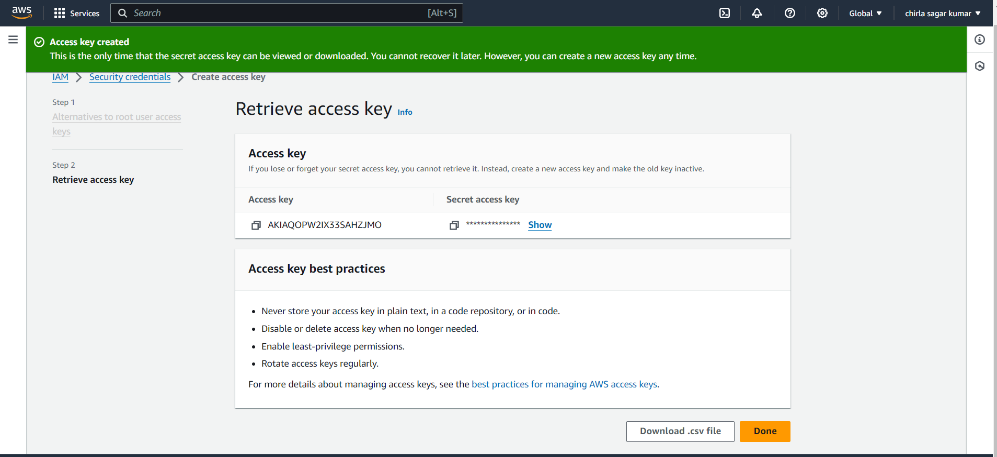


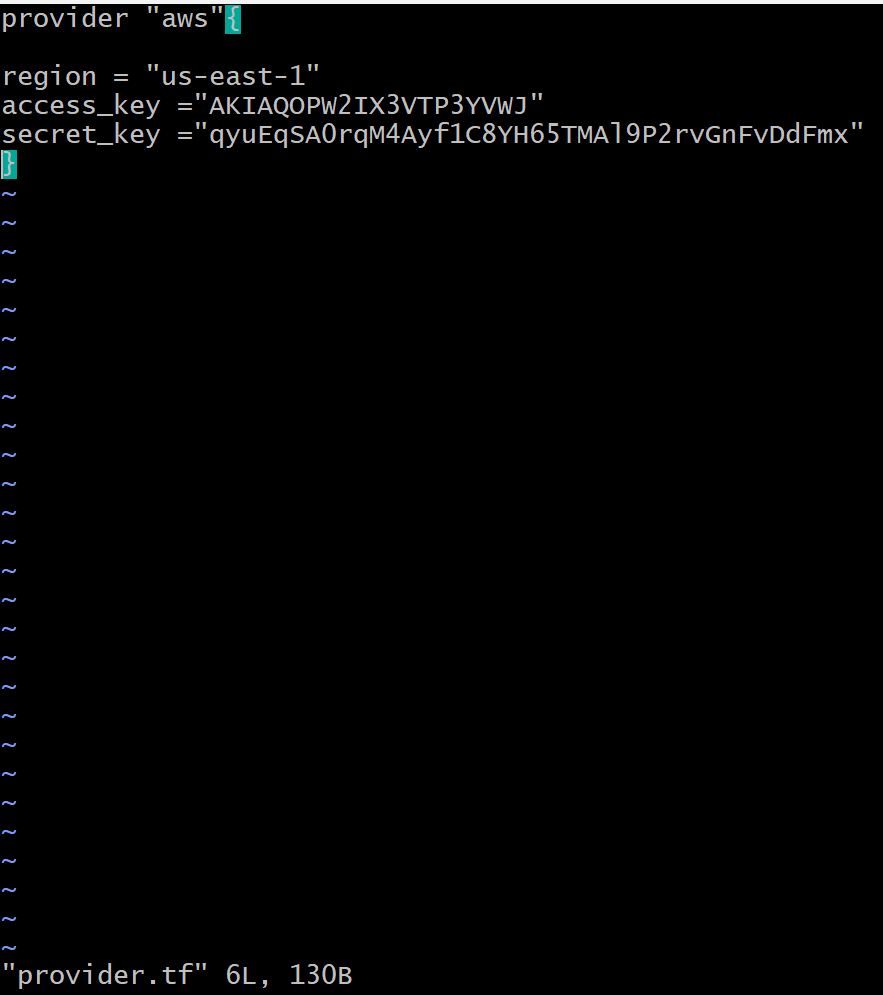


In the provider file we need to give the access key and security key

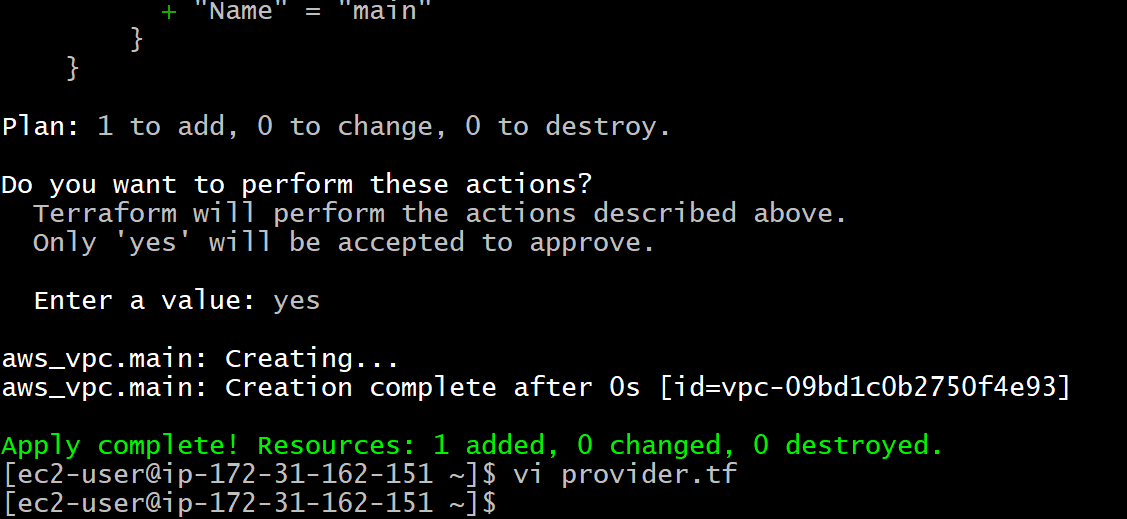
HOW TO GET ACCESS KEY AND SECURITY KEY?

* Go to the AWS console account search for IAM
* Go to security credentials click on create access key
* We need to use case in CLI and agree to all terms and conditions
* Give the tag value next you will find the access key and secret key as shown in fig below.
* Copy that access key and secret keys and paste in the created provider file and now click the command terraform validate

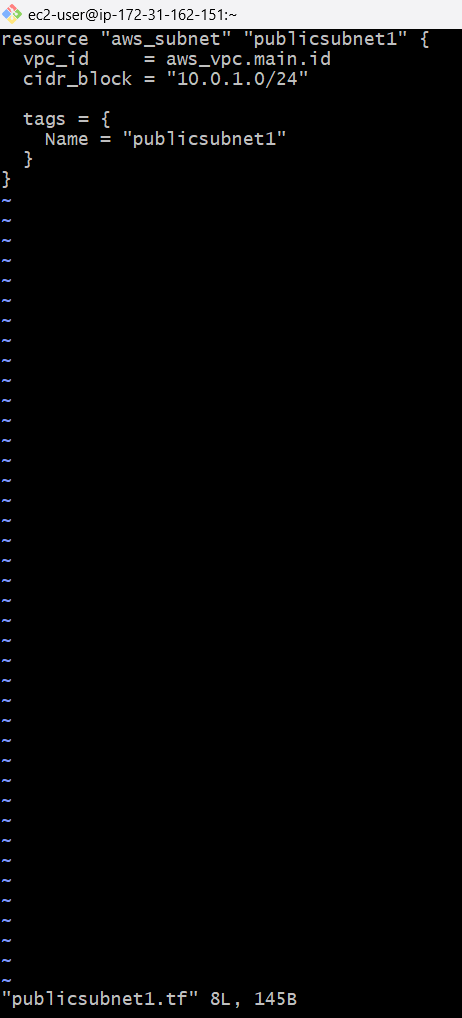


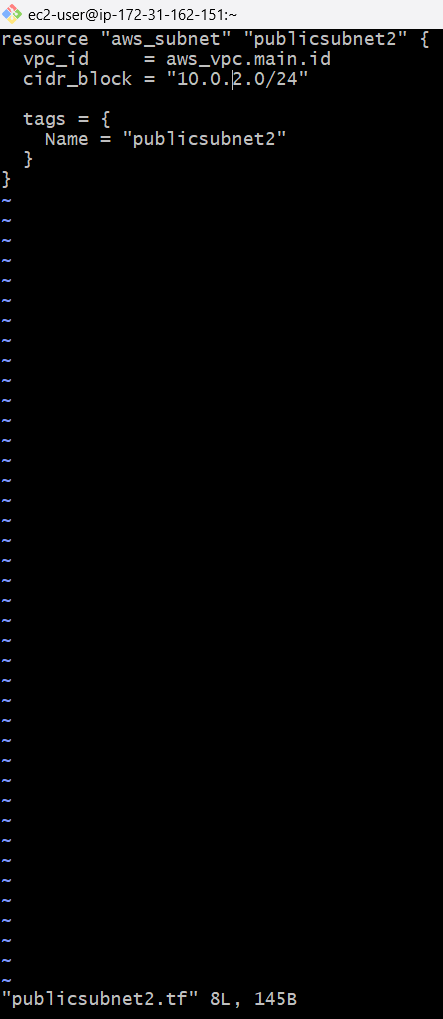


* After gives the security and access keys again give terraform apply command then it will work as shown in the figures below
* We can also check manually whether the VPC is creates or not when we see its was created manually as shown in fig bellow

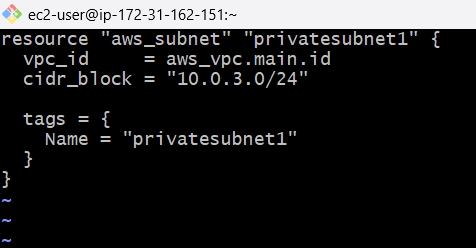
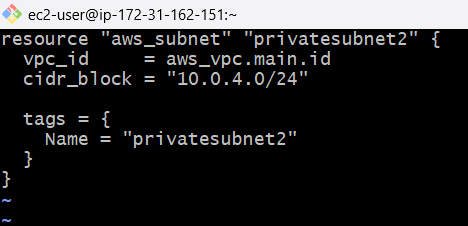


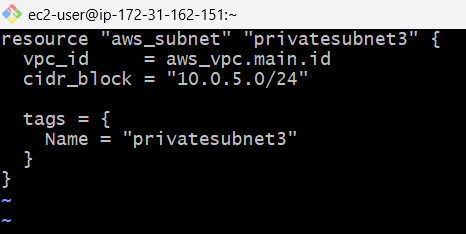
**Step -2 creating a file for subnet**

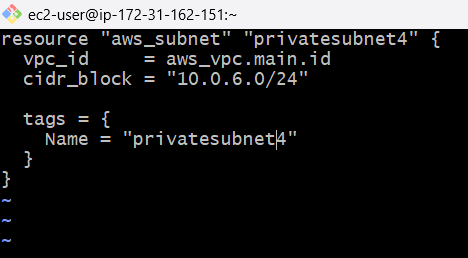
* For this deployment we need to create 4 private subnets and 2 public subnets
* We have to create each subnet file individually so that if any error occures we can find easily
* Create a file for the public subnet1 with the command vi publicsubnet1.tf
* After we enter into that file then write the script as shown below
* After creating 1st subnet check the validation with terraform validate command and next apply the terraform with terraform apply command
* With same process create another public subnet file and add the scrip as shown bellow



* For the private subnets create a private subnet file and add script as shown below
* We have to change the CIDR block number for each subnet
* After creation of each subnet whether private or public we need to validate the terraform and apply terraform so that we can know any errors are came or not

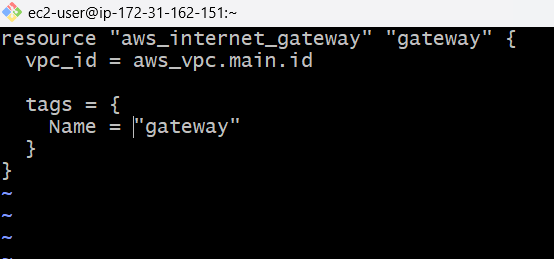




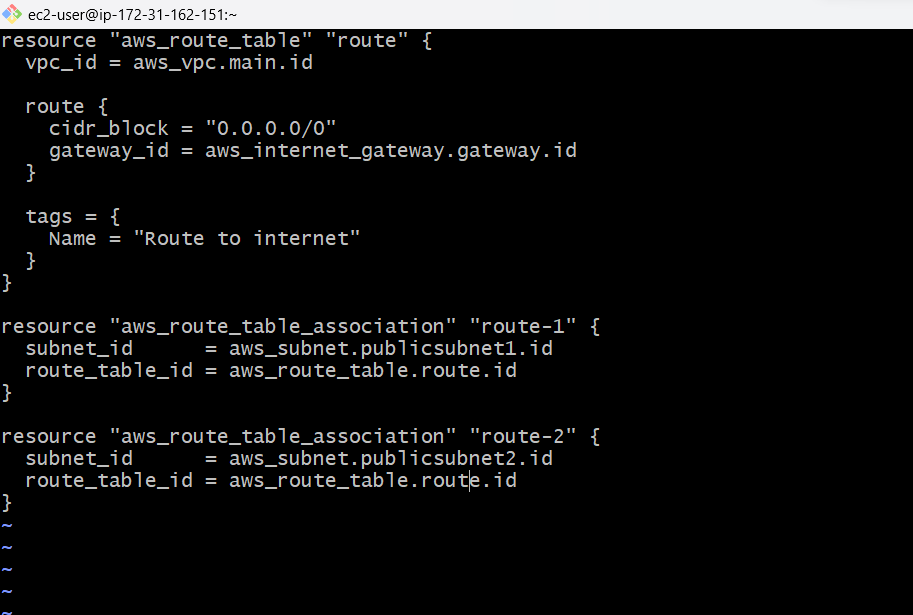


**Step -3 create a file for the internet gateway**

* We need to create internet gateway file with the command vi internet gateway.tf
* After that we need to give the script as shown in fig below
* After the enter script same as before validate the terraform and plan terraform commands

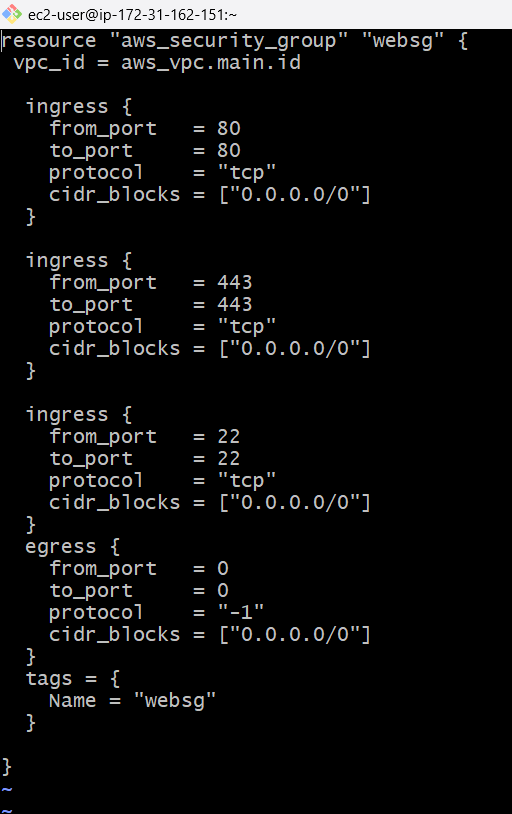


**Step -4 creating a file for the route table**

* We need to create a file for the routable by using vi routable.tf
* After creating file then we need to add script to it as show in fig below
* After adding script save it and again validate the terraform and plan terraform run those commands

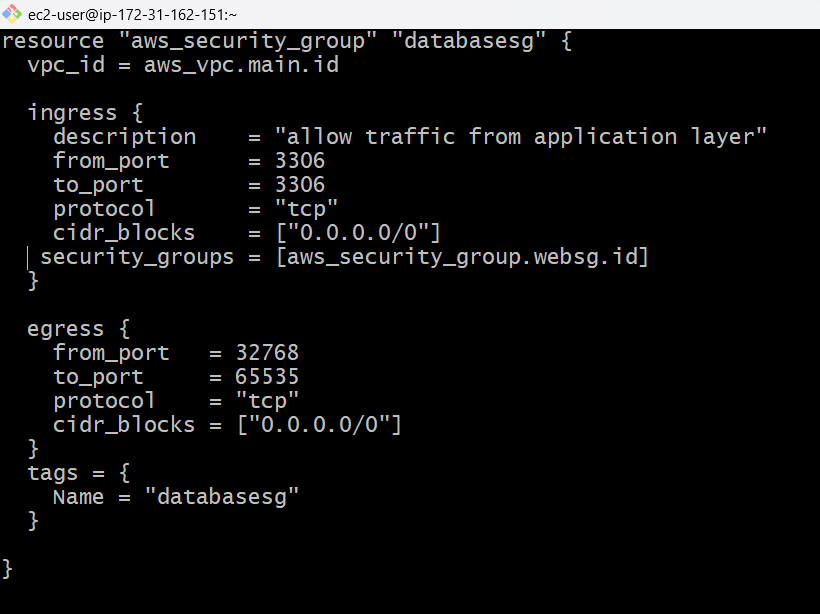
**Step -5 creating a file for the security group for frontend tier**

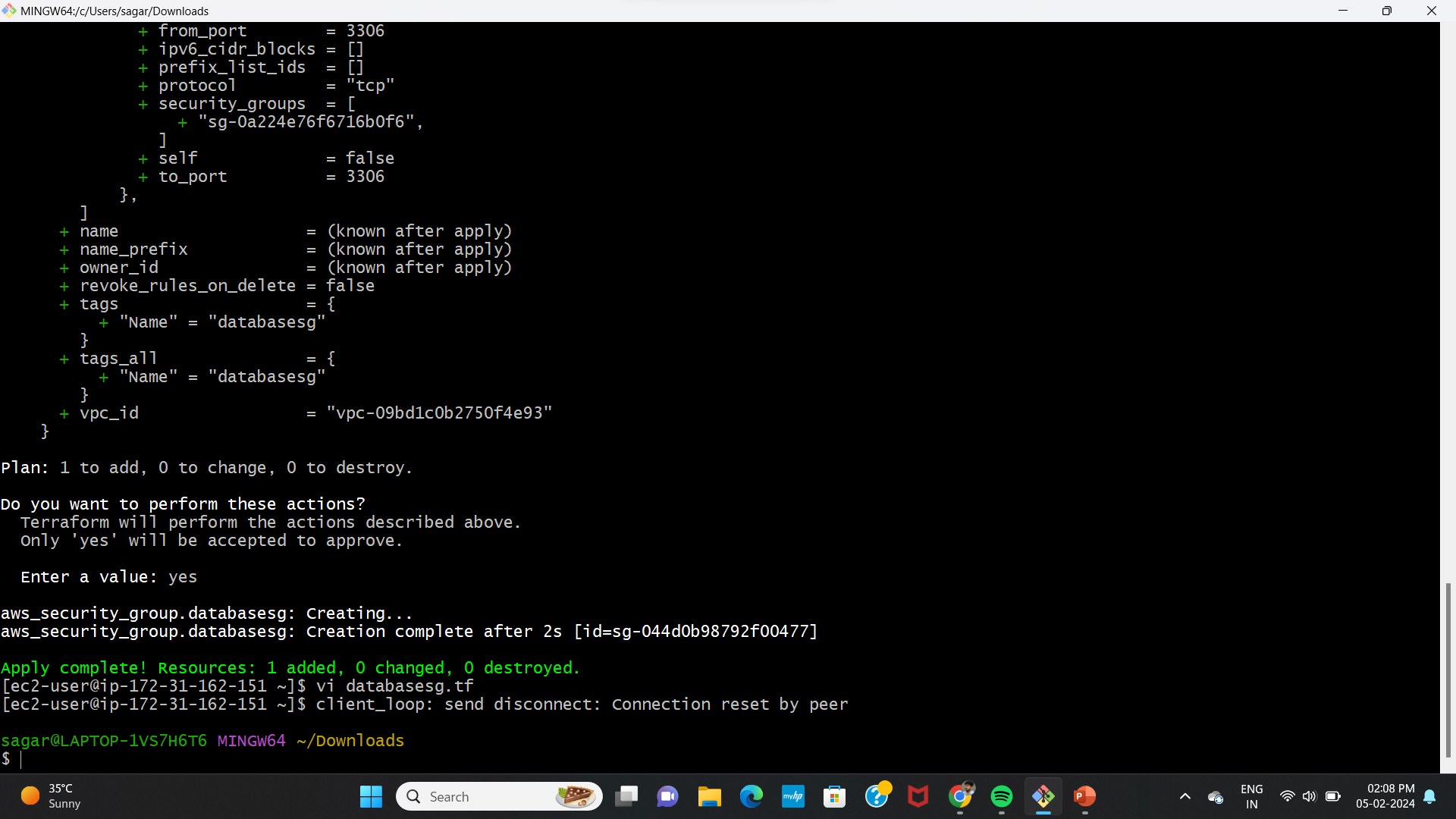
* We need to create a file by the vi web sg.tf and after that give a script for that as shown in fig below
* After that we have to validate and plan the terraform as before



**Step -6 Creating a file for the security group for database tier**

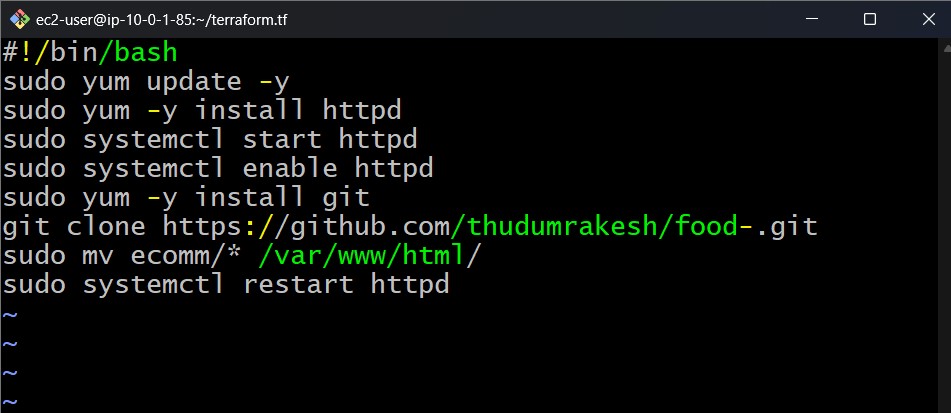
* We need to create a file for the security group for database tier
* For that we need to create one databasesg.tf file and add the script as show below
* After that we need to save and validate and plan terraform

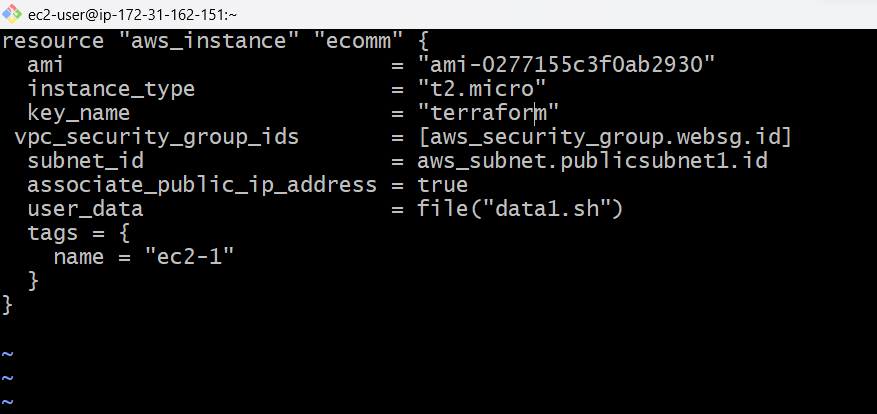
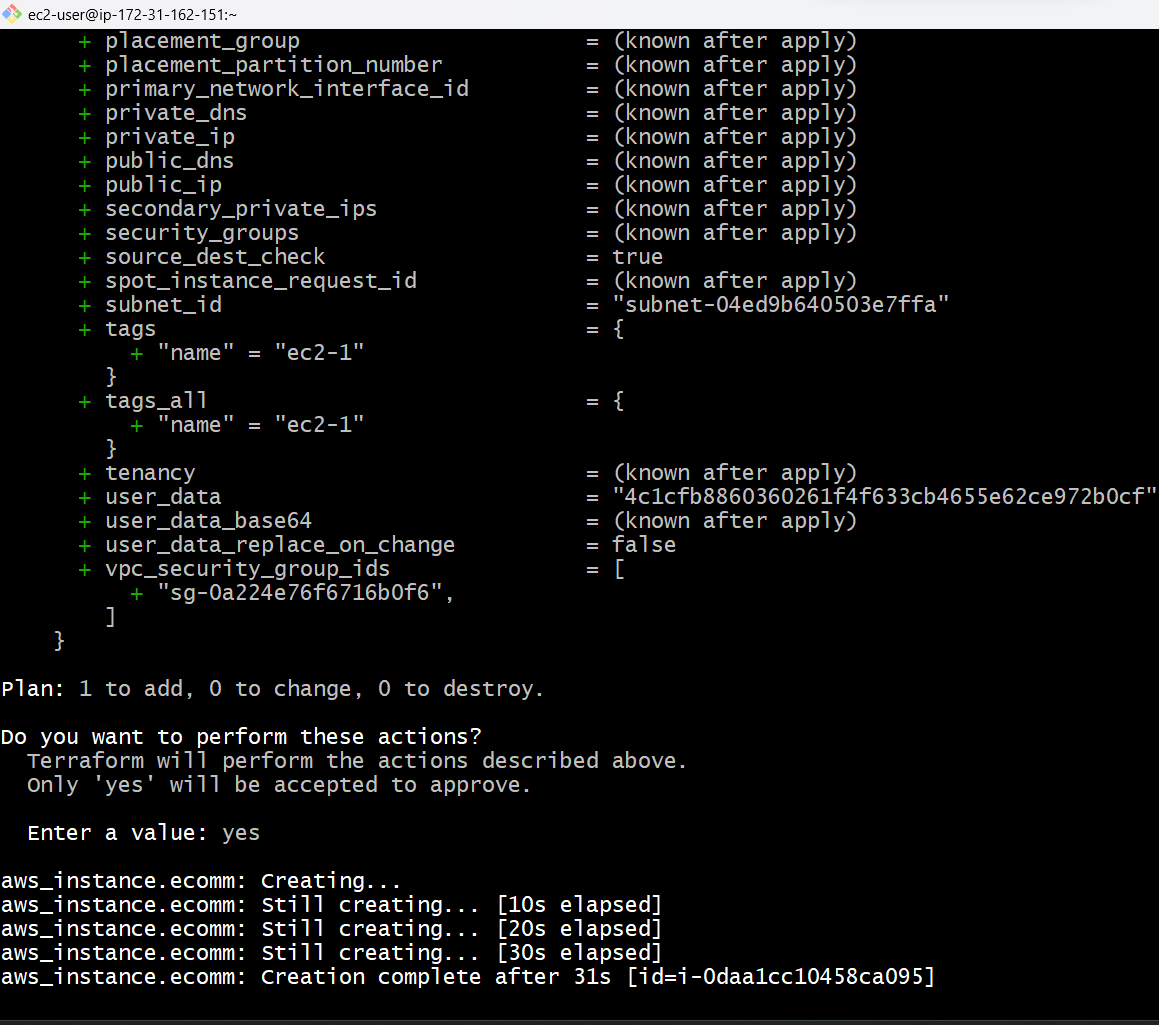


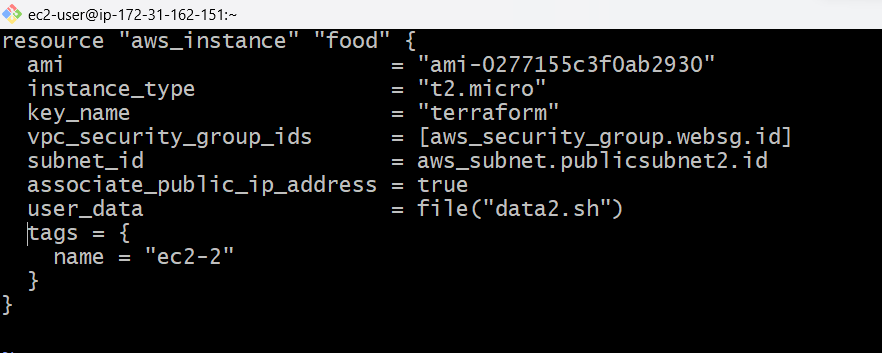
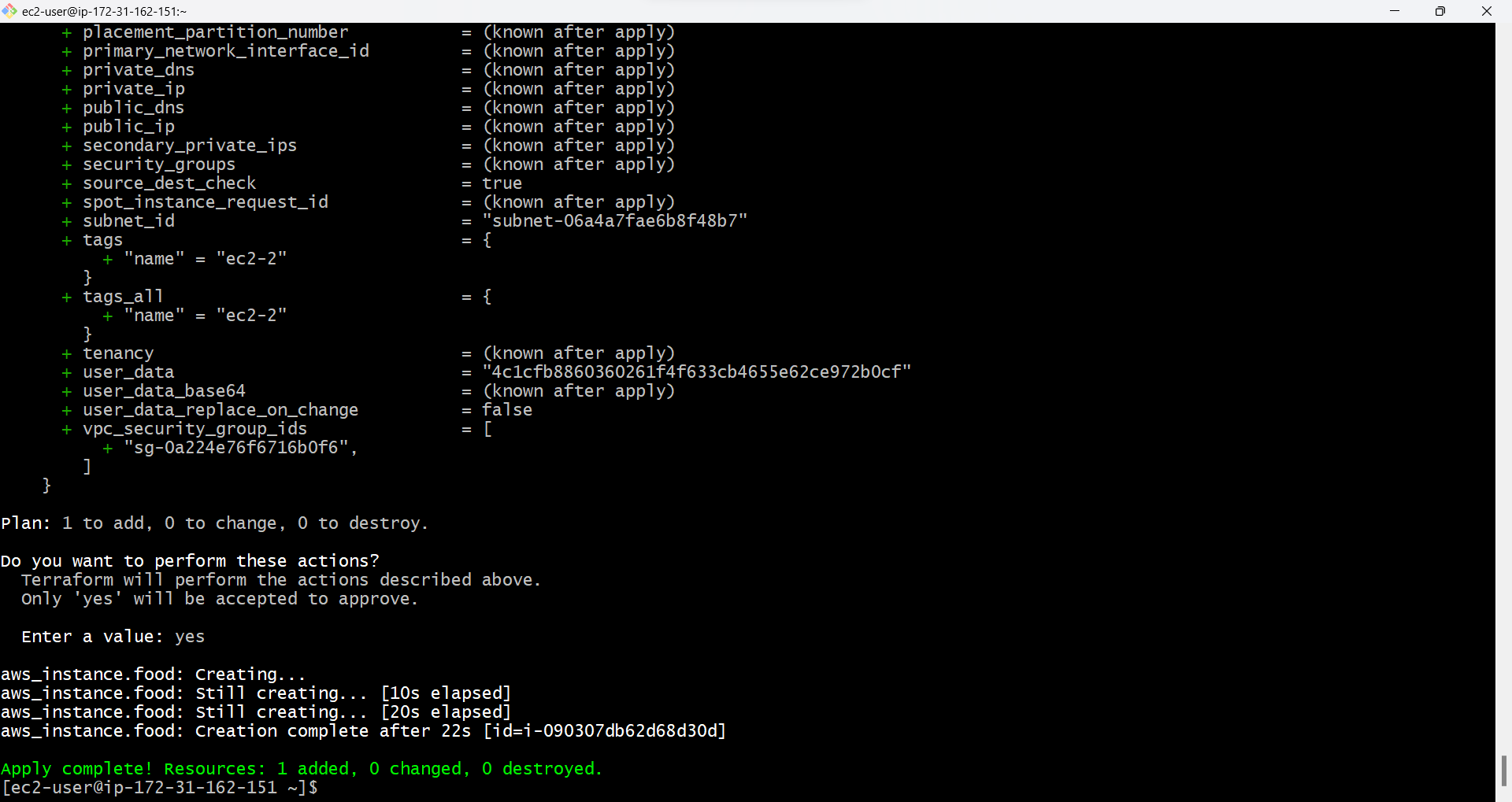


**Step -7 Create a file for the ec2 instance**

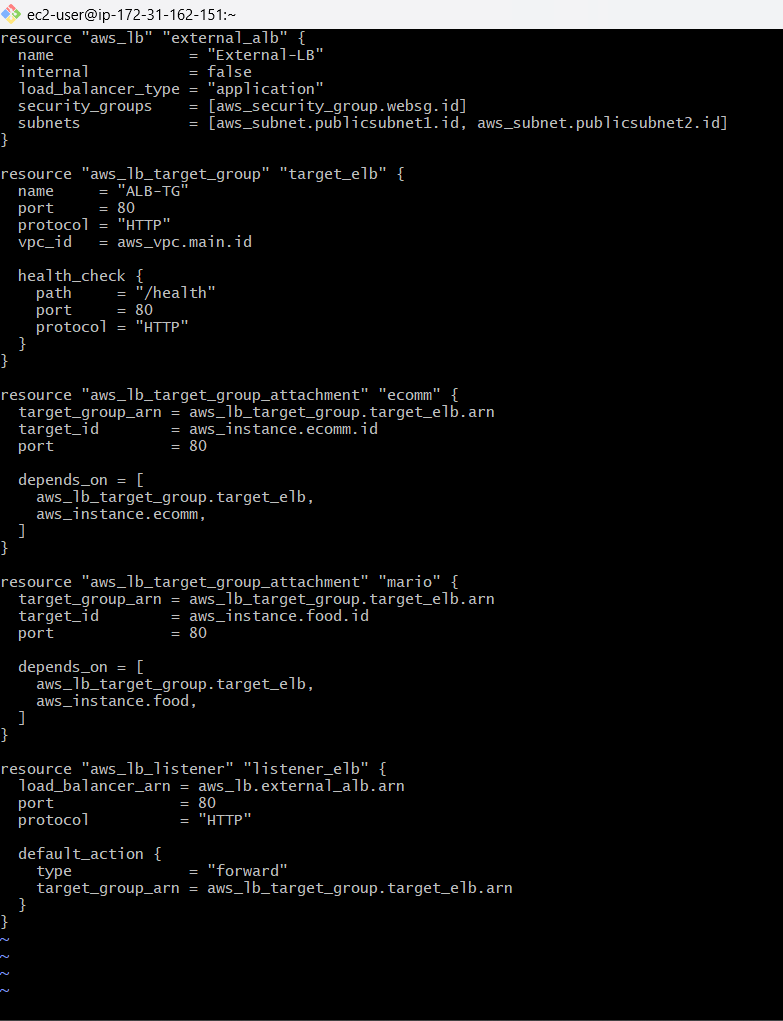
* Create a file for the ec2 instance with the command vi ec2-1
* We have to create2 instances for this another instance with ec2-2
* After creation of the 2 ec2 instances we need to give data for deployment so that we need to create a data 1.sh for 1st instance and data2.sh file for 2nd instance
* After creation of those data files add the below given bin bash script to it and saveit

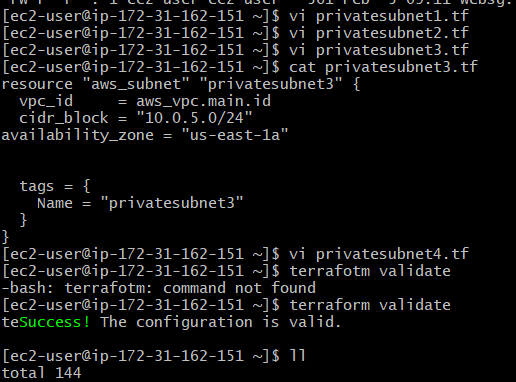


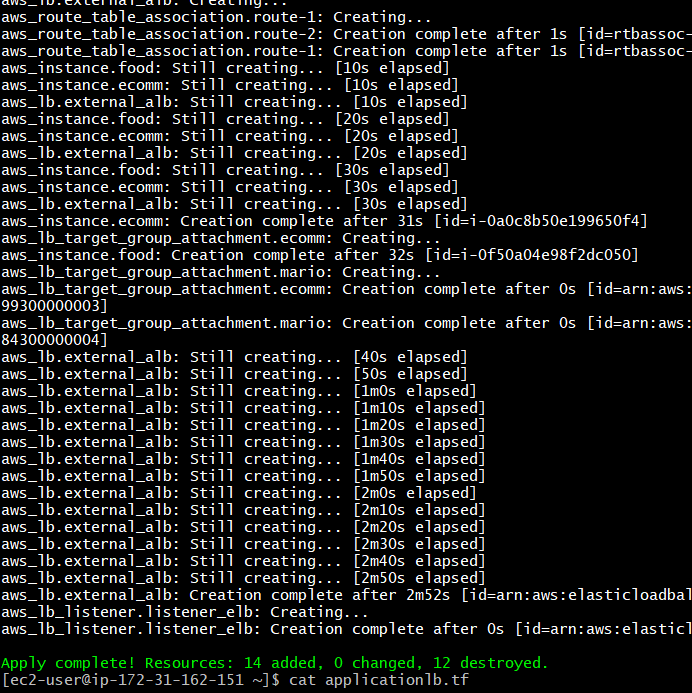


* Creation of 2nd instance follows like given bellow

**Step -8 create a file for the application load balancer**

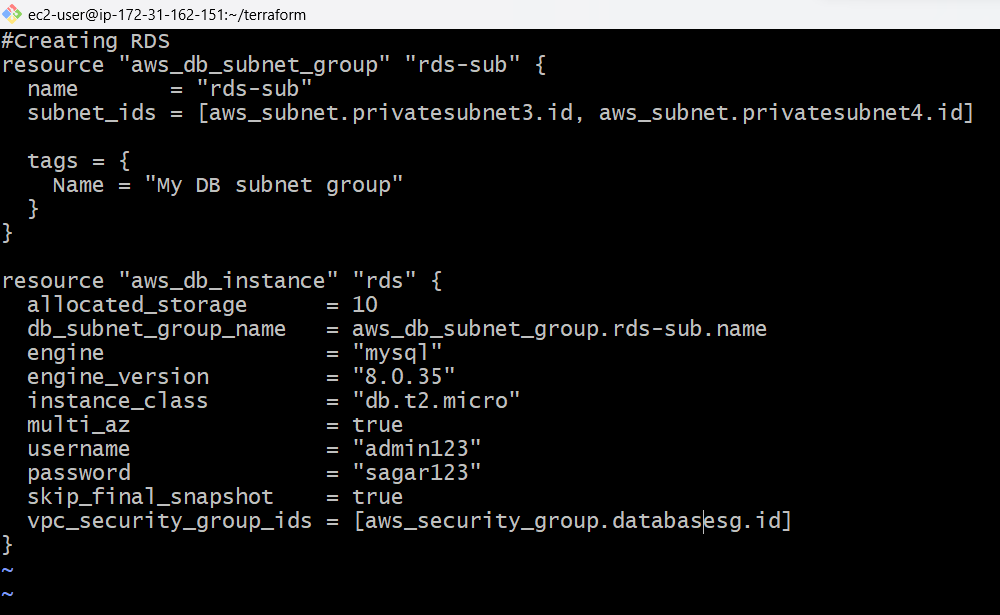
* Create a file with name application LB
* After finishing script validate and apply terraform so that the load balancer will creates





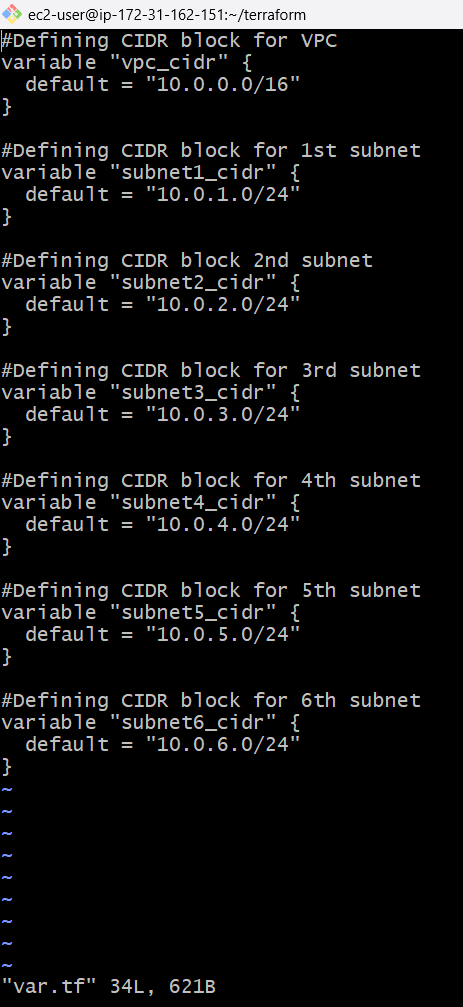
**Step-9 create a file for RDS**

* Create a file with the name as RDS with the command vi rds.tf
* Give the bellow script tin the file and save it and validate terraform and apply terraform



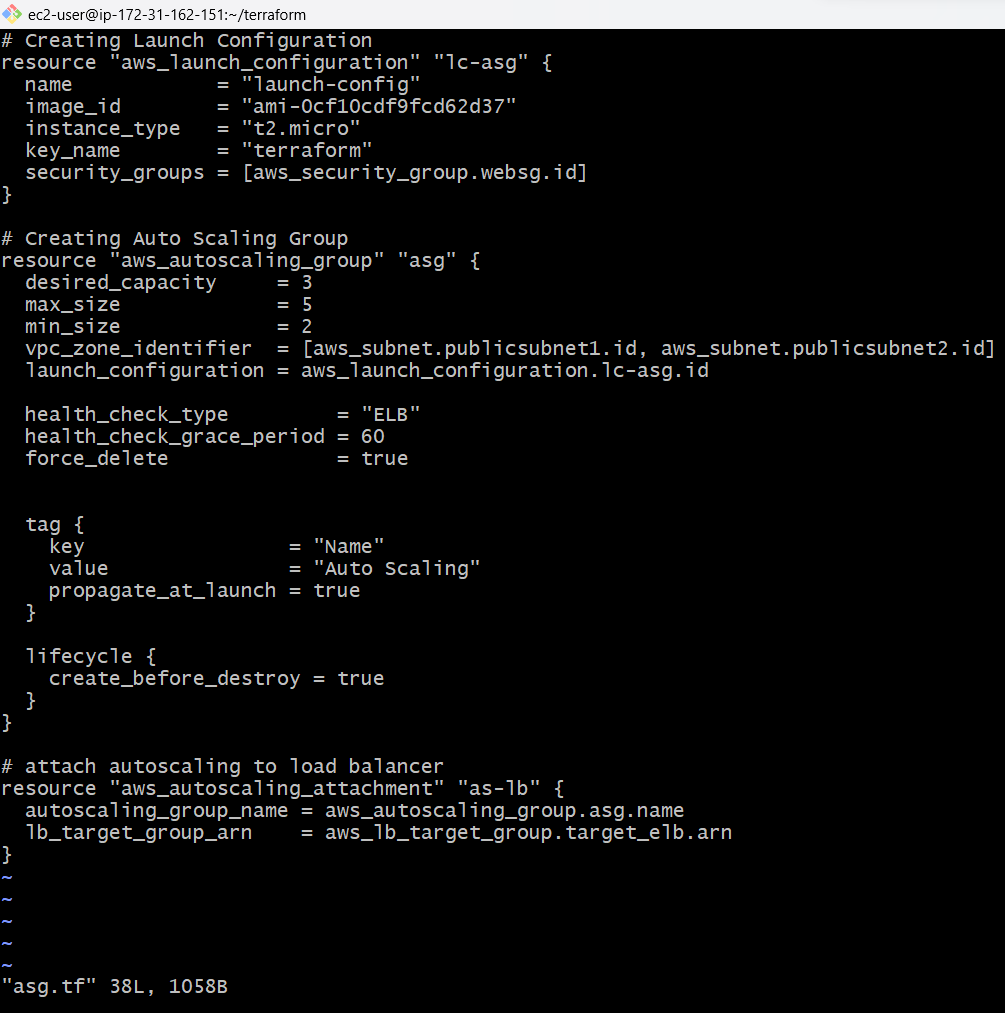
**Step-10 create a file for the variable**

* Create a variable file and give the script shown bellow
* After that validate and apply the terraform

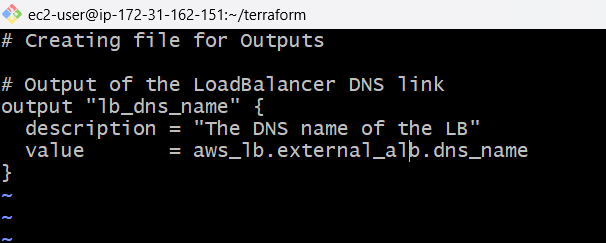
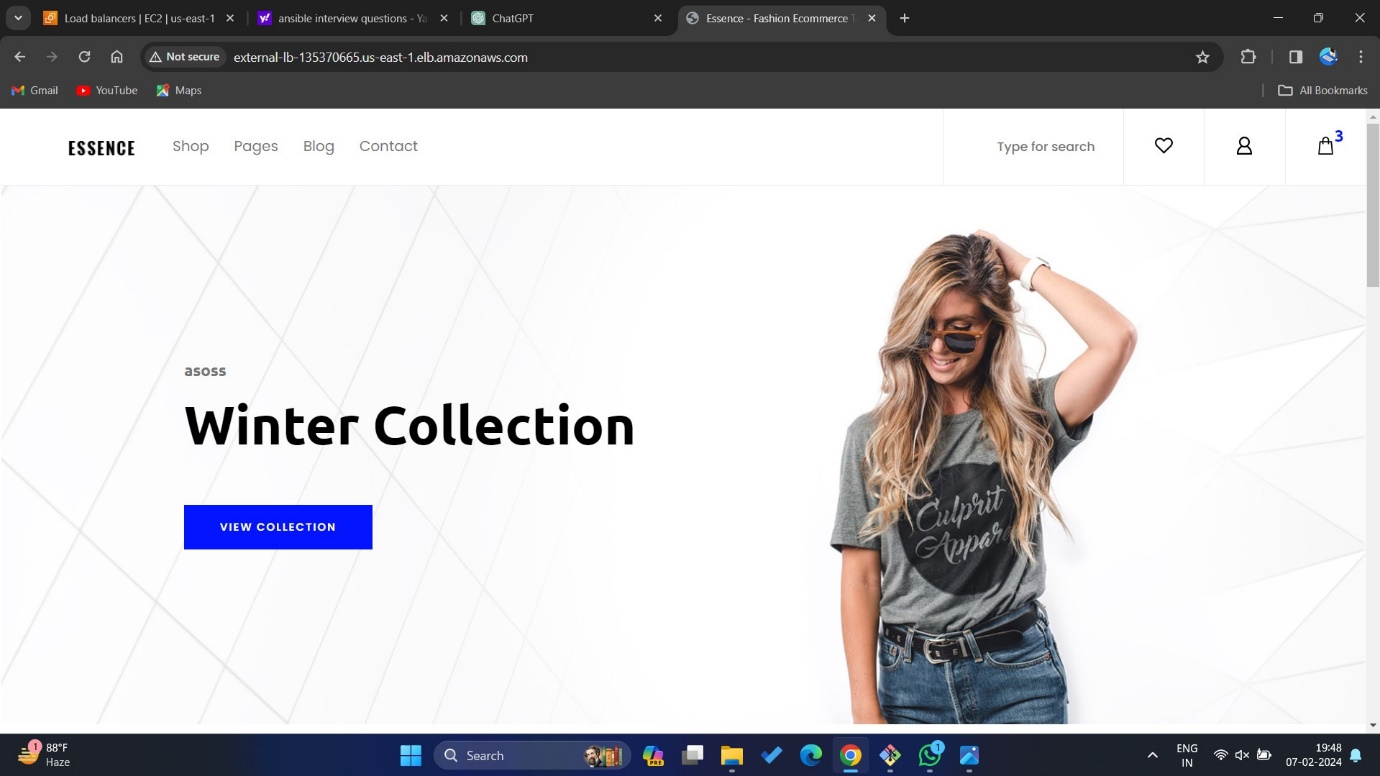


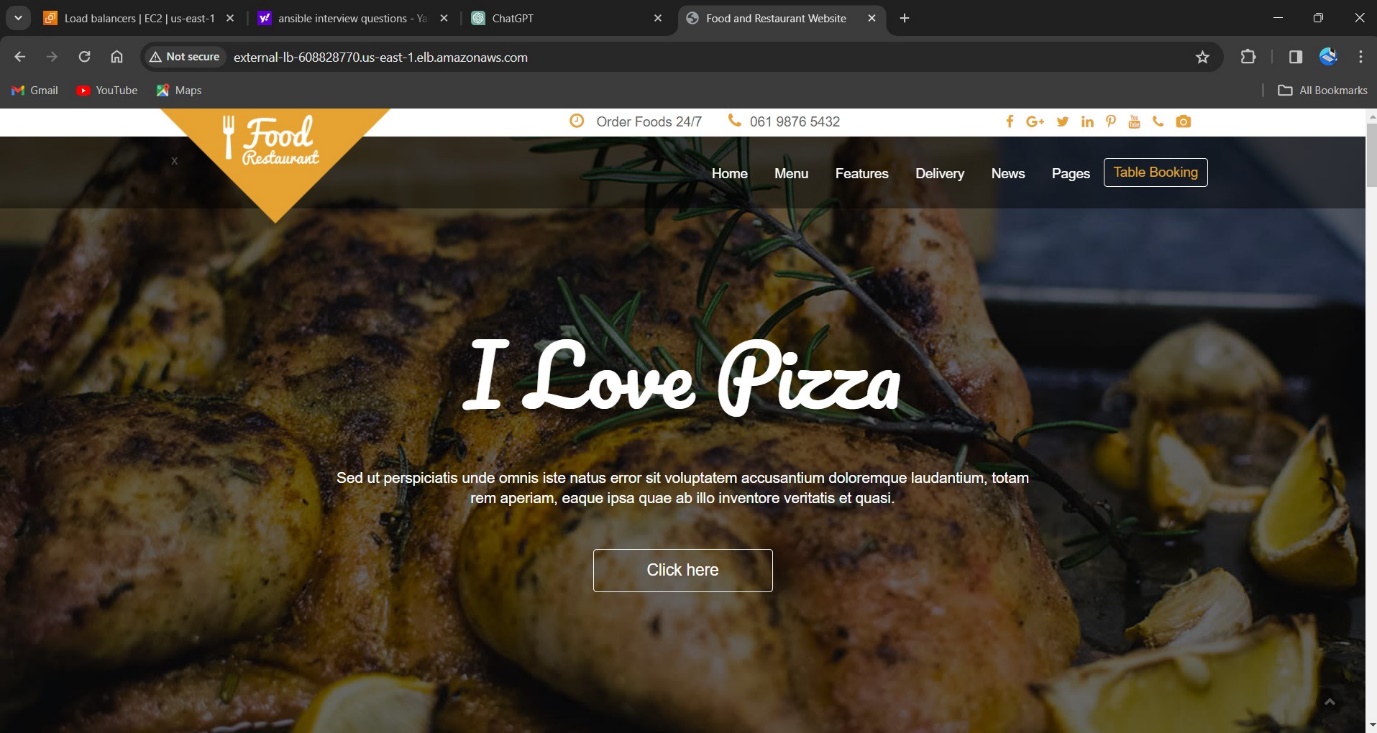
**Step-11 create a file for the auto scaling group**

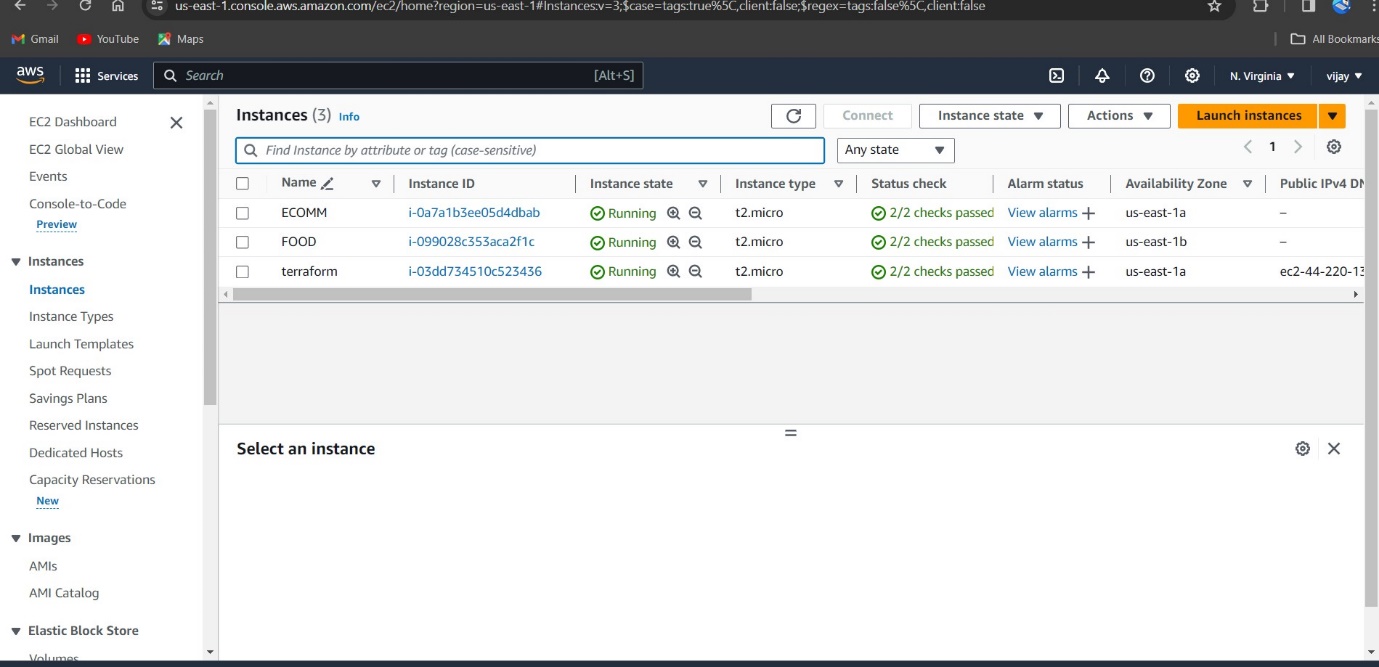
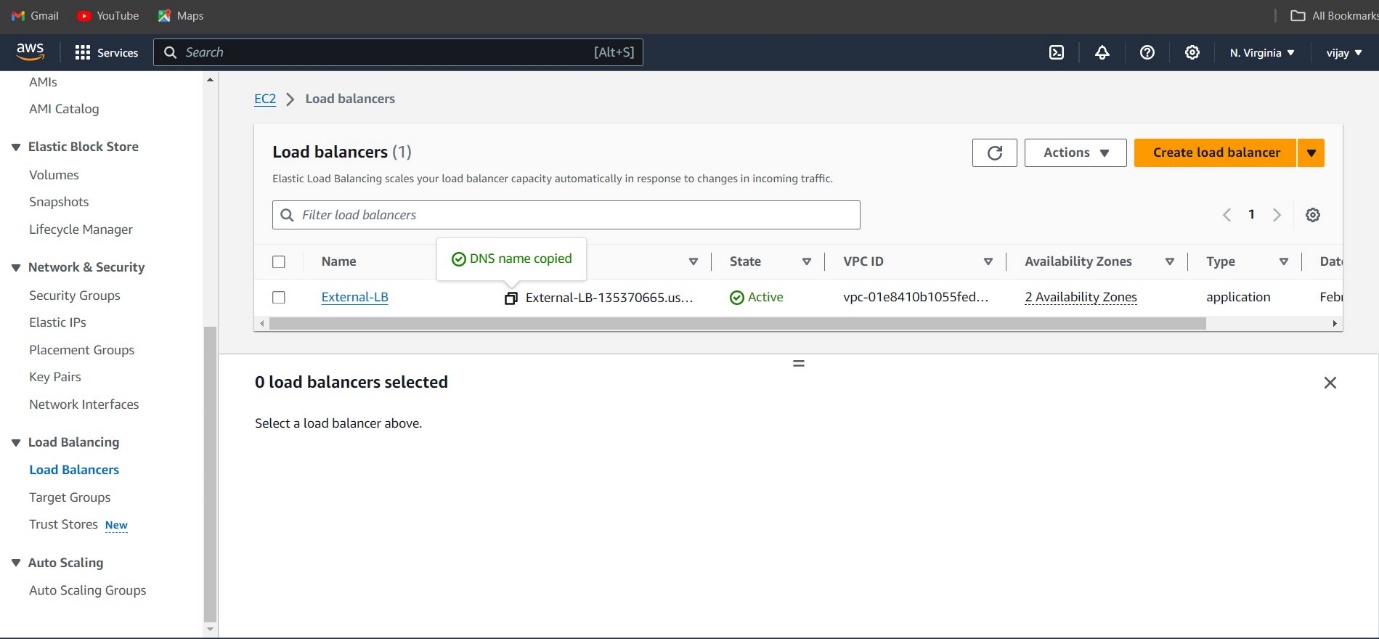
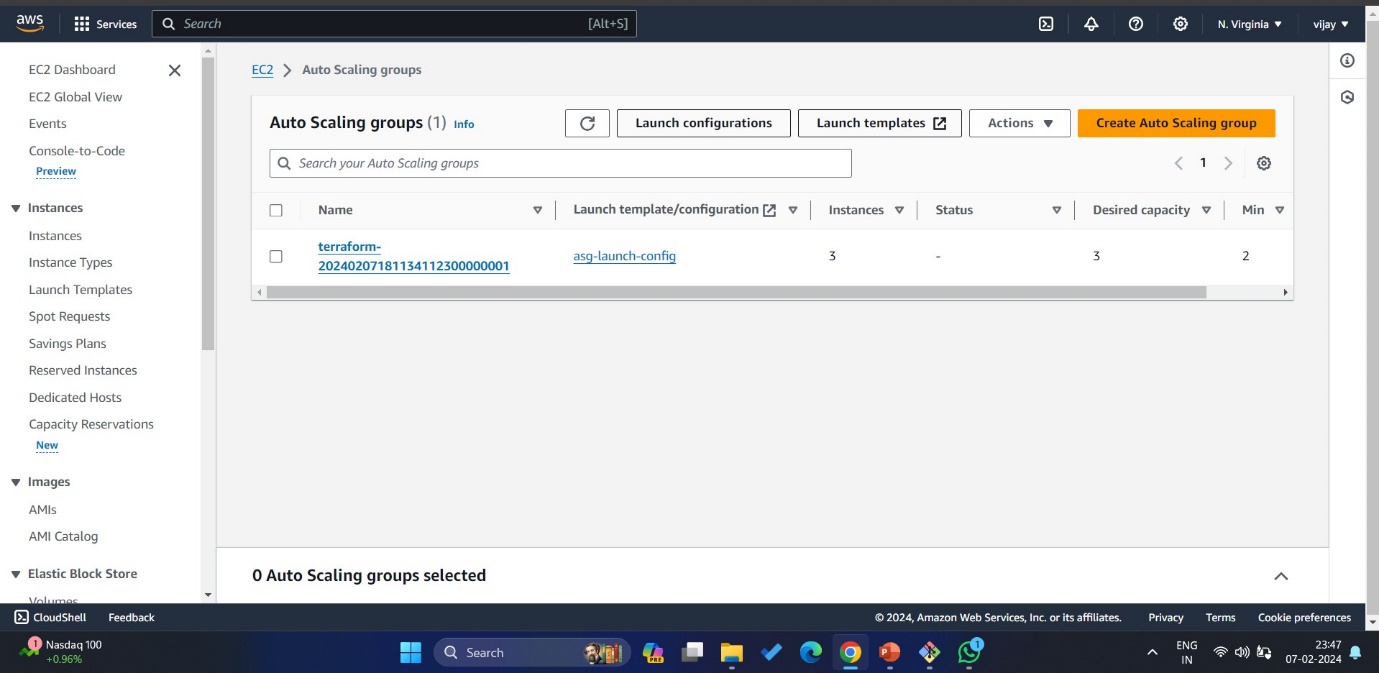
* Create a file for autoscaling group
* Add the script as shown in fig bellow after that we need to save and validate and apply terraform

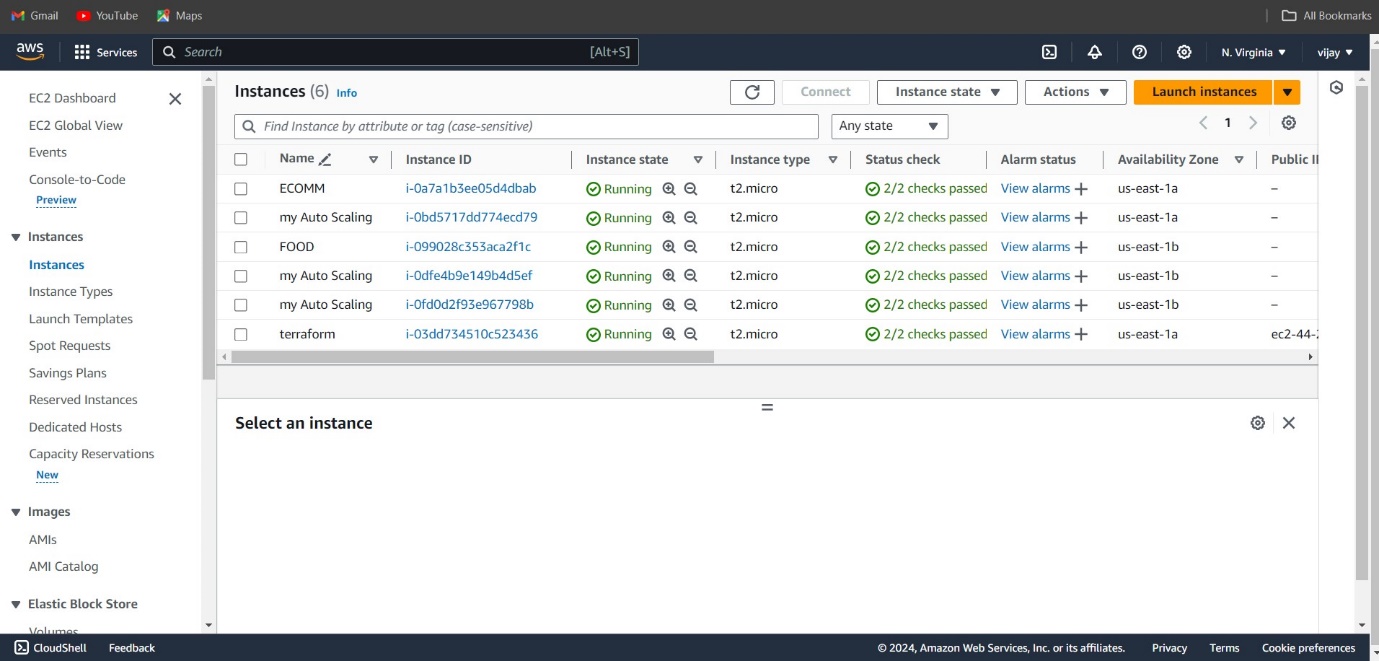


**Step-12 create a file for the output**

* Create a file as name output.tf and give the script as shown bellow
* After that validate and apply the terraform
* 







**THANK YOU**