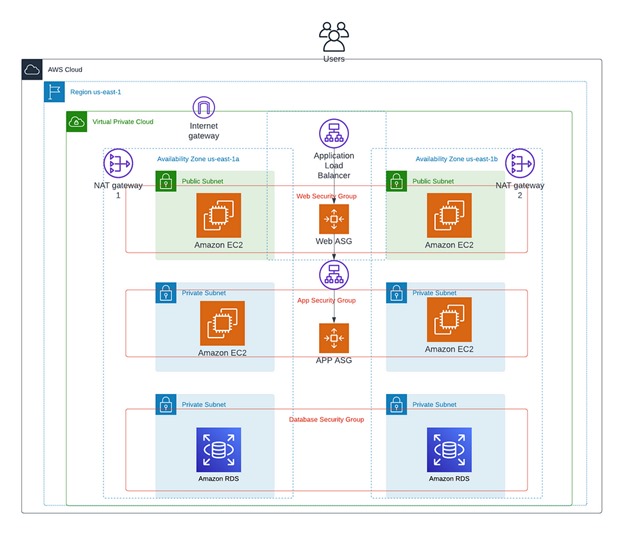
# RAM CHARAN

# 125-Batch

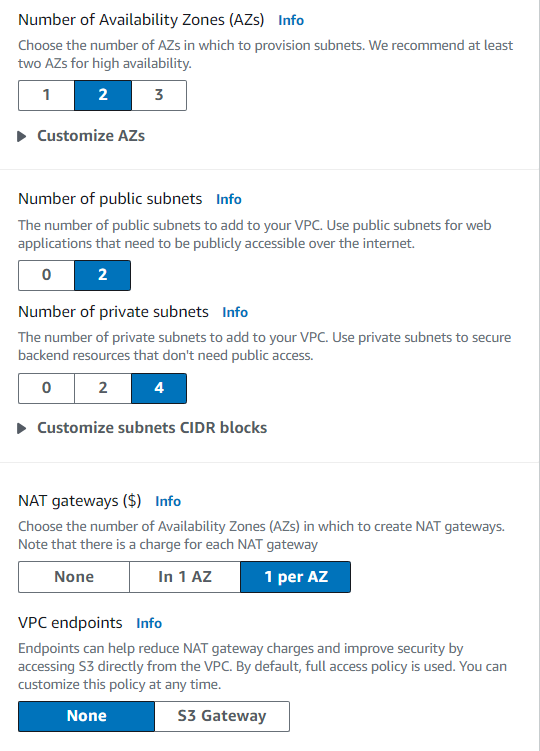
# 6304335587

# 1-PROJECT

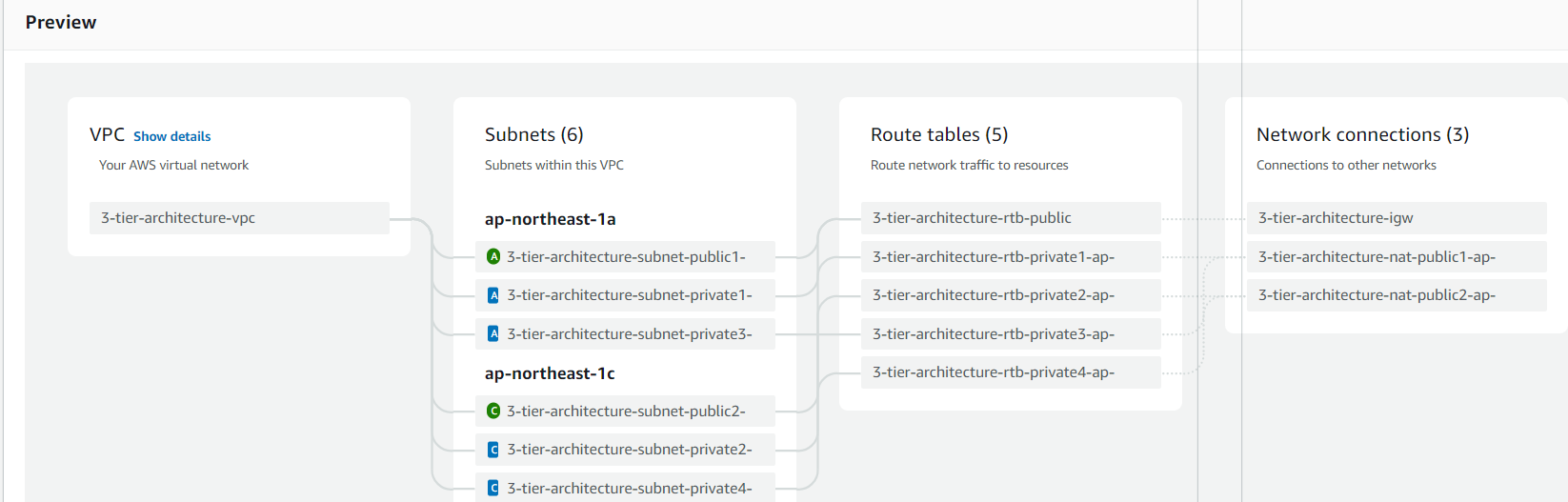
# 3-TIER ARCHITECTUR



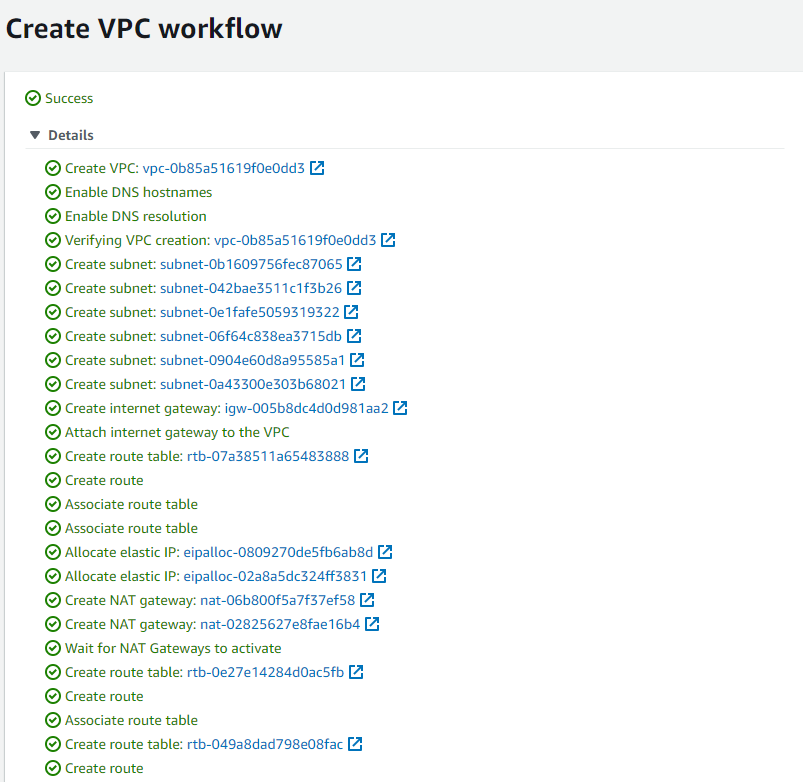
* First create VPC (or) More VPC
* Give the name 3-tier-VPC
* Give the IP address 10.0.0.0/16
* All press No IPv6 CIDR block
* Give the number of availability zone (AZs)
* There are two availability zones in the given diagram 1A-1C
* Number of public subnets -2 as per given in the diagram which are front-end
* And number of private subnets-4 in which 2-private-subnet are for Backend & another 2-private-subnet are for Data-base which means RDS (relational data-base service
* Give the NAT gate per 1-Zone because we have 2- private-subnet in two zones 1A-1C



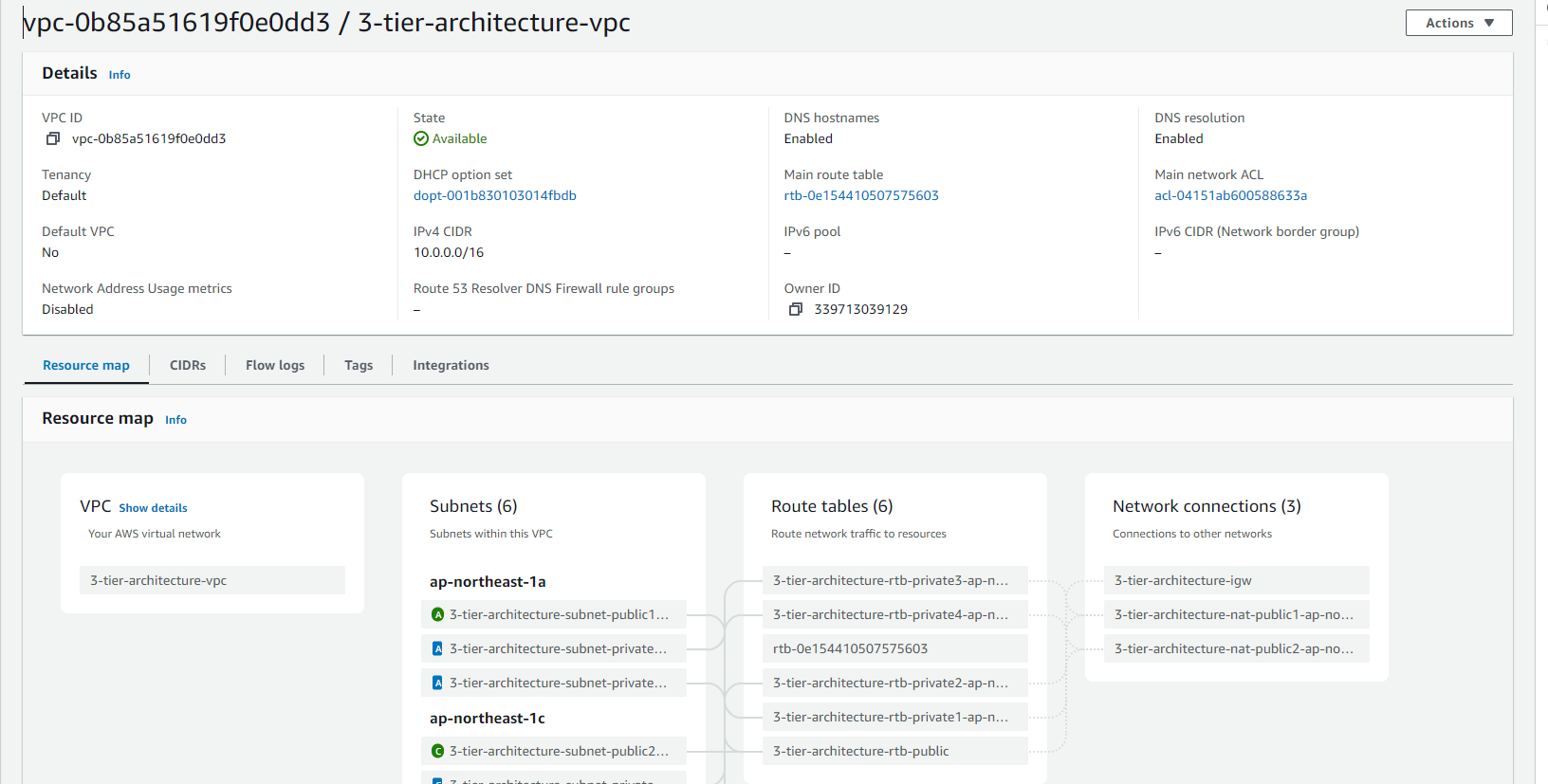
* Give the VPC endpoints – None
* Finaly see the more VPC diagram



* And create more VPC

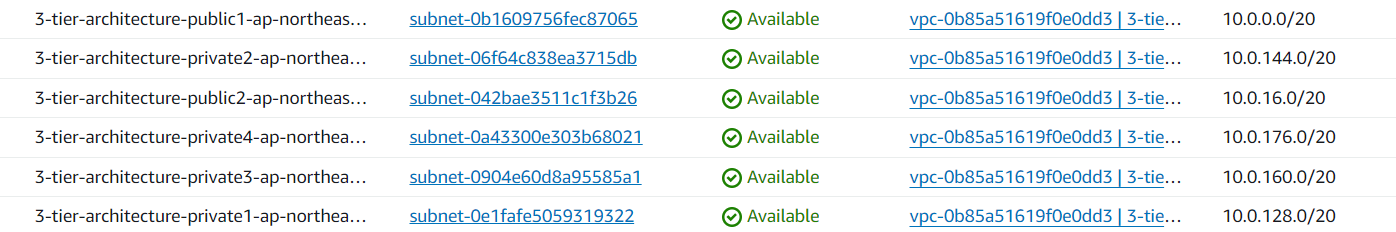


* Click on view VPC & VPC name 3-tier-architectur



# CIDR VALUES OF 6-SUBNETS

* Public-subnet-1a-frontend = 10.0.0.0/20
* Public-subnet-1c-frontend = 10.0.16.0/20
* Private-subnet-1a-backend = 10.0.128.0/20
* Private-subnet-1c-backend = 10.0.144.0/20
* Private-subnet-1a-Database = 10.0.160.0/20
* Private-subnet-1c-Database = 10.0.176.0/20



* Next, we should create4 - ec2 instance,2 - instance for public subnets in frontend,2 - instance for private subnets in backend

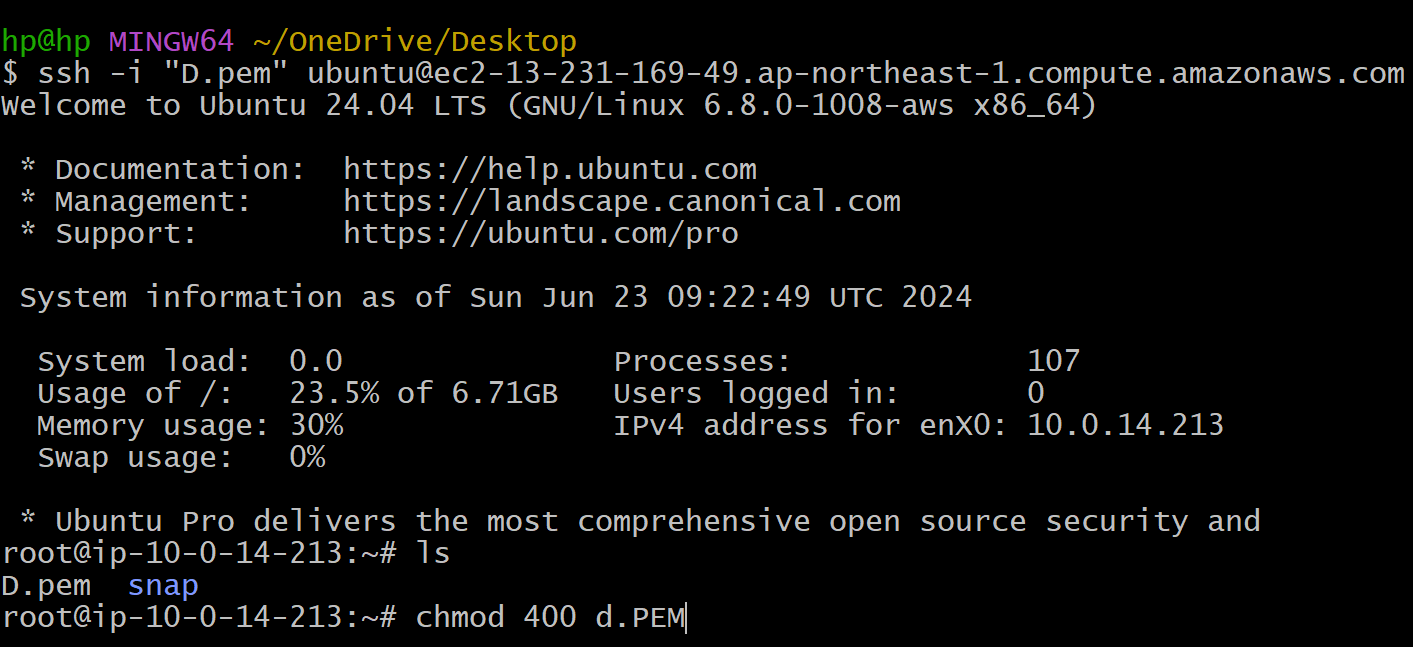
# EC2-INSTACE CREATION

* Create 2-(public-ec2) instance for two public subnets in the frontend, 1a-1c zones which are ec-1-public, ec-2-public with two public subnets which are given below & give the VPC which is created (3-tier-architectur)
* Mainly create security name-(sgroup) with inbound rules (SSH-port number-22), (HTTP-port number-80)

(MYSQL, AURORA-port numer-3306),

(custom – 0.0.0.0/0)

* Add the same security group to all the four instance
* Enable Auto-assign public IP
* 3-tier-architecture-public1-ap-northeast-1a-frontend
* 3-tier-architecture-public2-ap-northeast-1c-frontend
* Now create 2-(private-ec2) instance for two private subnets in the backend, 1a-1c zones which are ec-3-pravite, ec-4-pravite with two private subnets which are given below
* 3-tier-architecture-private1-ap-northeast-1a-backend
* 3-tier-architecture-private2-ap-northeast-1c-backend
* Check the connection b/w ec1-public to ec3-pravite



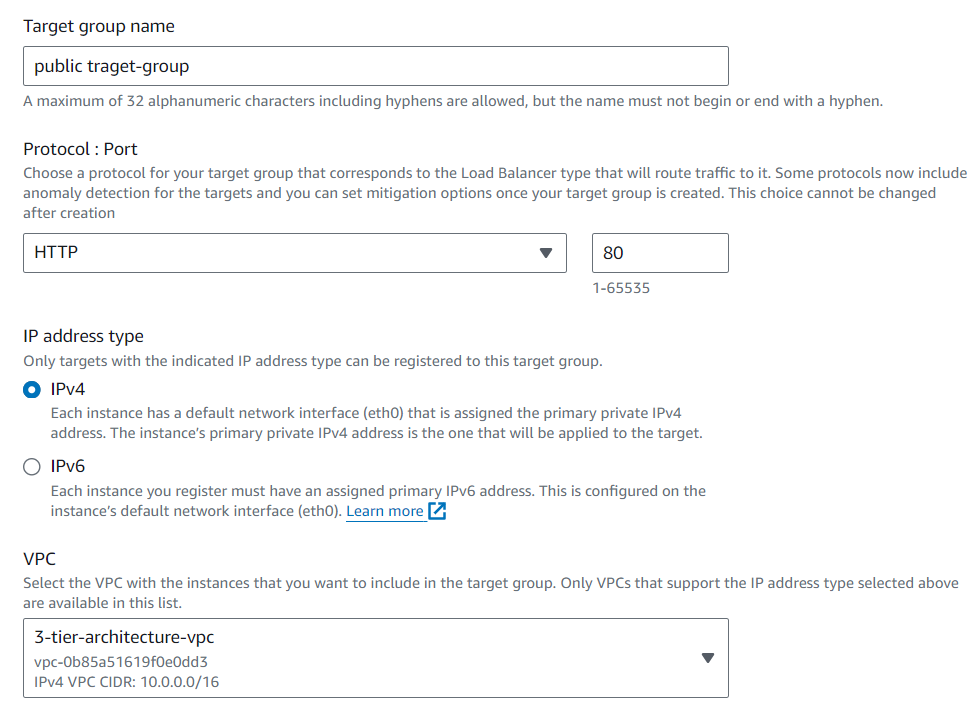
* After connecting to ec1-public to server in Git-bash create the Pem file in the root user. Copy the private Pem key and give the permission (chmod 400 (with Pem file name) & press entre it connects to UBUNTU private instance
* Copy the SSH code of private instance & edit IP address and copy the private IP of private instance

# CREATE 2 AUTO-SCALING & 2 LOAD BALANCER & 2 TARGET GROUP & 2 LANUCH TEMPLATES FOR (2-public & 2-private instance)

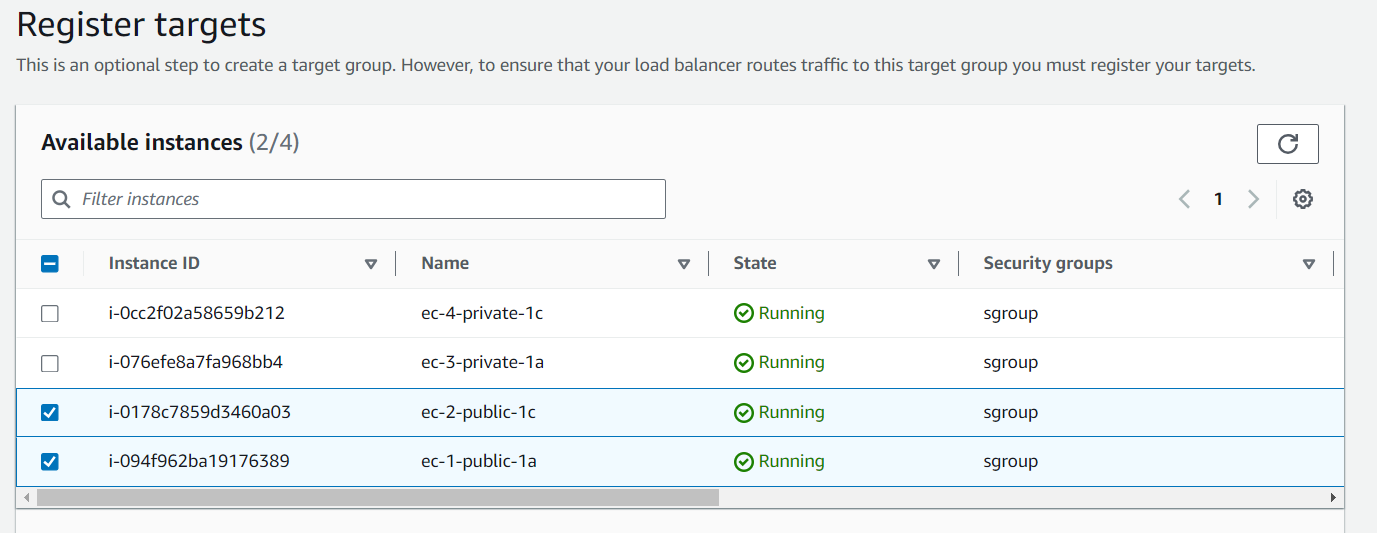
* First create target group for 2 public instances

(public target group)

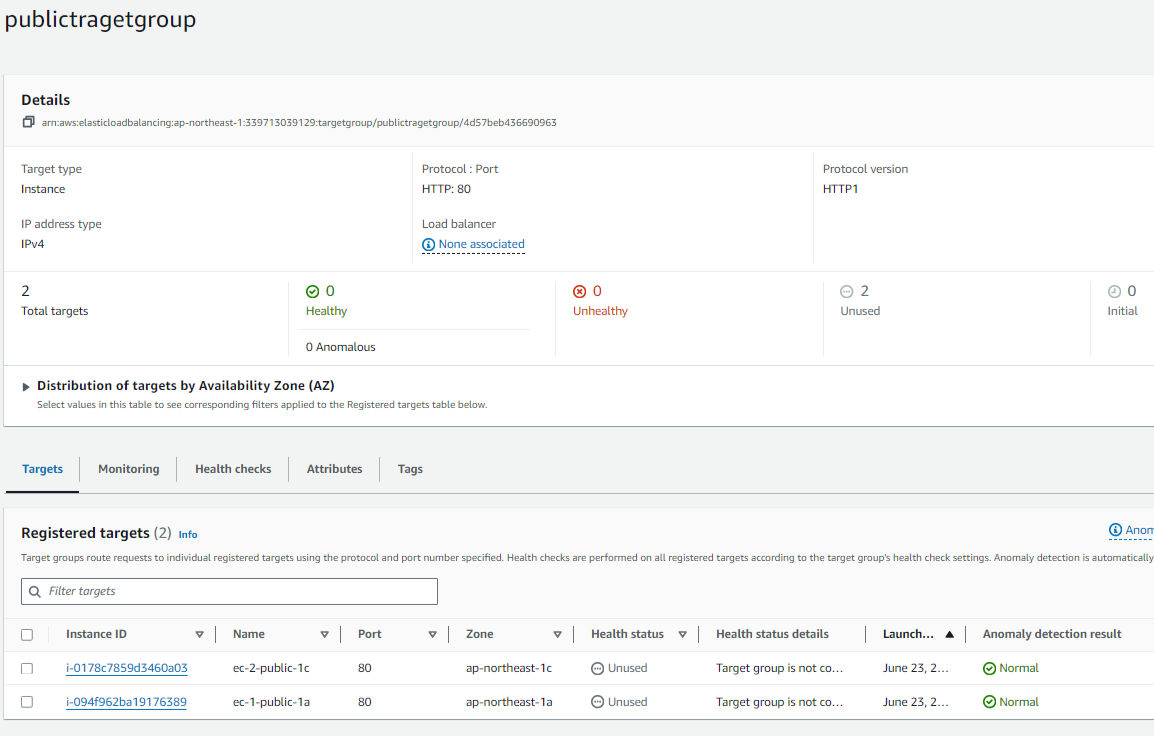
* Select the (target type instance) in the beginning



* Protocol port should be – HTTP (80)
* click on next
* select the two public instance and press include as pending below



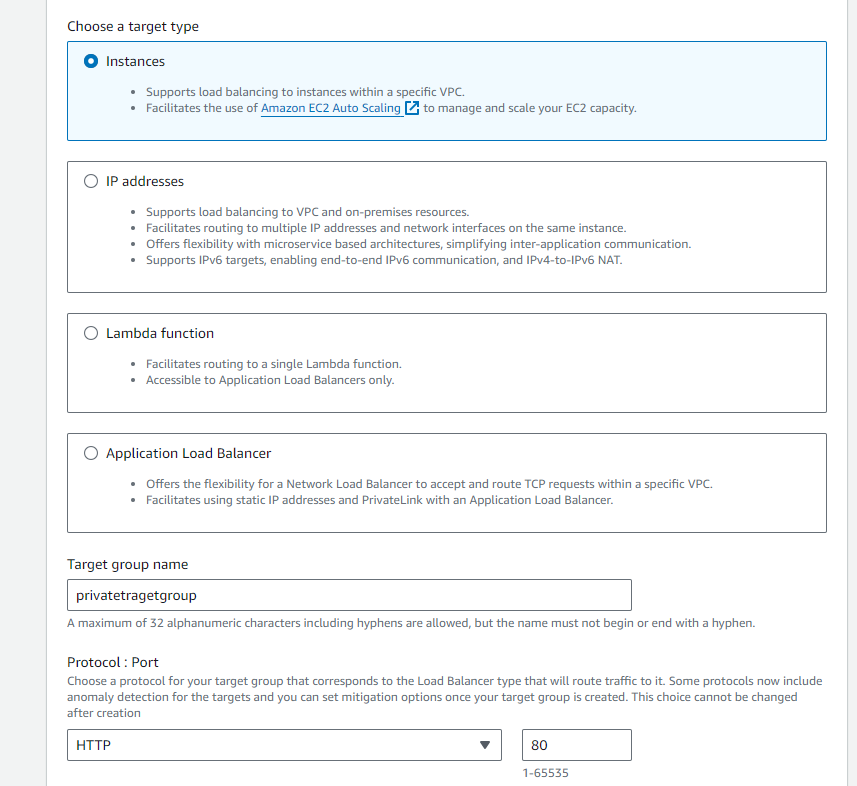
* scroll down press on create target group



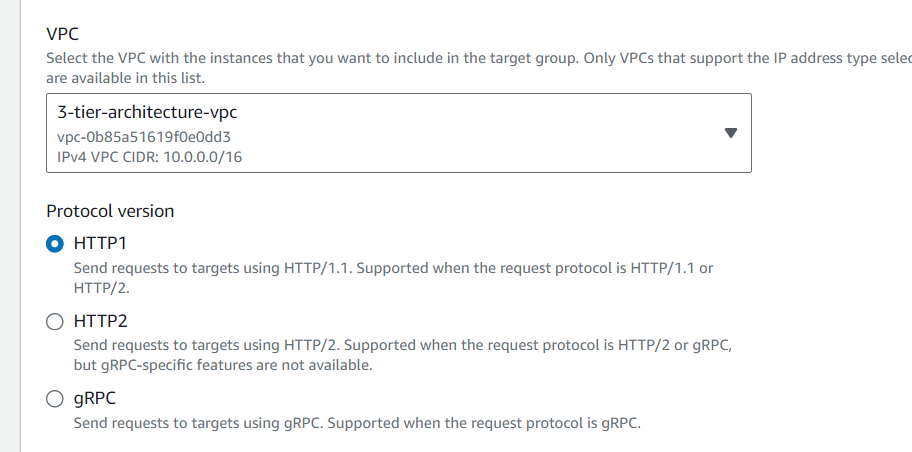
* Second create target group for 2 private instances

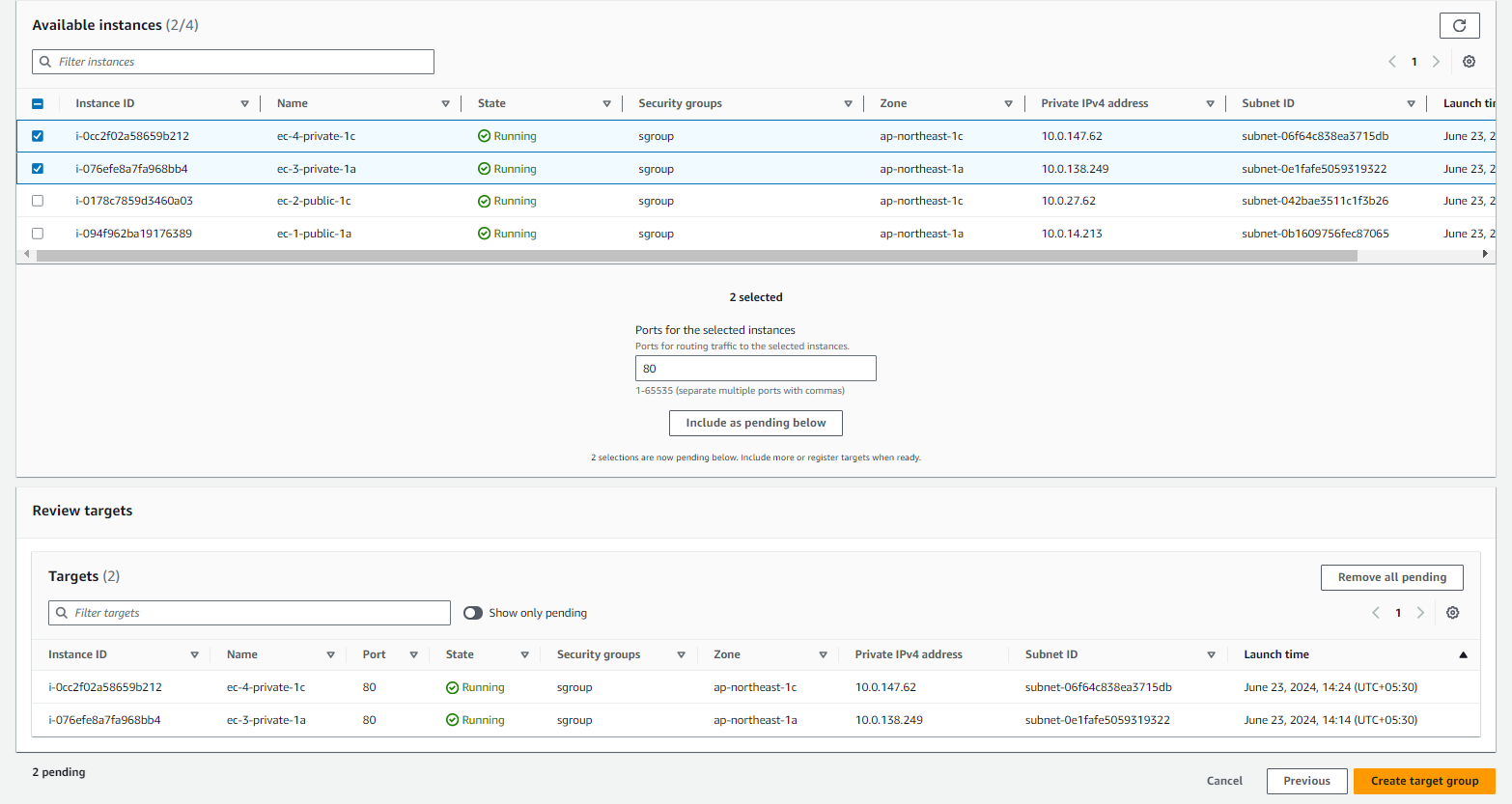
(private target group)

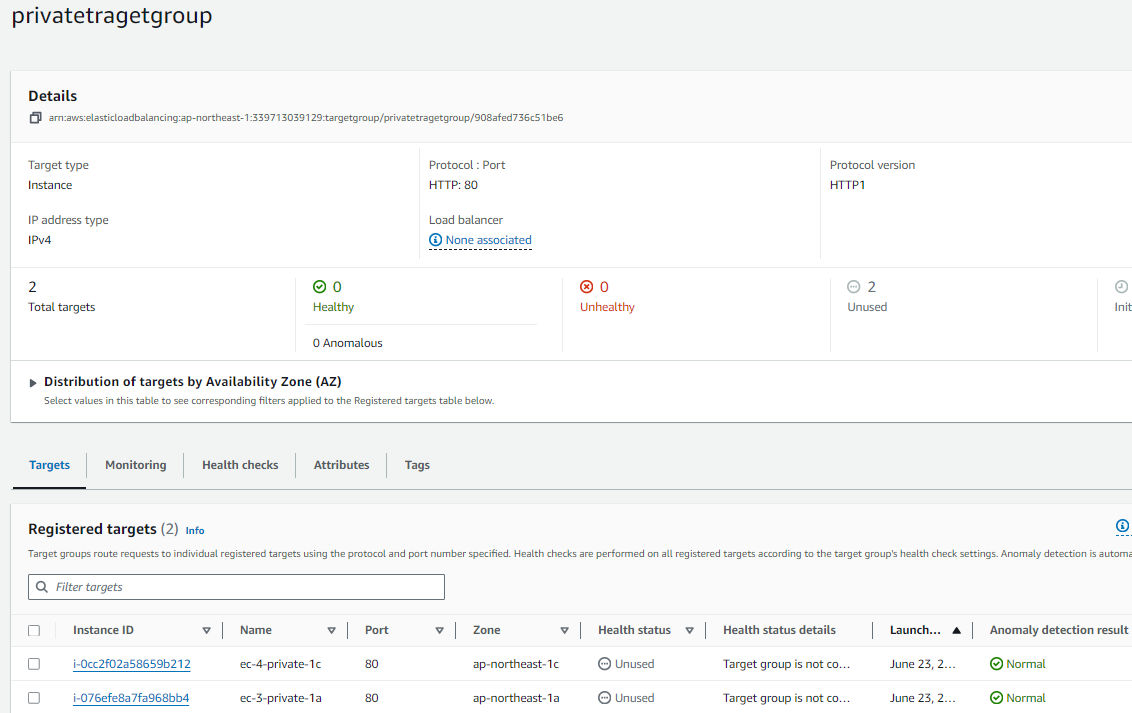
* Create the private target group



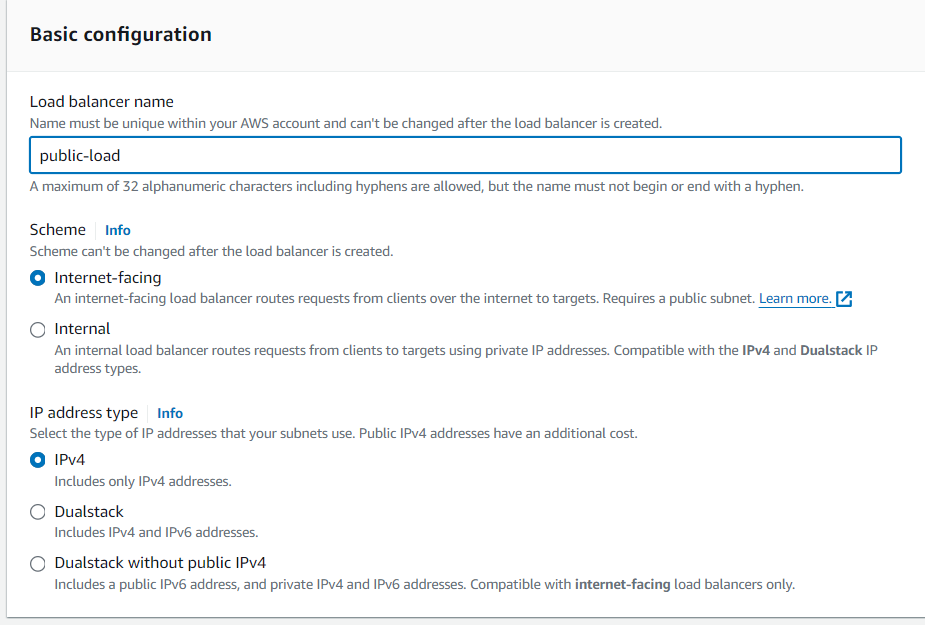
* Select the VPC and click on next



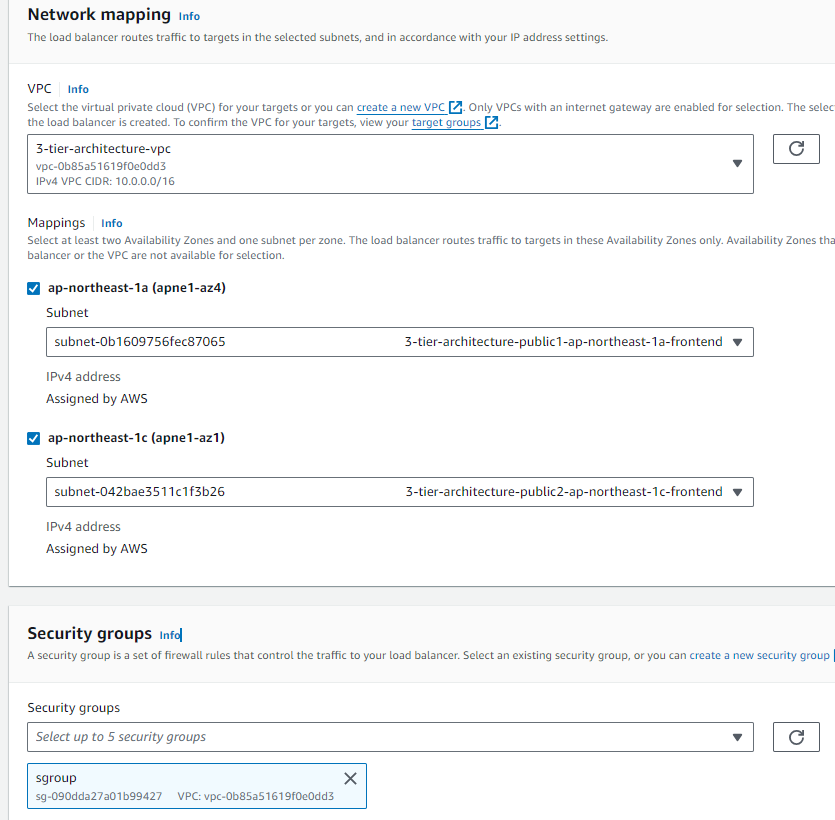
* As I said the protocol port should be -HTTP (80)
* Click on next
* Select the two private instance & press include pending below
* Next click on create target group



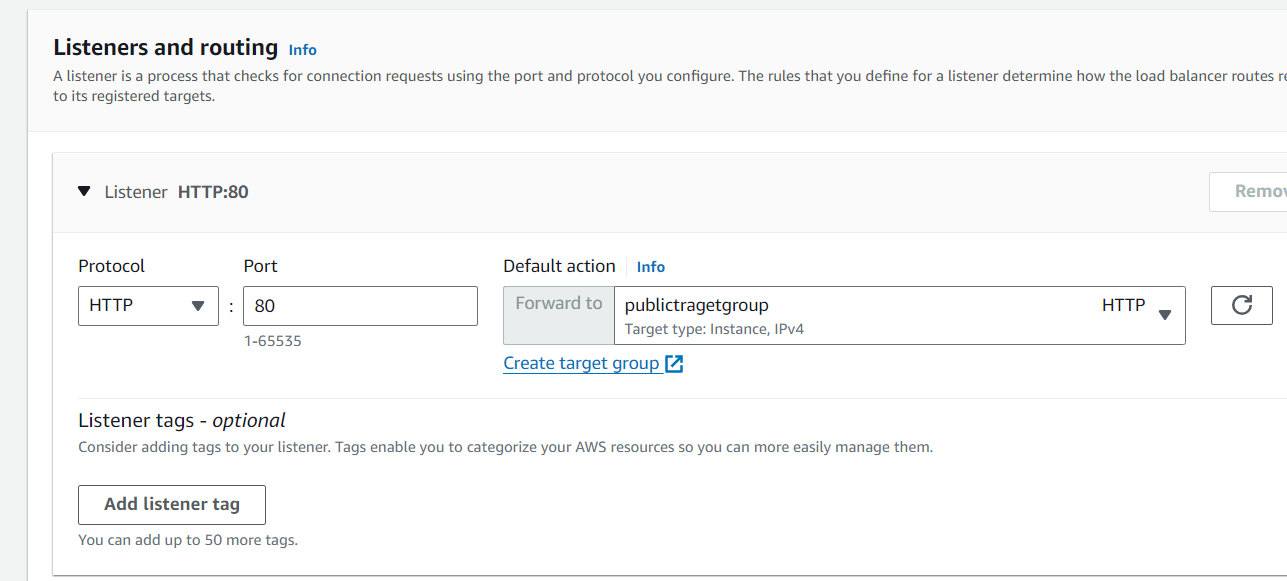
* Next go to load balancer create two load balancers for public & private target group
* Click on load balancer and select (Application load balancer)
* Give the first APL as public-load
* Select scheme- internal facing



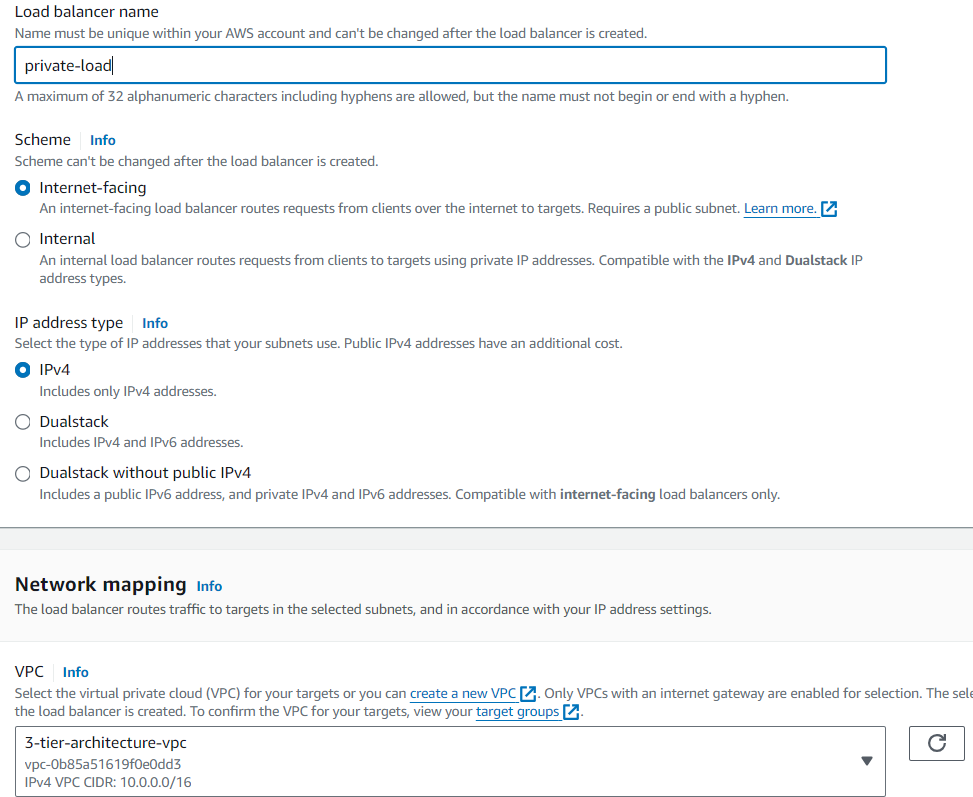
* In mapping Select the VPC (3-Architecter)
* Select the public-subnets from the 1a & 1c
* Select the security group (sgroup) which is created in the beginning



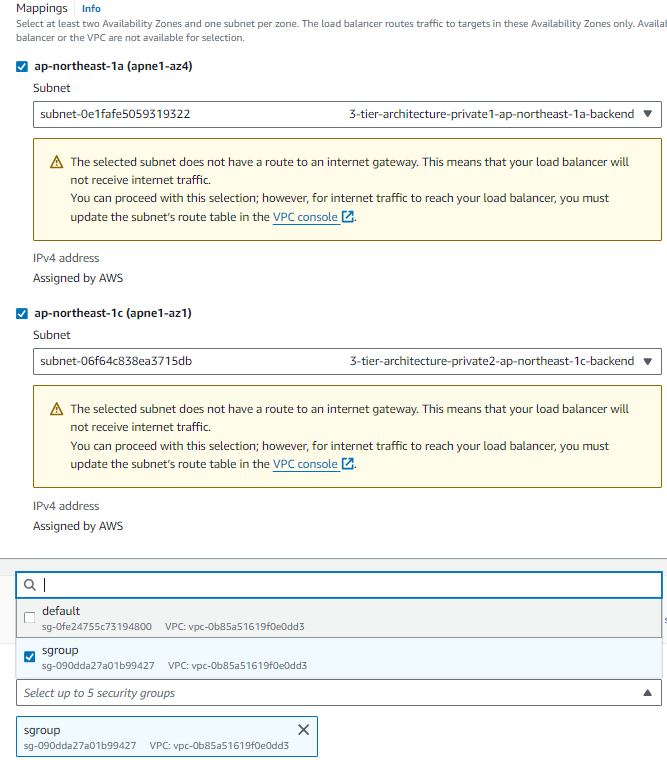
* Select the public target group in protocol port (HTTP)



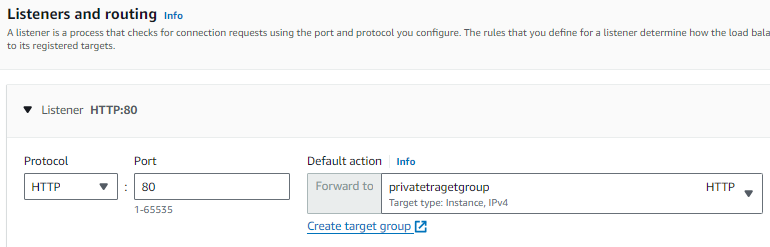
* Next scroll down & click on create a load balancer (public -load)
* Next create the private load balancer
* Select the Application load balancer
* Select the VPC(3-tier-architecture)



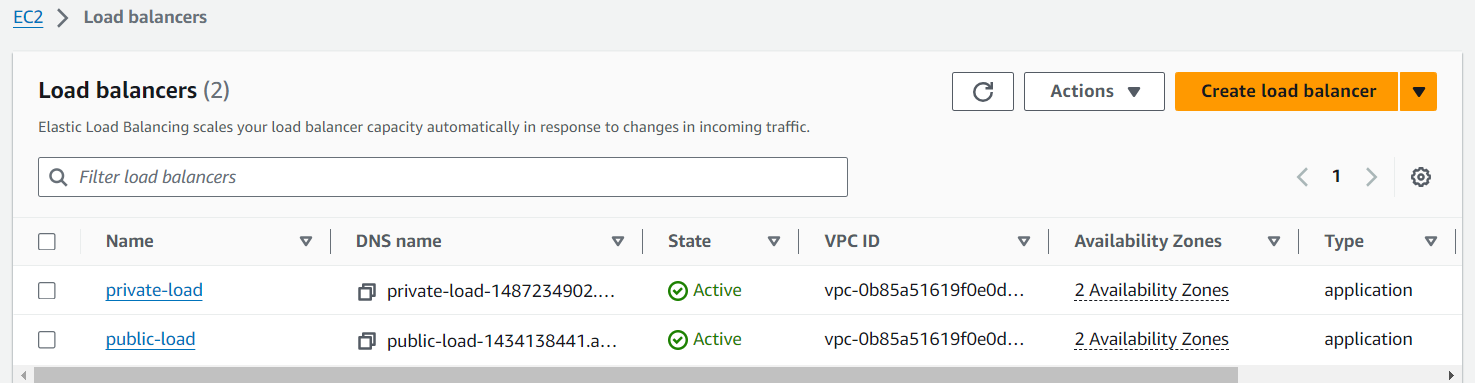
* Next go to mapping, select private-subnets from 1a-1c zones & give the security group as (sgroup) which is created in the beginning



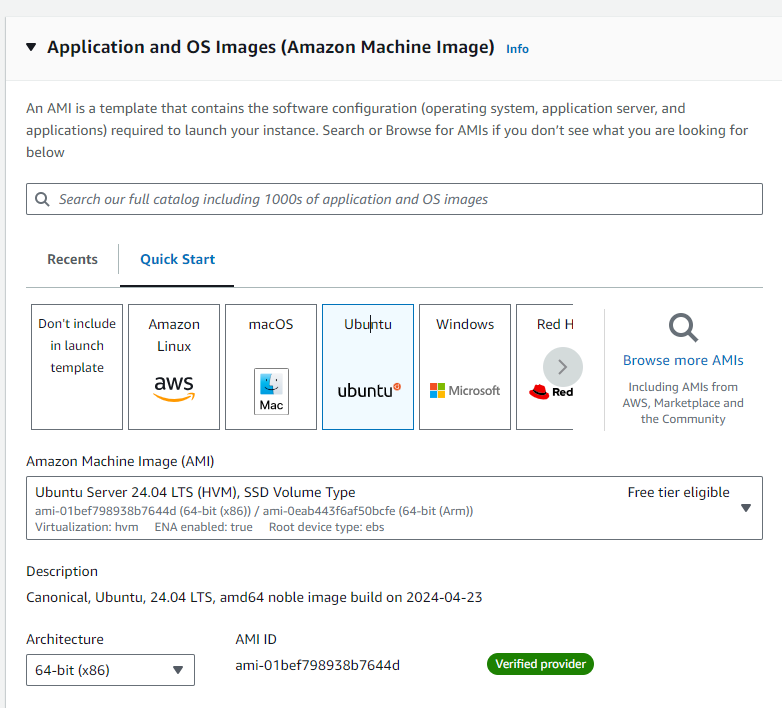
* Select the private-target group

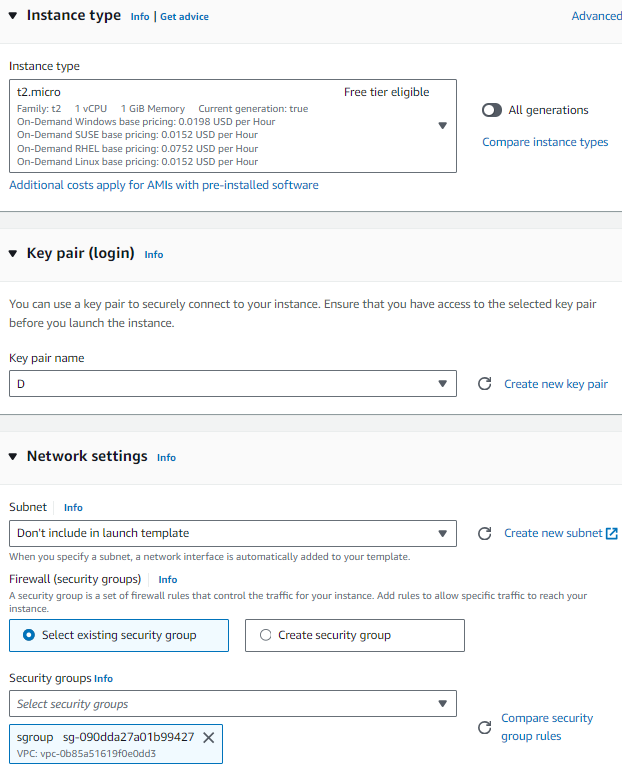


* Now click on create load balancer
* Two load balancers are created



* Next move to create lunch template only one template is enough for 2-Auto-scaling-groups
* Creating lunch template
* Template name – my template
* Choose Application and OS Images (Amazon Machine Image) click on UBUNTU

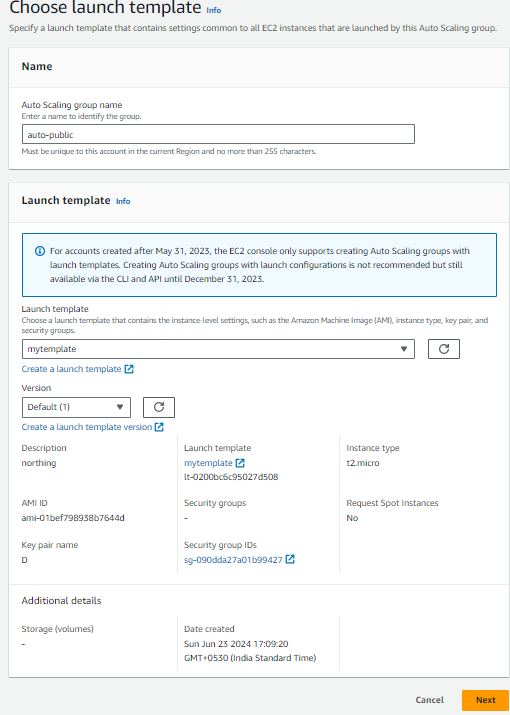




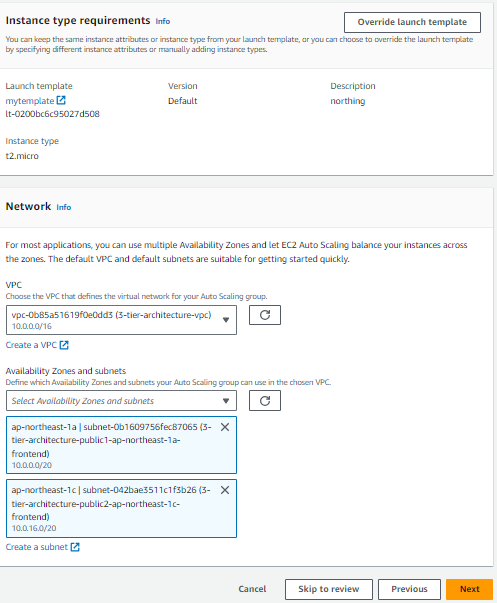
* Give the instance type – t2 micro
* Select the key pair
* Select the security group (sgroup)
* Click on create lunch template



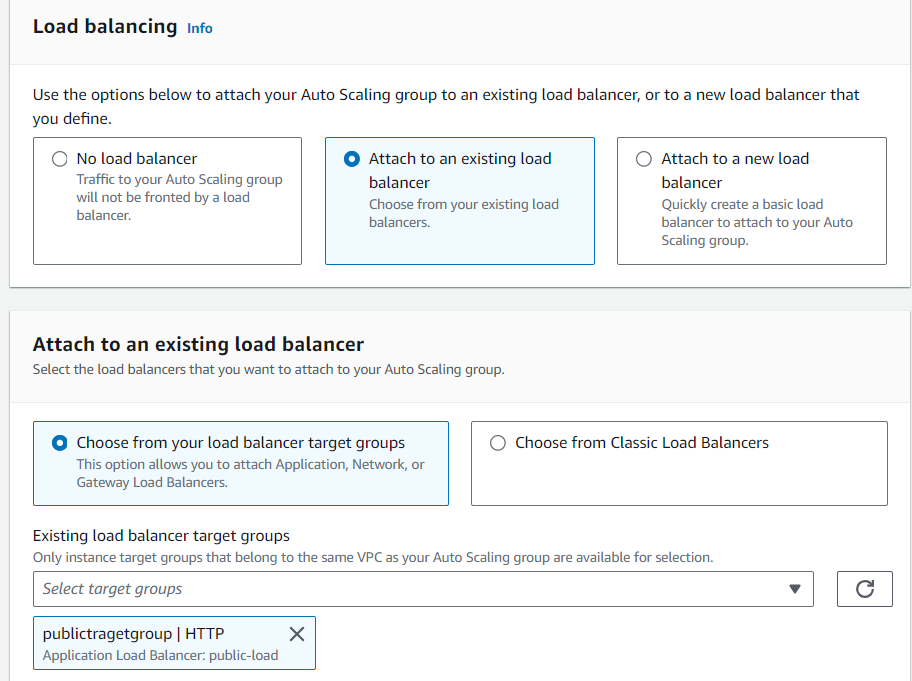
* Now create 2-Auto-scaling-groups with single template
* Auto-scaling for public & select the launch template



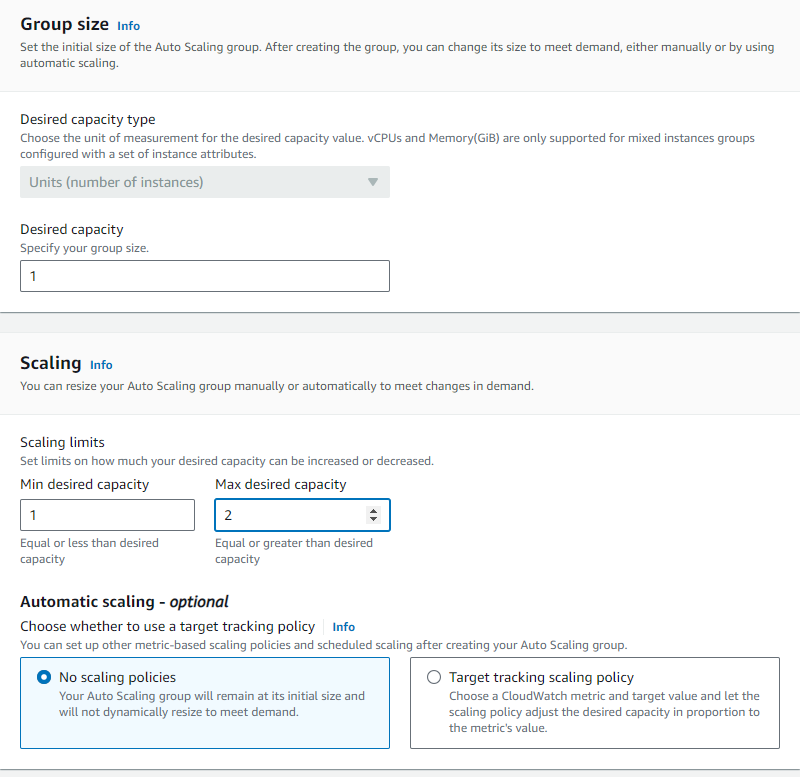
* Click on next
* Networks
* Select the VPC & Availability zones and subnets
* Off 1a & 1c public subnets



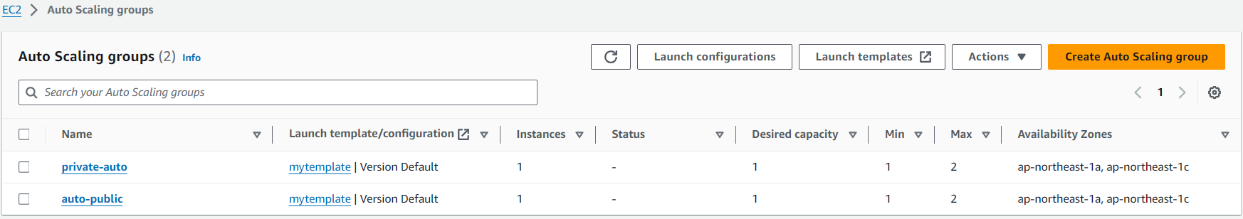
* Click on next
* Attach the existing load balancer & select the public target group



* Scroll down & press next



* Select the Desired capacity-1
* Give the Min desired capacity-1 & Max desired capacity-2
* No scaling policies in Automatic scaling
* Scroll down & click on next & next skip add notifications & next skip add tags which is optional
* Click on next & press create auto-scaling
* Next create private-auto scaling
* Click on the lunch template which is previously created
* Attach an existing load balancer and select the private-target-group
* Click on next skip the add notifications optional & skip the tags and click on create auto-scaling (private-auto)

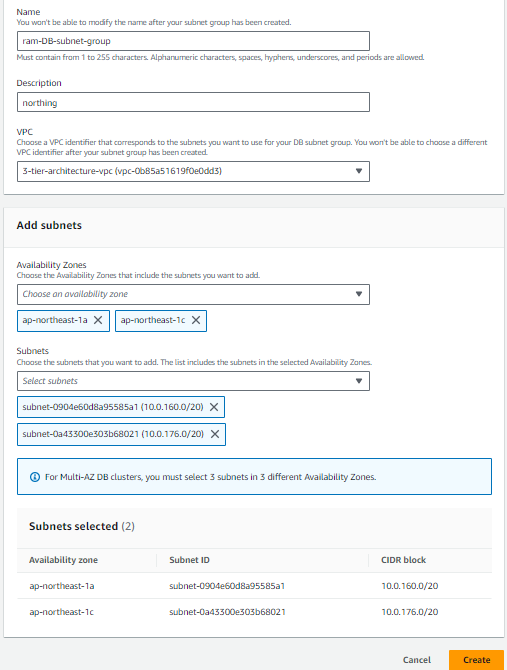


* Two-auto scaling groups are created
* And check either they are reflected or not in the

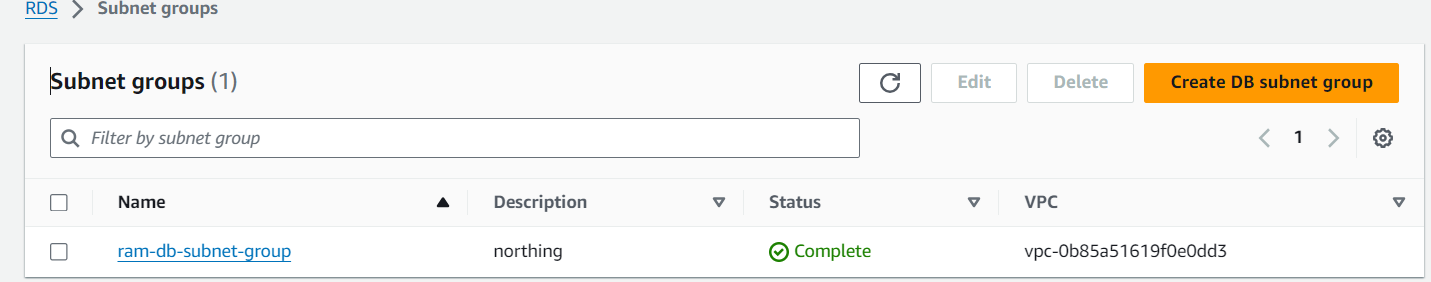
ec2-instance

# CREAT RDS

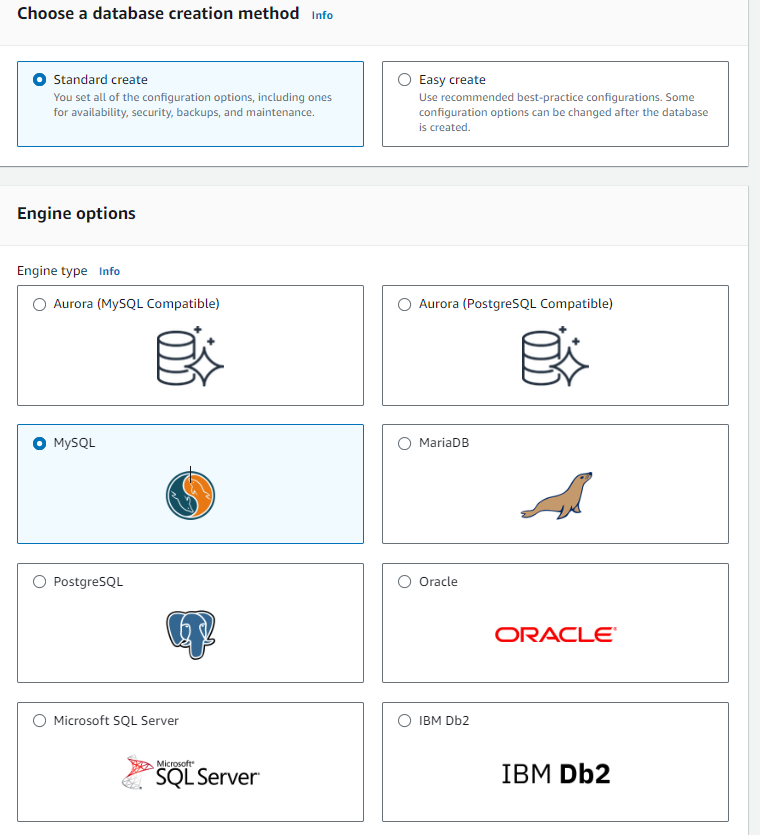
* Next create One Database in two availability zones as per the diagram
* Go to RDS (Managed Relational Data-Base Service)
* First click on subnet group in data-base & click on DB subnet group
* Give subnet-group details
* And give VPC (3-tier-architecture)
* Add Availability zone (1a & 1c)
* Give the subnet to the Data-Base
* Ip-(10.0.160.0/20)-subnet to data-base-1a
* Ip-(10.0.176.0/20)-subnet to data-base-1c
* Scroll down & click create



* The subnet group is created

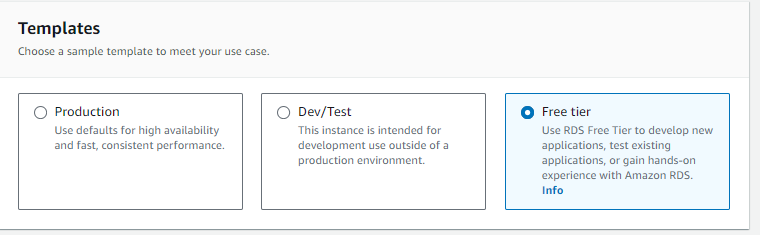


* Go to Data-base and create RDS Data-base
* Choose data-base creation method- (standard creation)
* And go to engine type & select the engine type (MYSQL) which I prefer to use frequently

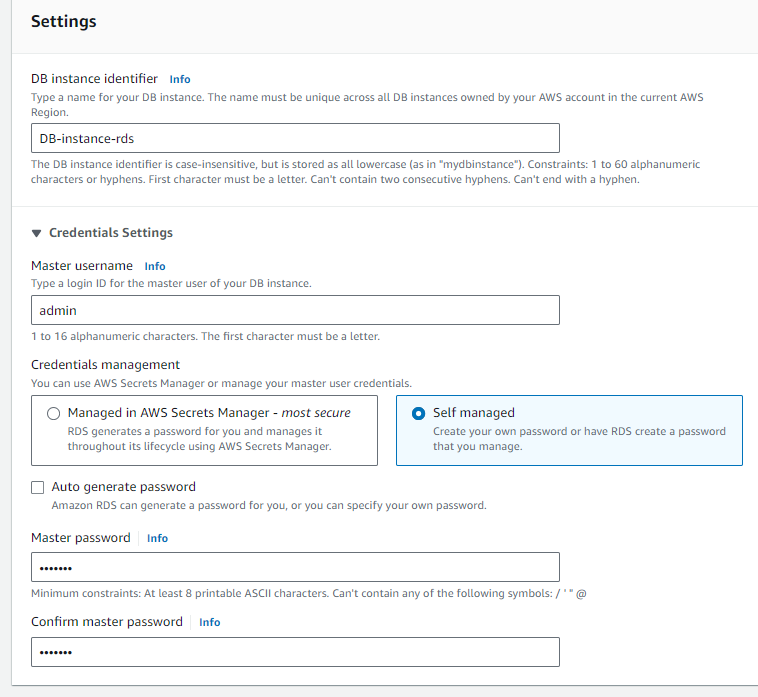


* Choose the sample templates to meet your use case

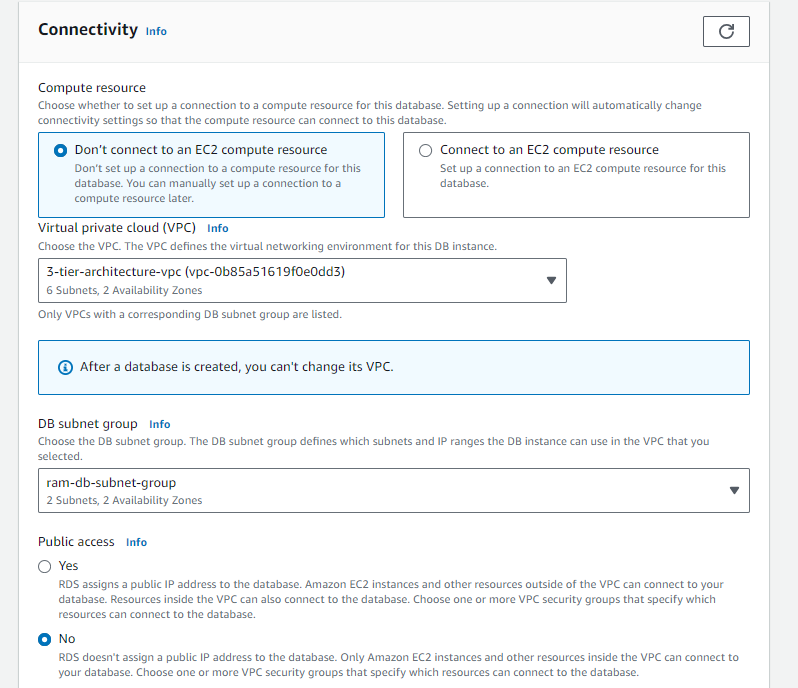
(Free tier) - use RDS free tier to develop new applications



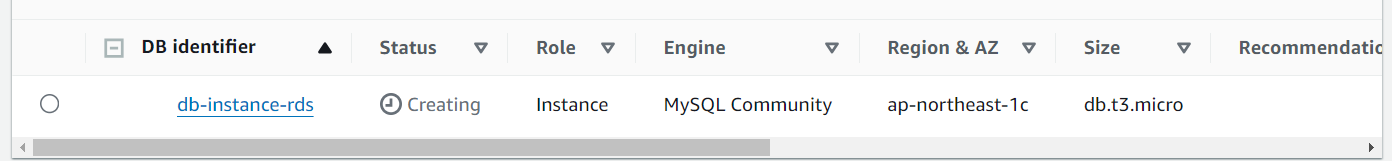
* Scroll to setting
* Give the DB-instance-identifier – (name)
* Next go to credentials settings & give the Master username (admin) it is used by default
* Credentials management🡪 select (self-managed)
* Click Master Password & confirm master password



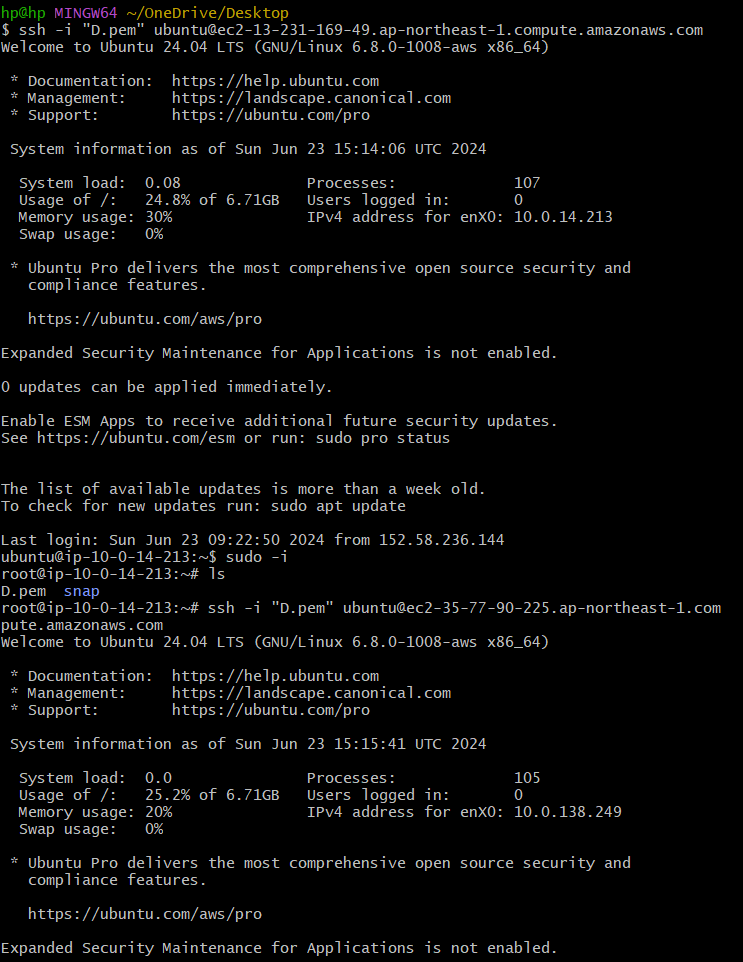
* Connectivity 🡪 compute resource🡪click on Don’t connect to an ec2 compute resource



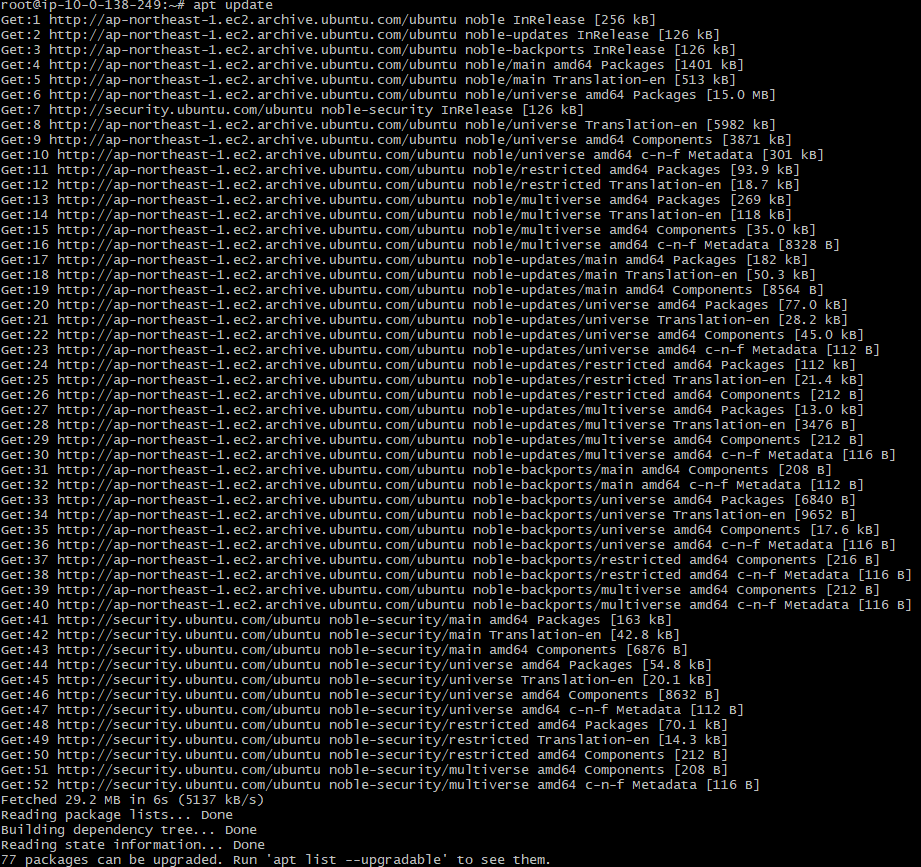
* Select the Virtual private cloud (VPC)
* Select the DB subnet group (ram-DB-subnet-group)
* Press no for public access
* Select the VPC security (sgroup)
* Scroll down and press create database



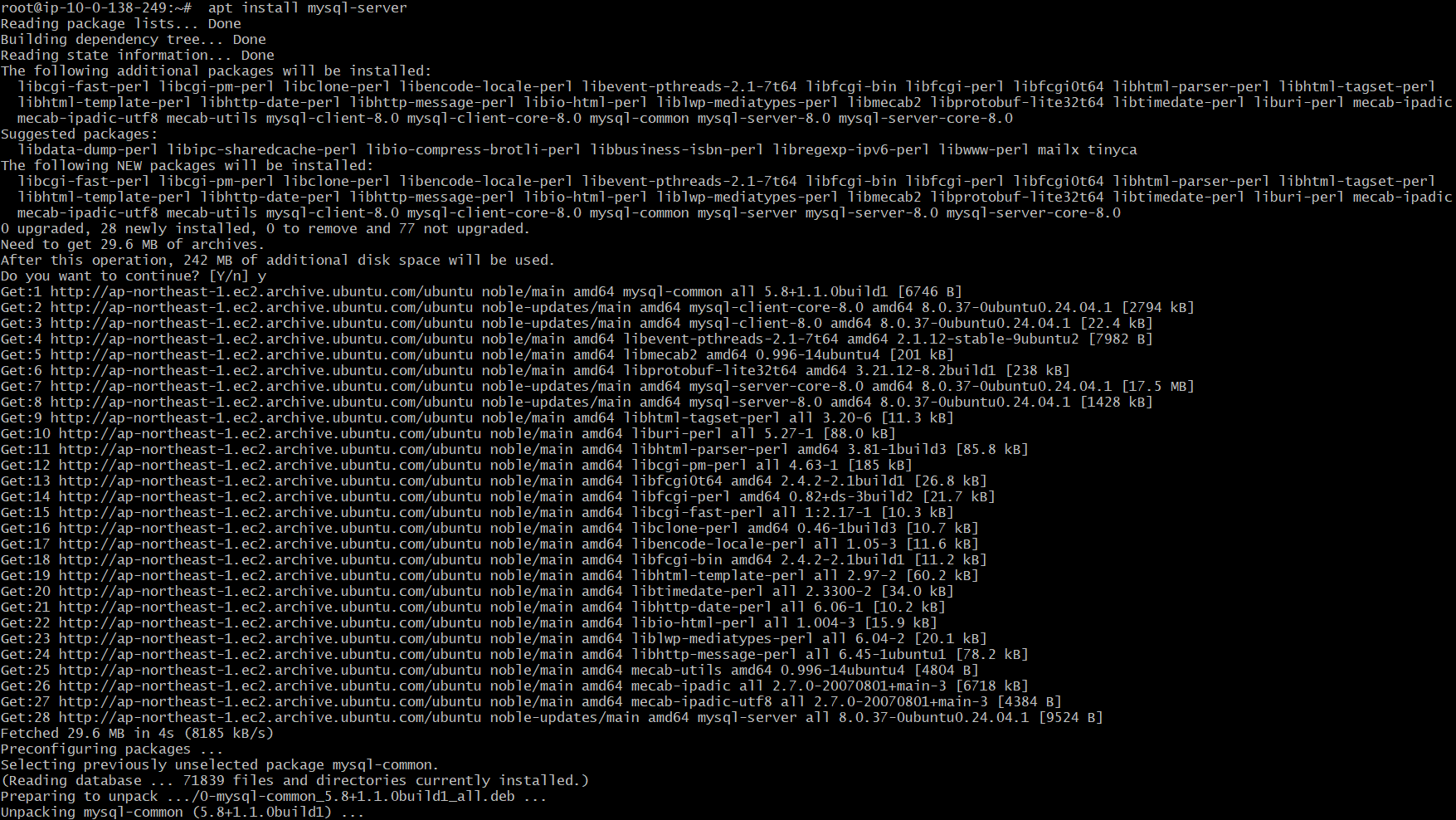
* Now connect to ec1-public to ec3-private



* Connect to ec3-private and press sudo – i
* Then you will be in root user install MYSQL
* Press (apt update) & it will update



* Press (apt install MYSQL-server) then MYSQL will be installed



* To restart MYSQL press (systemctl start mysql.service)
* Then copy the End-point from RDS-database
* Press(db-instance-rds.crc0ykue05ub.ap-northeast-1.rds.amazonaws.com)
* Press entre then you should press password to connect to MySQL

