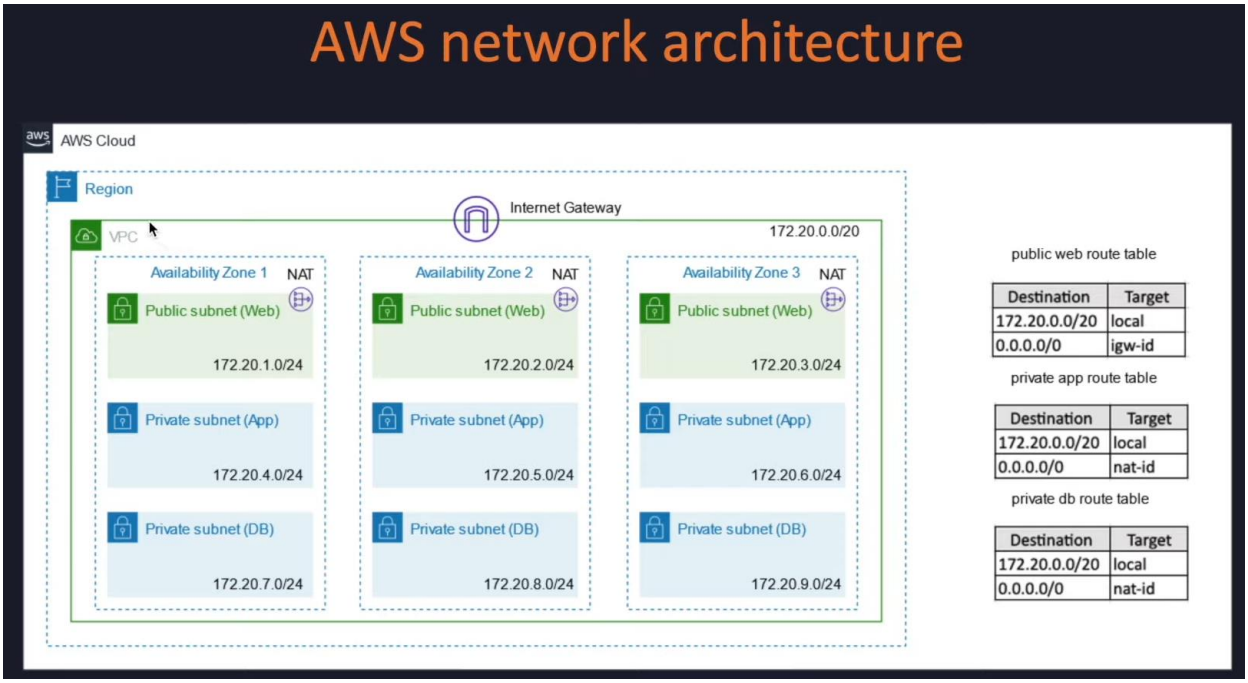
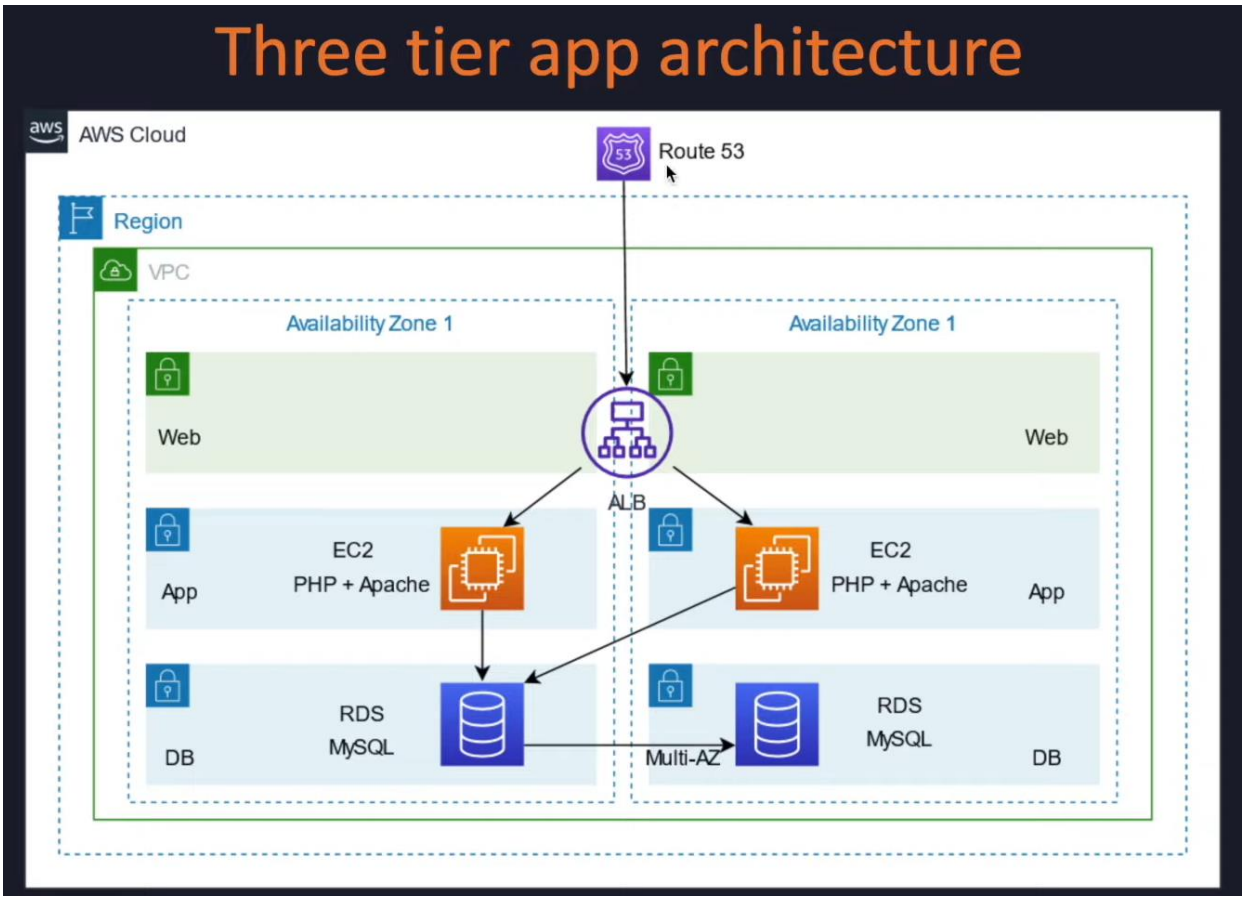
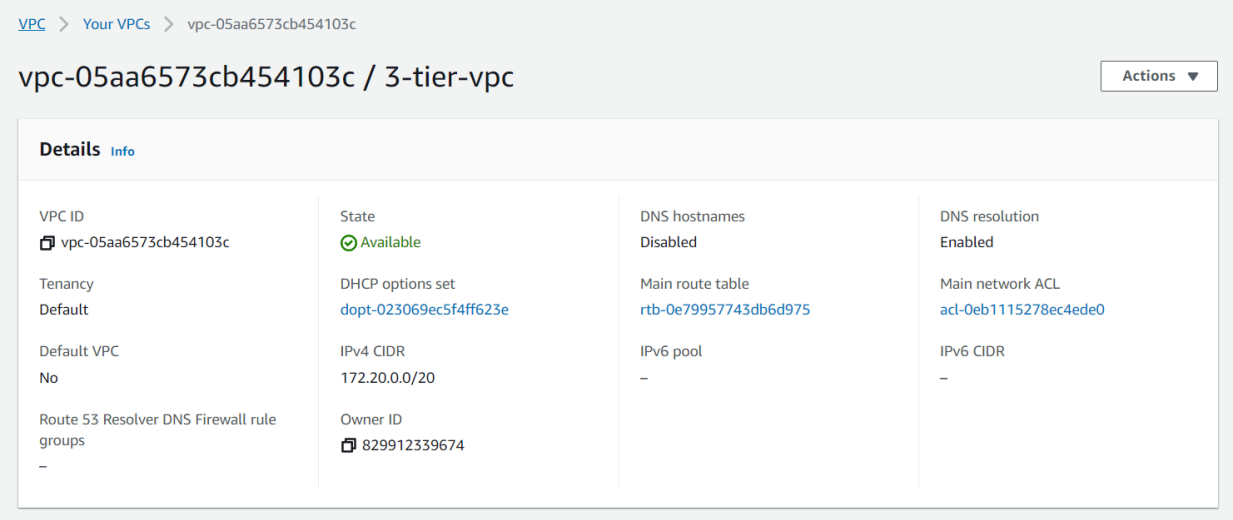


Services used:- VPC, ALB, EC2, RDS, Route53, Amazon Certificate Mnagaer



Step 1:- Select Any Region as per your choice, I choosed Ohio

Step 2:- Create VPC in Ohio region with IPv4 CIDR 172.20.0.0/20



Step 3:- Create 9 subnets for 3 servers

Public-web-subnet-1 →us-east-2a→172.20.1.0/24

Public-web-subnet-2 →us-east-2b→172.20.2.0/24 For Jump Server instance-1

Public-web-subnet-3 →us-east-2c→172.20.3.0/24

……

Private-App-subnet-1 →us-east-2a→172.20.4.0/24

Private -App-subnet-2 →us-east-2b→172.20.5.0/24 For App Server instance-2

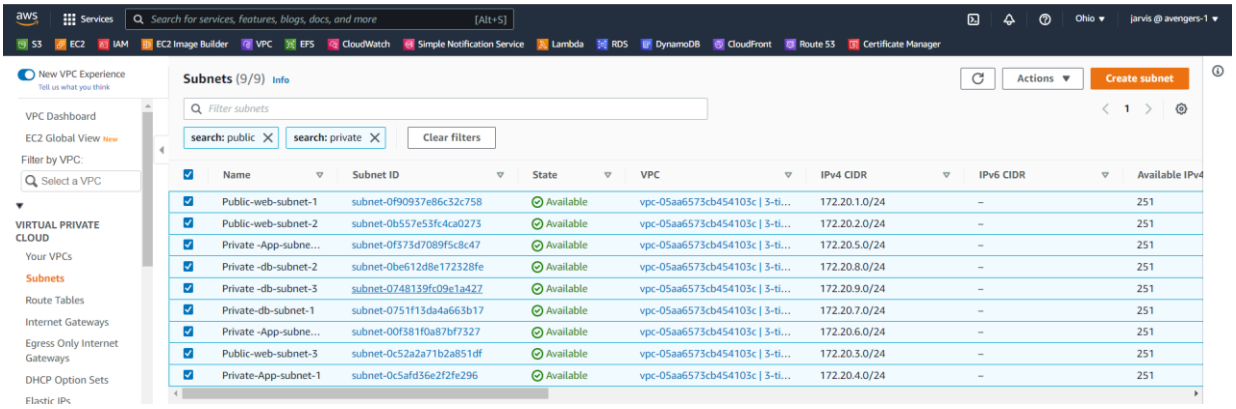
Private -App-subnet-3 →us-east-2c→172.20.6.0/24

……

Private-db-subnet-1 →us-east-2a→172.20.7.0/24

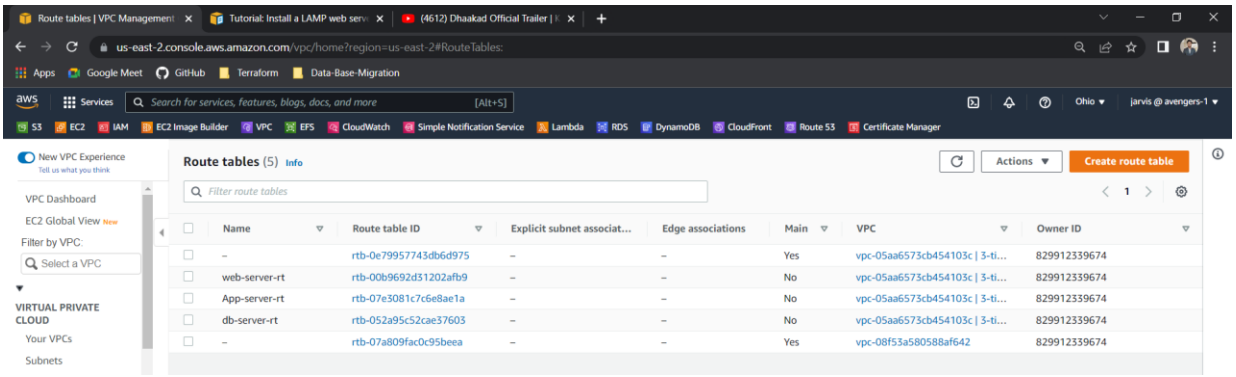
Private -db-subnet-2 →us-east-2b→172.20.8.0/24 For db Server instance-3

Private -db-subnet-3 →us-east-2c→172.20.9.0/24



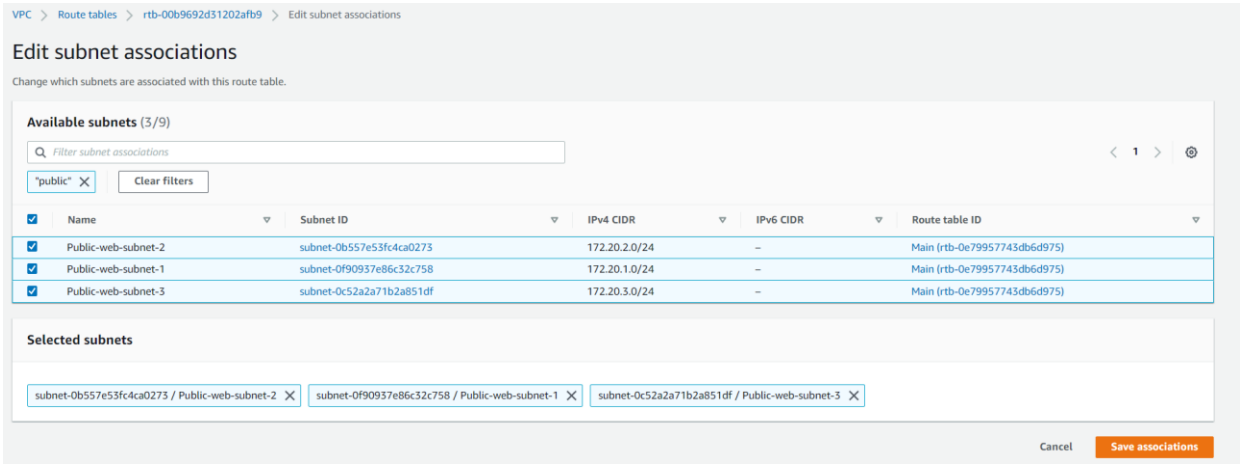
Step 4: - Create 3 route table for 3 instances don't use by default route table of created VPC

Web-server-rt, App-server-rt, db-server-rt

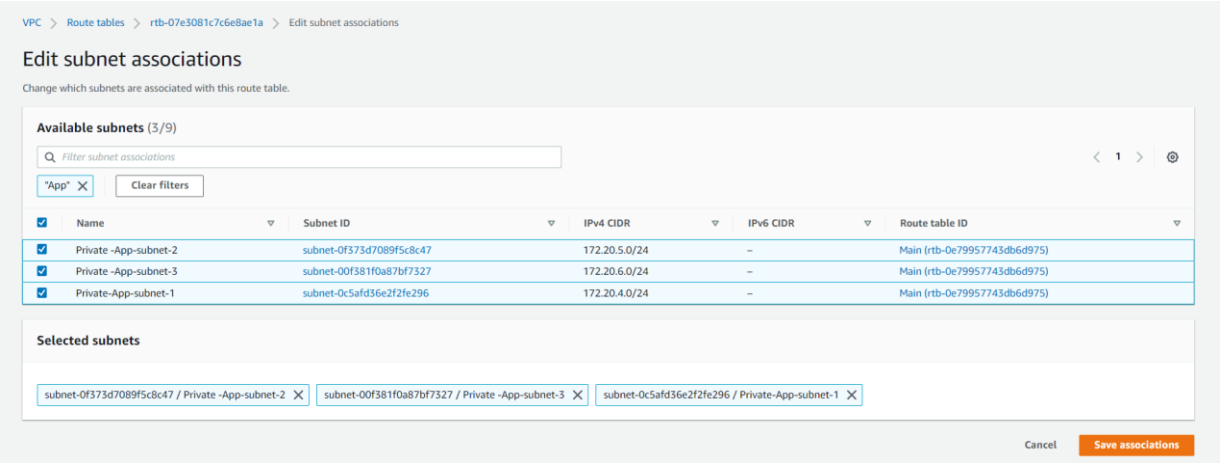


Step 5: - Associate all subnet to related specific route tables.

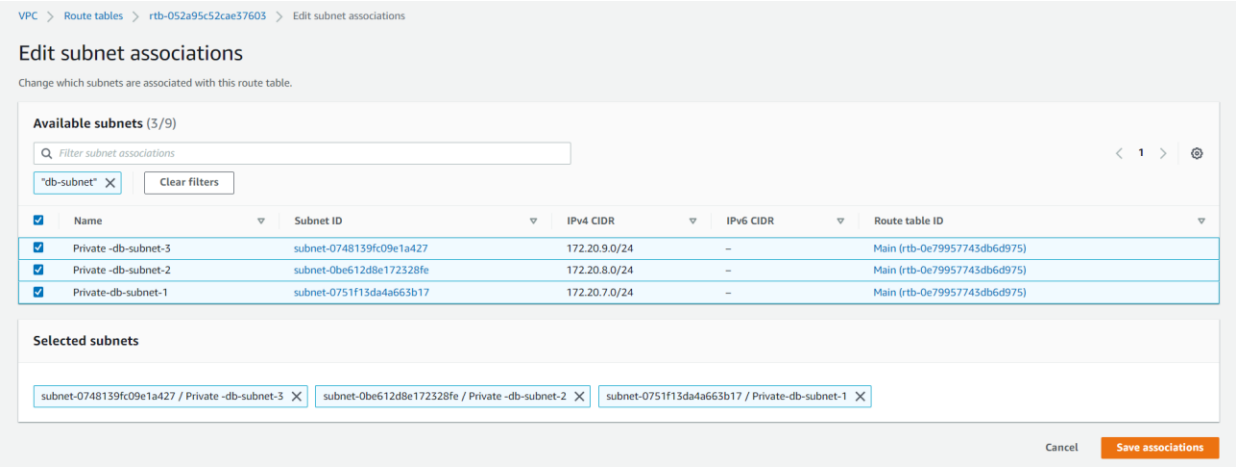
Web-server-rt



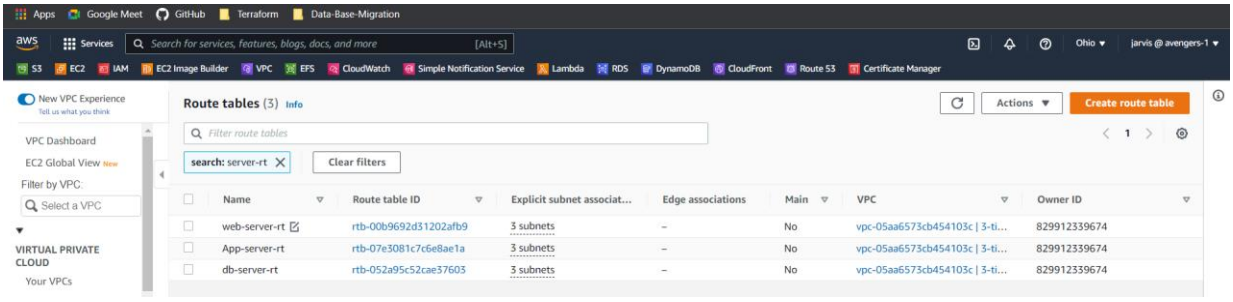
App-server-rt



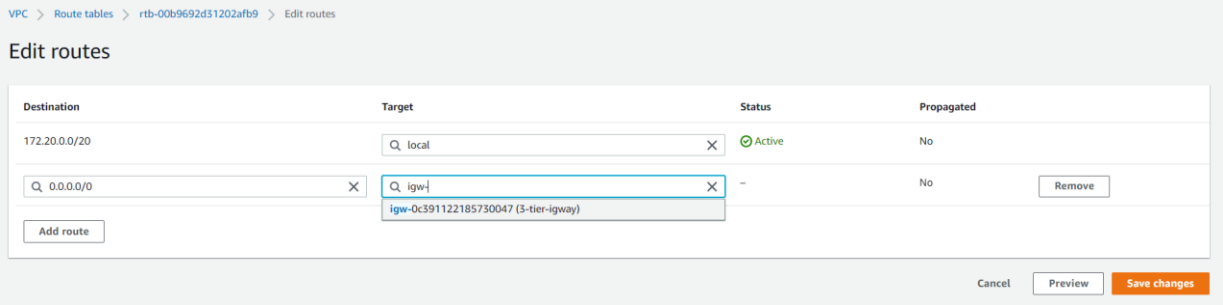
Db-server-rt



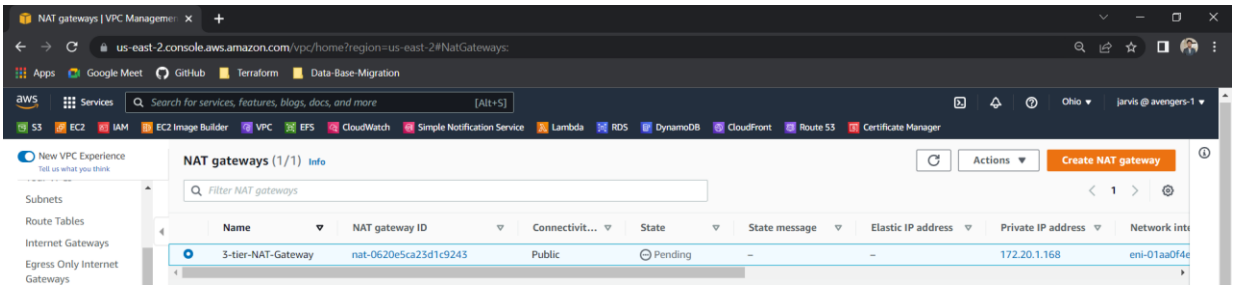
Succsefully all subnet are explicite associated with their route tables.



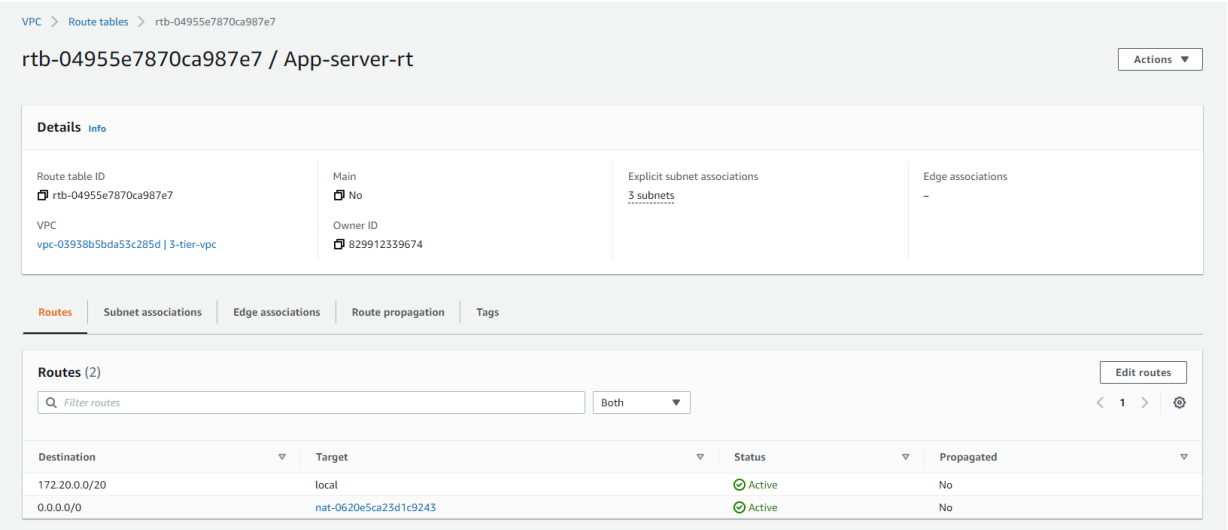
Step 6: - Create internet gateway→ Attach to VPC→ do entry in web-server-rt

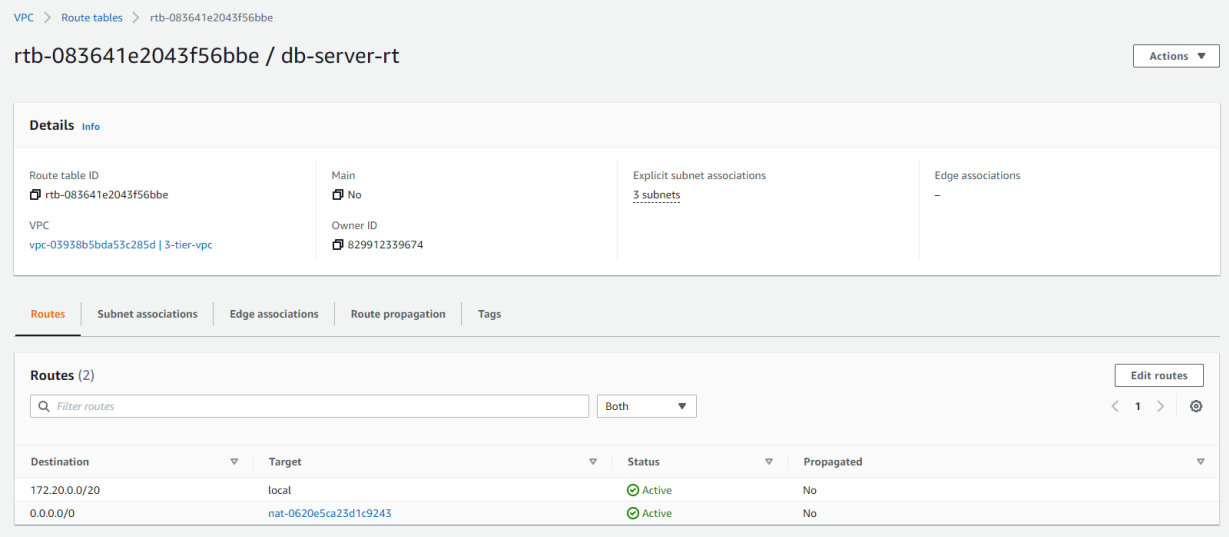


Step 7: - Create NAT Gateway→Attach to public-web-subnet-1→Allocate Elastic Ip



Attach NAT gateway to the App-server-rt & db-server-rt





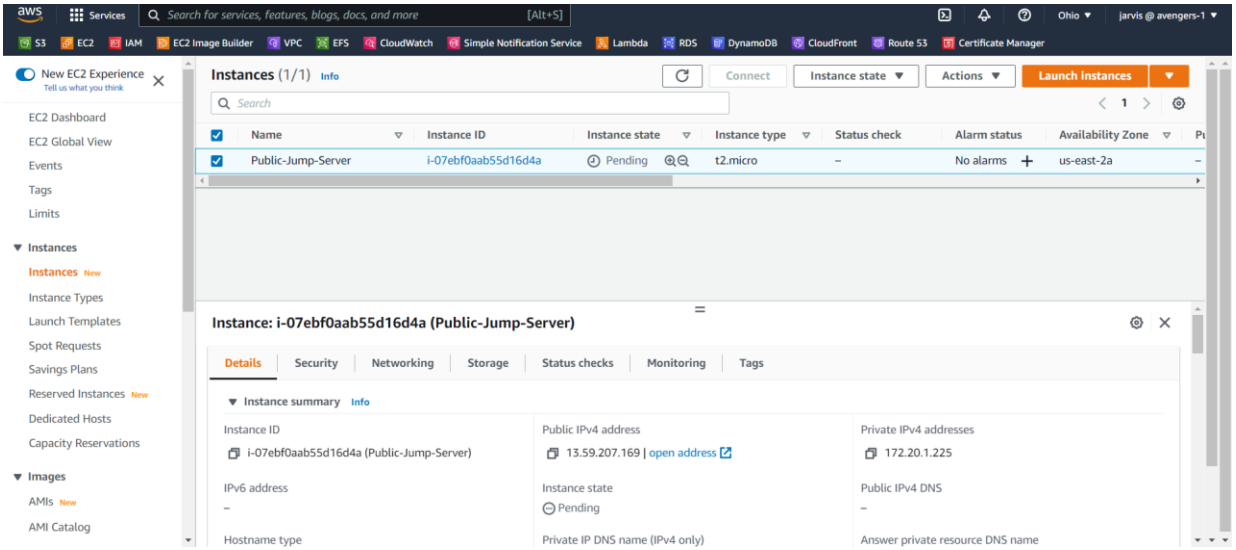
Step 7:- Create 3 EC2 instances with

1. Public-Jump-server (Public+Private)
2. Private-App-server (Private IP only)
3. Private-db-server (Private IP only)

App-server and db-server is only for routing traffic from both server, if any server failovers.

Public-Jump-Server:-

- Create new security group with ssh & http [jump-server-sg]
- Public-web-subnet-1
- Auto assign public IP enable

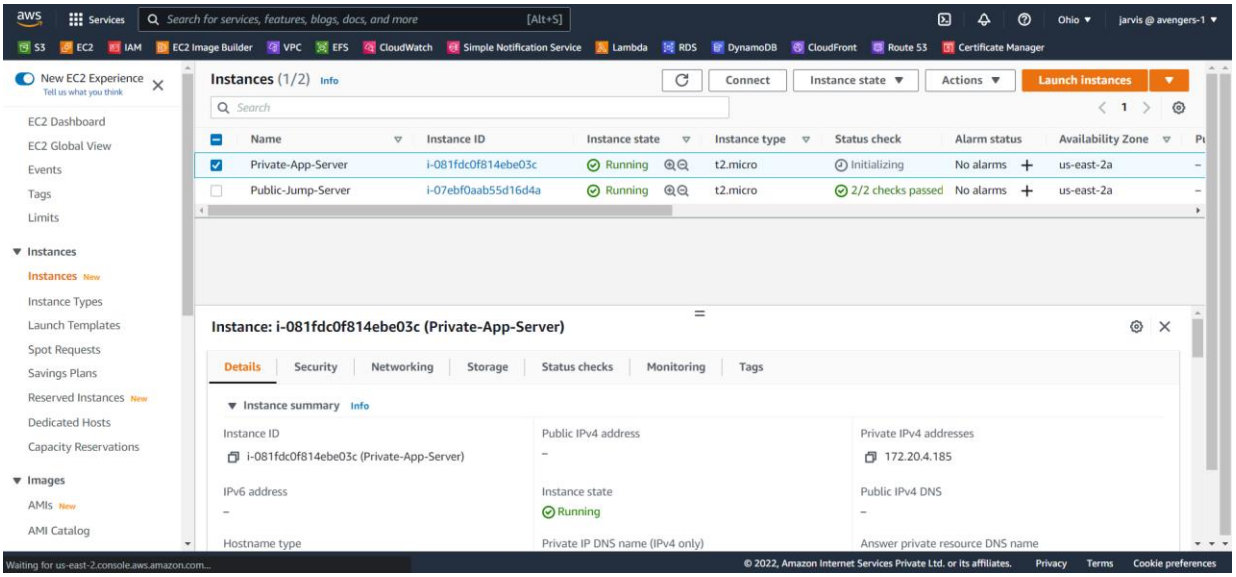


Private-App-Server:-

- Private-App-subnet-1
- Auto assign public IP disable
- Create Security group [App-server-sg]

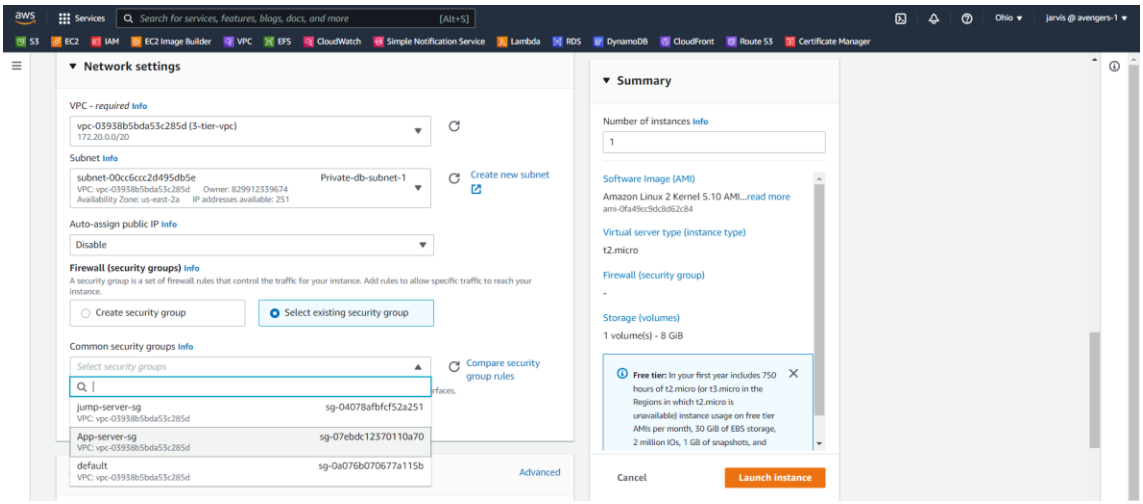
  SSH→custom→ jump-server-sg

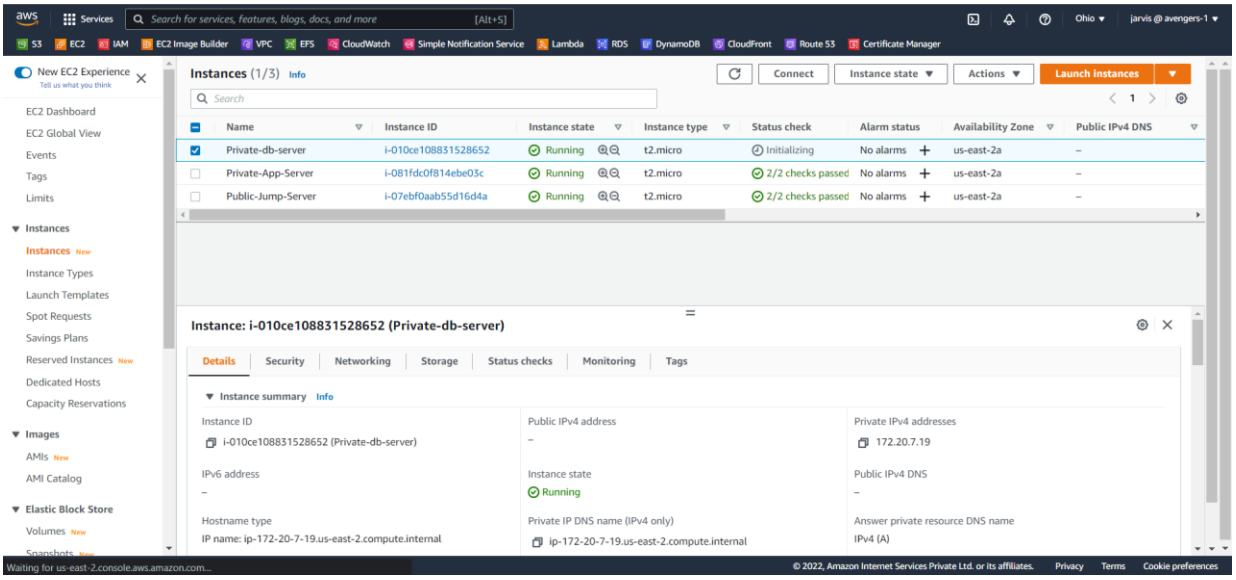
  Http→Anywhere



Private-db-Server:-

- Private-db-subnet-1
- Auto assign public IP disable
- Choose existing security group [App-server-sg] of Private-App-server

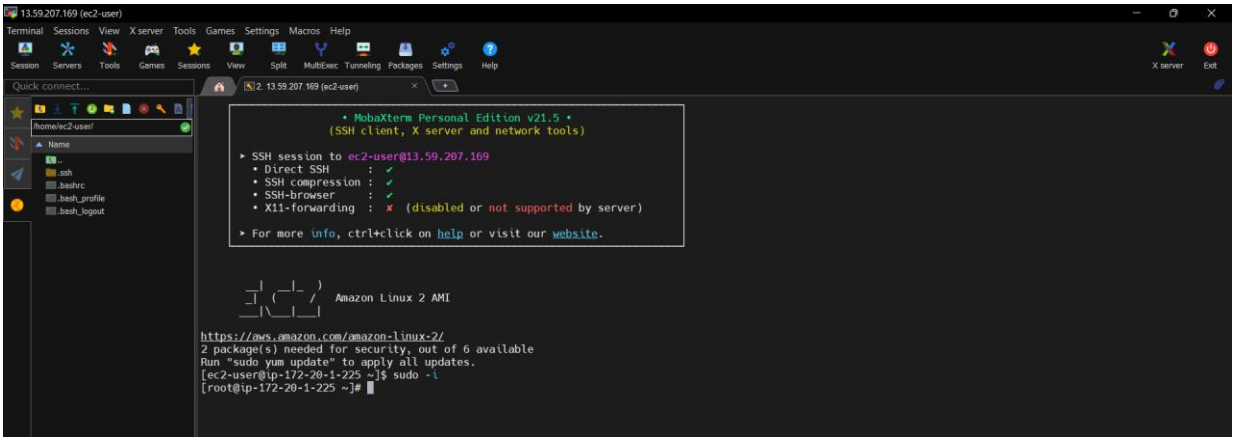




Step 8:- Connect public-jump-server and **get SSH into both private servers** (App-server & db-server)

- Install php and apache
- Install LAMP web server on the Amazon Linux AMI

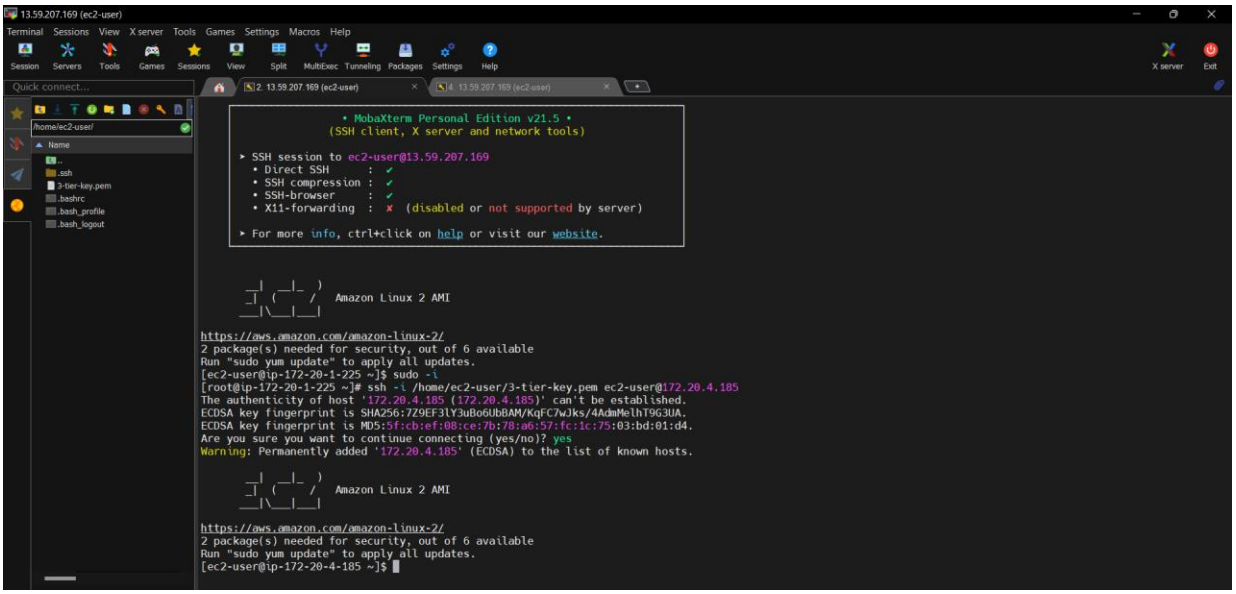
Public-Jump-server connect with moba succsesfully



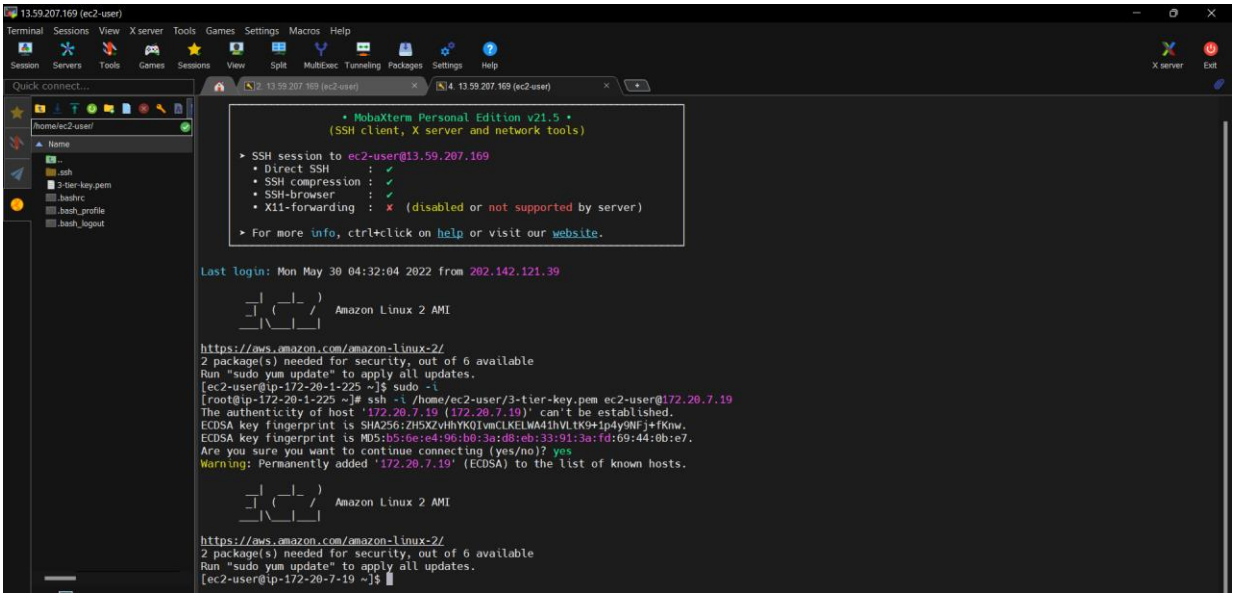
Now, we have to connect Private-App-Serve and Private-db-server for installing php and lamp server

- Opent 2 tabs of jump-server
- Upload 3-tier-key pair in moba both tabs
- In one tab of jump-server get ssh of Private-App-server and same another tab get of jump-server get ssh of Private-db-server

ssh -i /home/ec2-user/3-tier-key.pem ec2-user@<Private-ip of app-server>



ssh -i /home/ec2-user/3-tier-key.pem ec2-user@<Private-ip of db-server>



Step 9:- Install Lamp server into the both Private instance via getting SSH from moba

- Install Lamp server
- Set file permission
- Install phpMyAdmin

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/install-LAMP.html refer the Doc

Prepare the Lamp server

[ec2-user ~]$ sudo yum update -y

[ec2-user ~]$ sudo amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2

cat /etc/system-release

[ec2-user ~]$ sudo yum install -y httpd mariadb-server

[ec2-user ~]$ sudo systemctl start httpd

[ec2-user ~]$ sudo systemctl enable httpd

[ec2-user ~]$ sudo systemctl is-enabled httpd

[ec2-user ~]$ curl localhost

……

To set file permissions

[ec2-user ~]$ sudo usermod -a -G apache *ec2-user*

[ec2-user ~]$ exit

[ec2-user ~]$ sudo chown -R ec2-user:apache /var/www

[ec2-user ~]$ sudo chmod 2775 /var/www && find /var/www -type d -exec sudo chmod 2775 {} \;

[ec2-user ~]$ find /var/www -type f -exec sudo chmod 0664 {} \;

……

Install my phpMyAdmin

[ec2-user ~]$ sudo yum install php-mbstring php-xml -y

[ec2-user ~]$ sudo systemctl restart httpd

[ec2-user ~]$ sudo systemctl restart php-fpm

[ec2-user ~]$ cd /var/www/html

[ec2-user html]$ wget *https://www.phpmyadmin.net/downloads/phpMyAdmin-latest-all-languages.tar.gz*

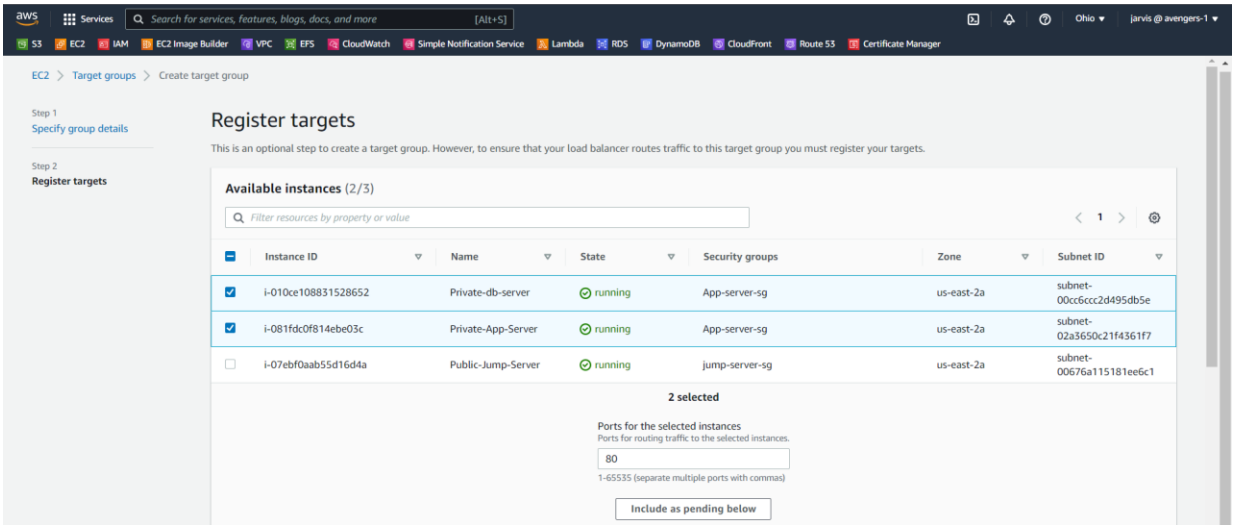
[ec2-user html]$ mkdir phpMyAdmin && tar -xvzf *phpMyAdmin-latest-all-languages.tar.gz* -C phpMyAdmin --strip-components 1

[ec2-user html]$ rm *phpMyAdmin-latest-all-languages.tar.gz*

[ec2-user ~]$ sudo systemctl start mariadb

Step 10: - Create Target group [TgforALB]

- Target type: - Instances
- Register target for both private instances



Step 11:- Create Application Load balancer

VPC:- 3- tier-vpc

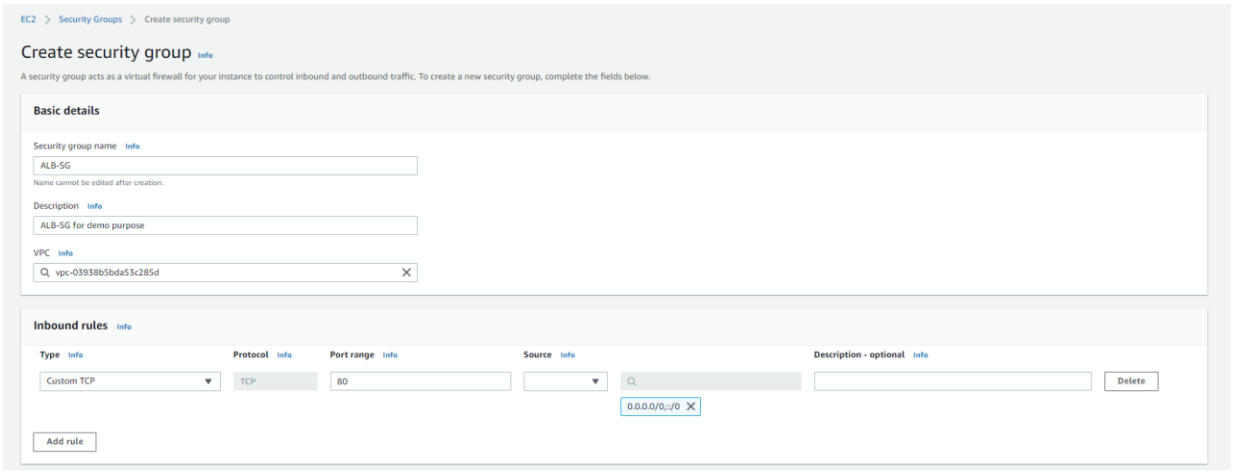
Mapping:- us-east-2a (public-web-subnet-1)

us-east-2b (public-web-subnet-2)

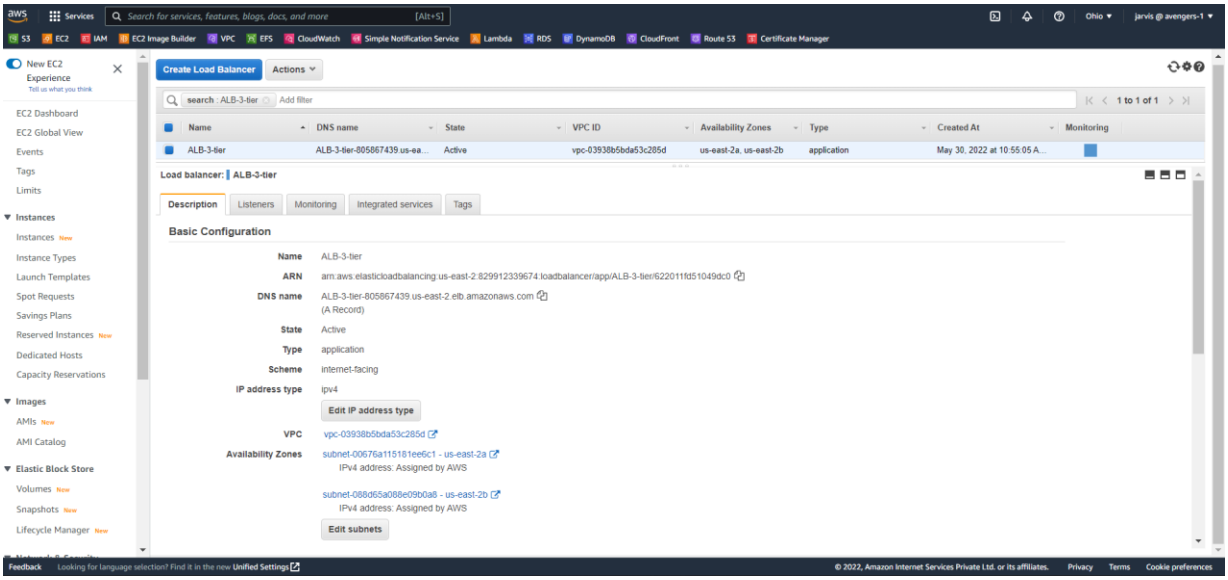
Attach saperately created security group for load balancer

Attach created TgforALB

Create New Security group for Application Load Balancer



Application Load balancer created succsesfully

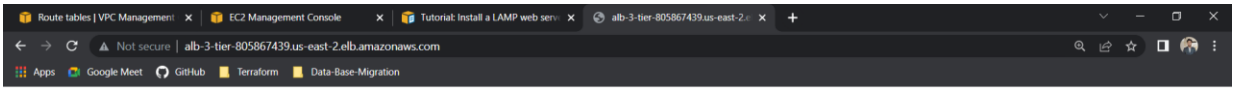


Step 12:- check wetaher it working or not ?

echo "Hello chetan" > /var/www/html/index.html → Private-App-server

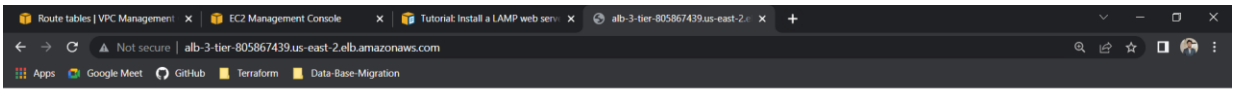
echo "Hello Jarvis" > /var/www/html/index.html → Private-db-server

Hit DNS and check



Hello Jarvis

After HARD Rrefreshing



Hello chetan

Both pages are shown succsefully

Step 13:- Create Subnet group for 3-tier rds databse [subnetgroup-3-tier]

VPC:- 3-tier-vpc

Subnets selected (2)

Availability zone	Subnet ID	CIDR block
us-east-2a	subnet-00cc6ccc2d495db5e	172.20.7.0/24
us-east-2b	subnet-0fe58e506bfde6cc9	172.20.8.0/24

Step 14:- Create RDS Database

Standard Create

Engine:- MySQL

Template:- Free tier

DB instance identifier:- jarvis-db

Creadentials:- admin/admin123456

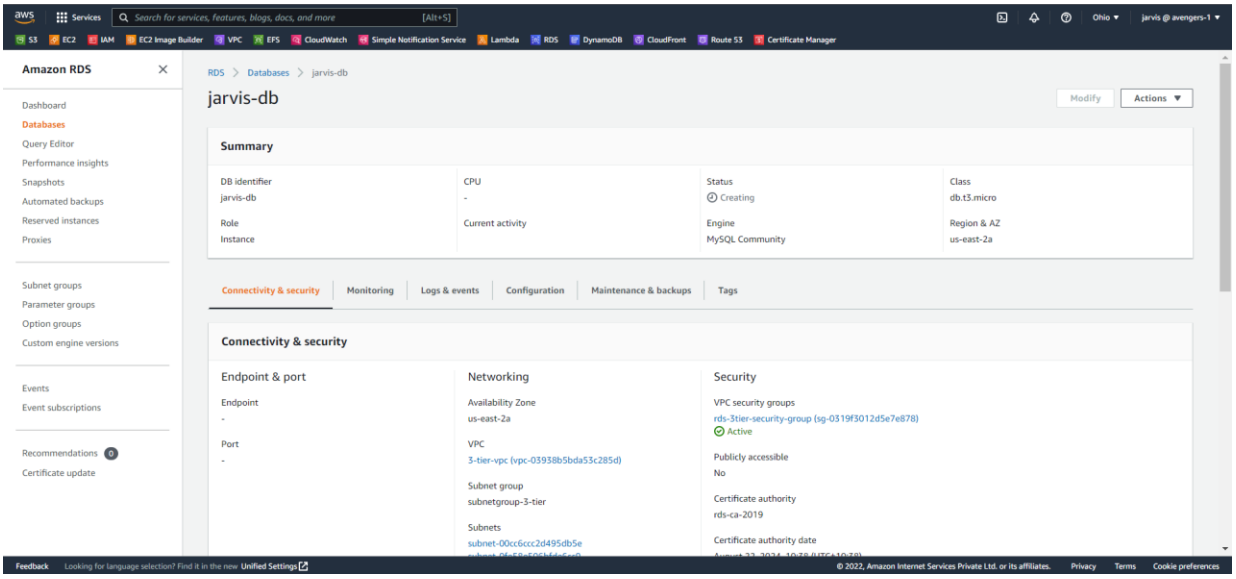
VPC:- 3-tier VPC

Subnet group:- automatically select created subnet group subnetgroup-3-tier

Public Access: - NO

Security group: - create New

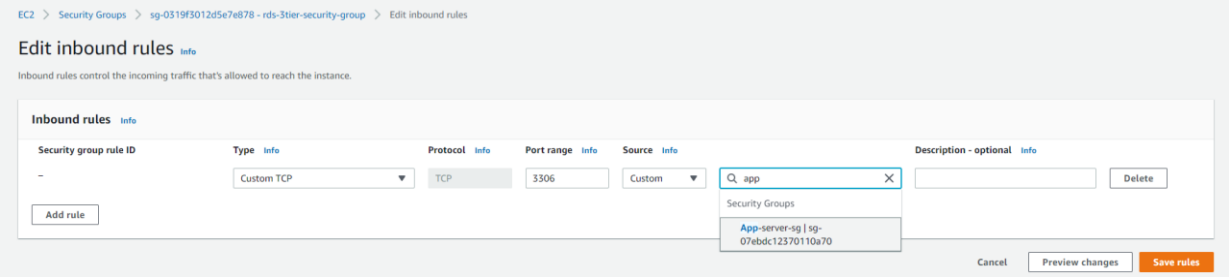
Availability zone: - No preferences



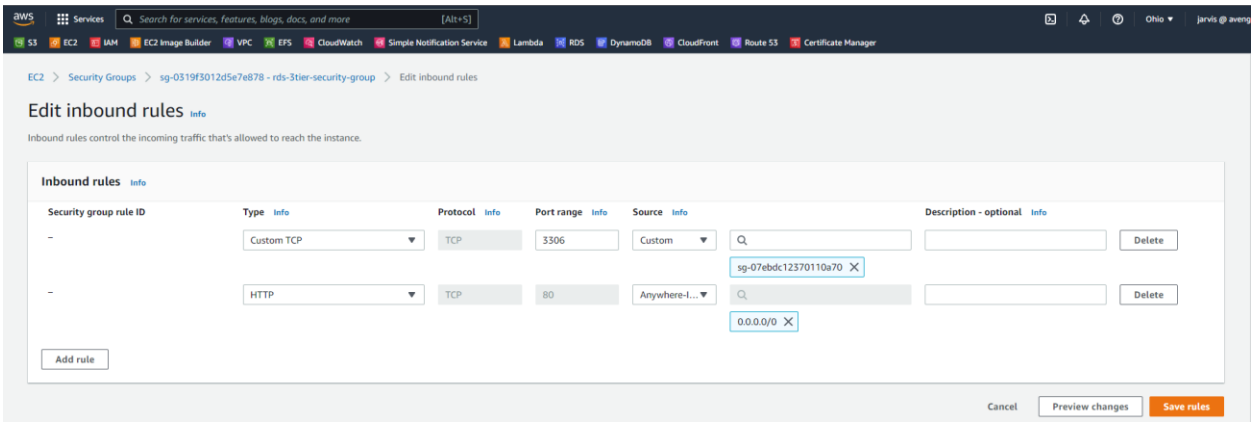
Succsesfully created Database

Step 15:- Modify securtiy group of jarvis-db

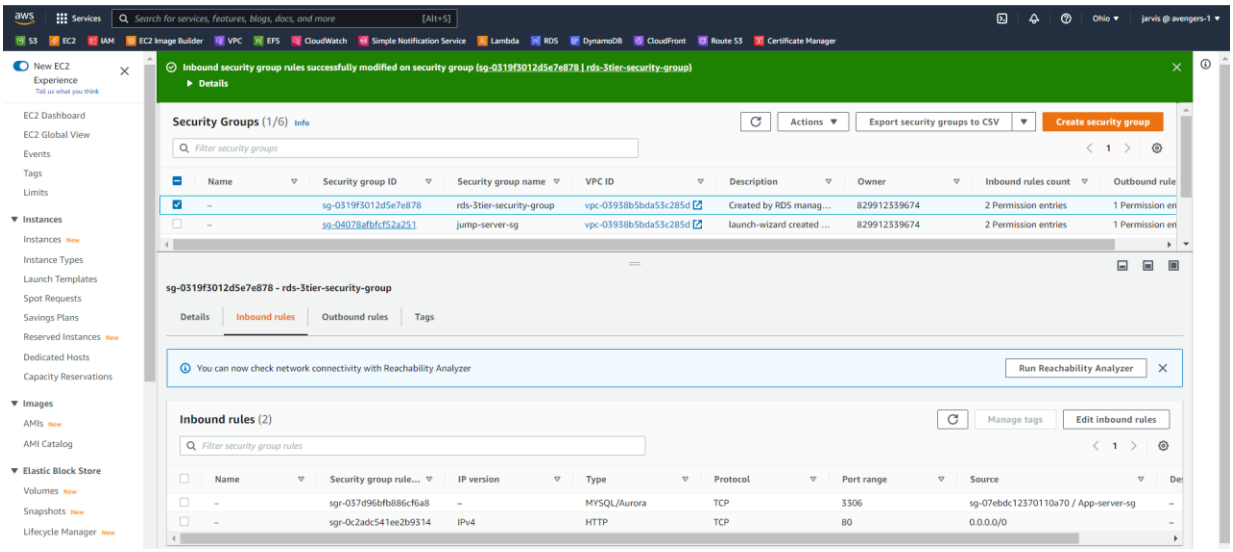
Here add Private-App-server security group [App-server-sg] into the jarvis db security group



Also add HTTP



Sucsessfully modified rds escurity group



Step 16:- Rename the file for identification in both servers

cd /var/www/html/phpMyAdmin/

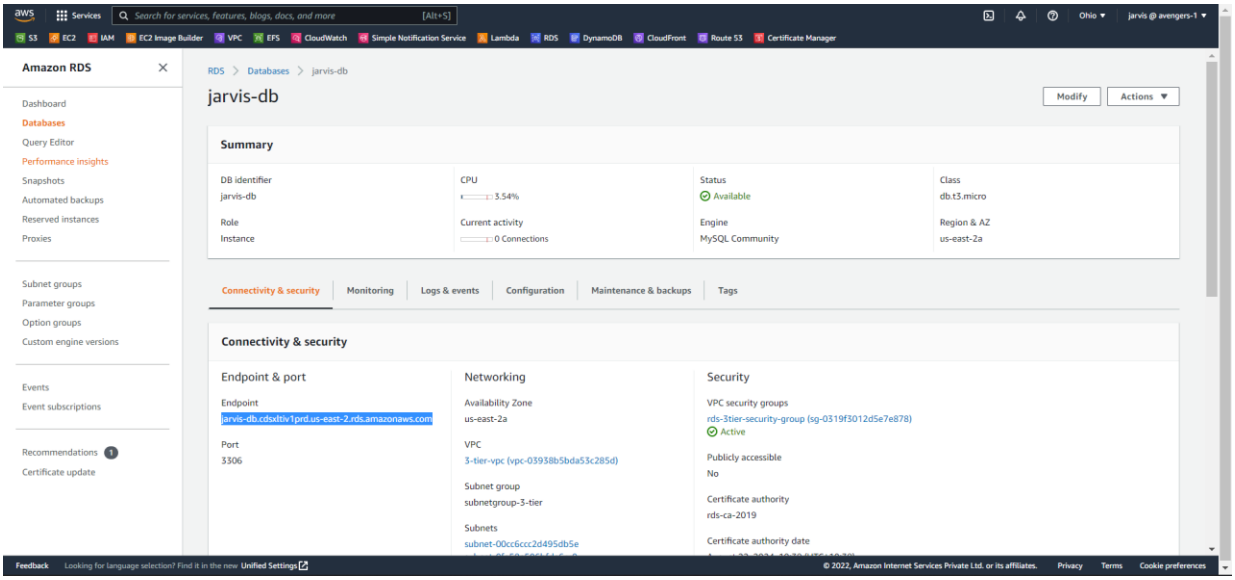
ls

mv config.sample.inc.php config.inc.php

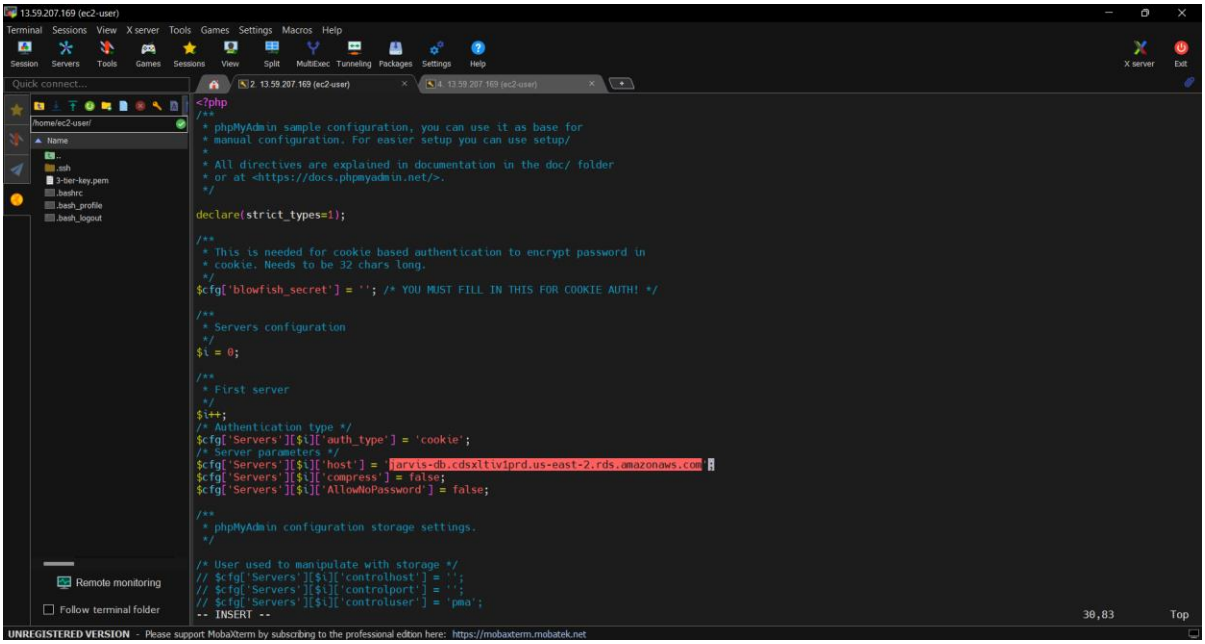
ls

vim config.inc.php

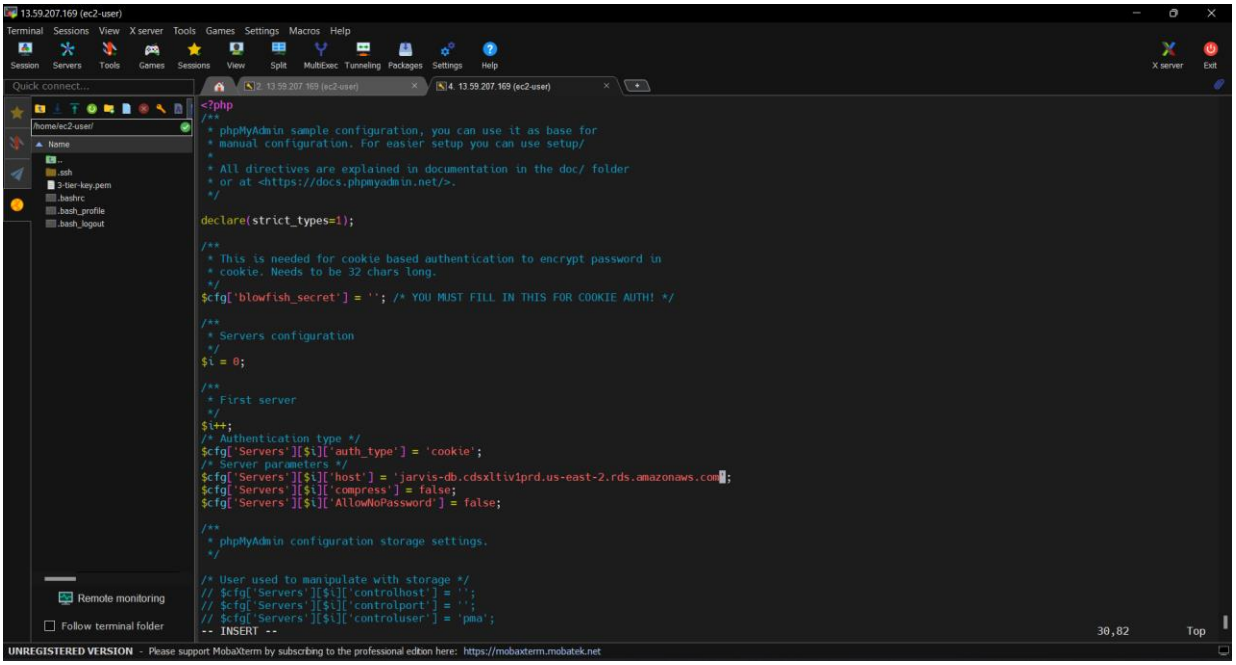
Here line no 30 paste the endpoint of [Jarvis-db] instead of localhost



Paste in Private-App-server

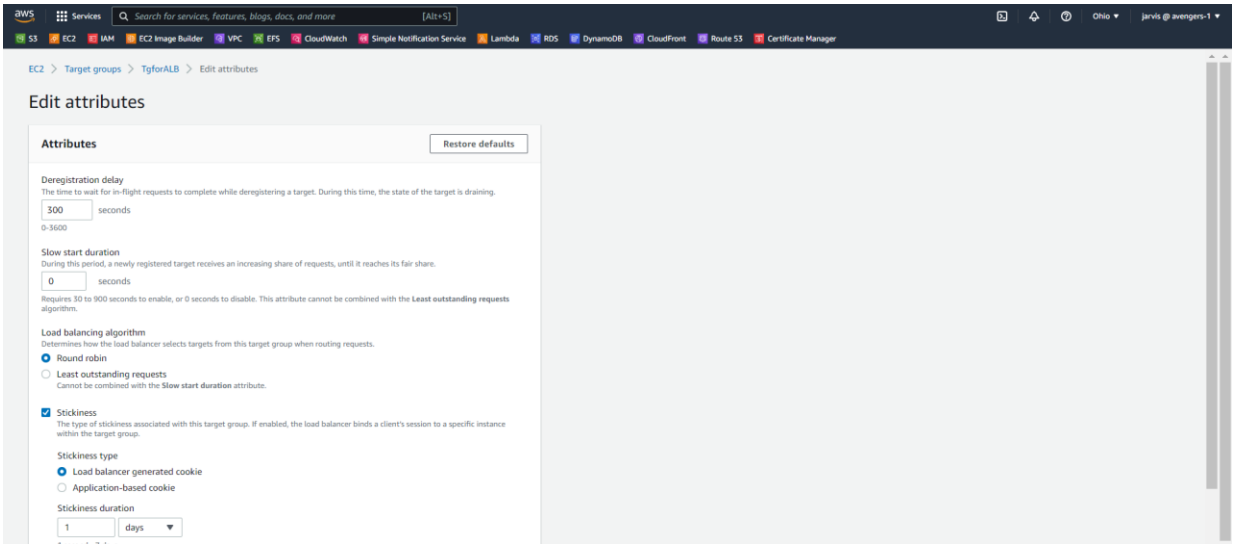


Also paste in Private-db-server



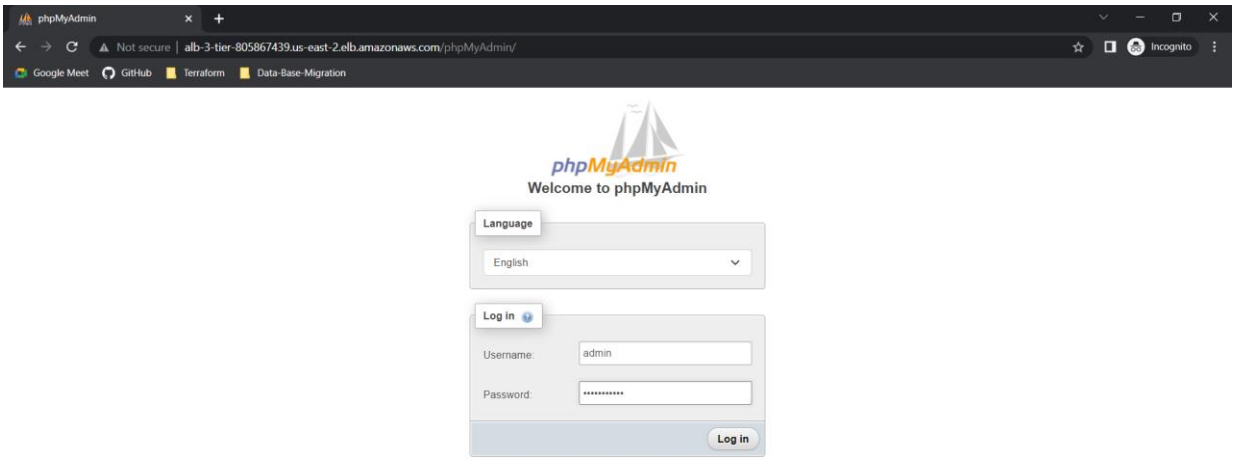
Step 17:- Go to created target group TgforALB of database and enable stickiness.

Atttributes --> Edit --> Enable Stickiness

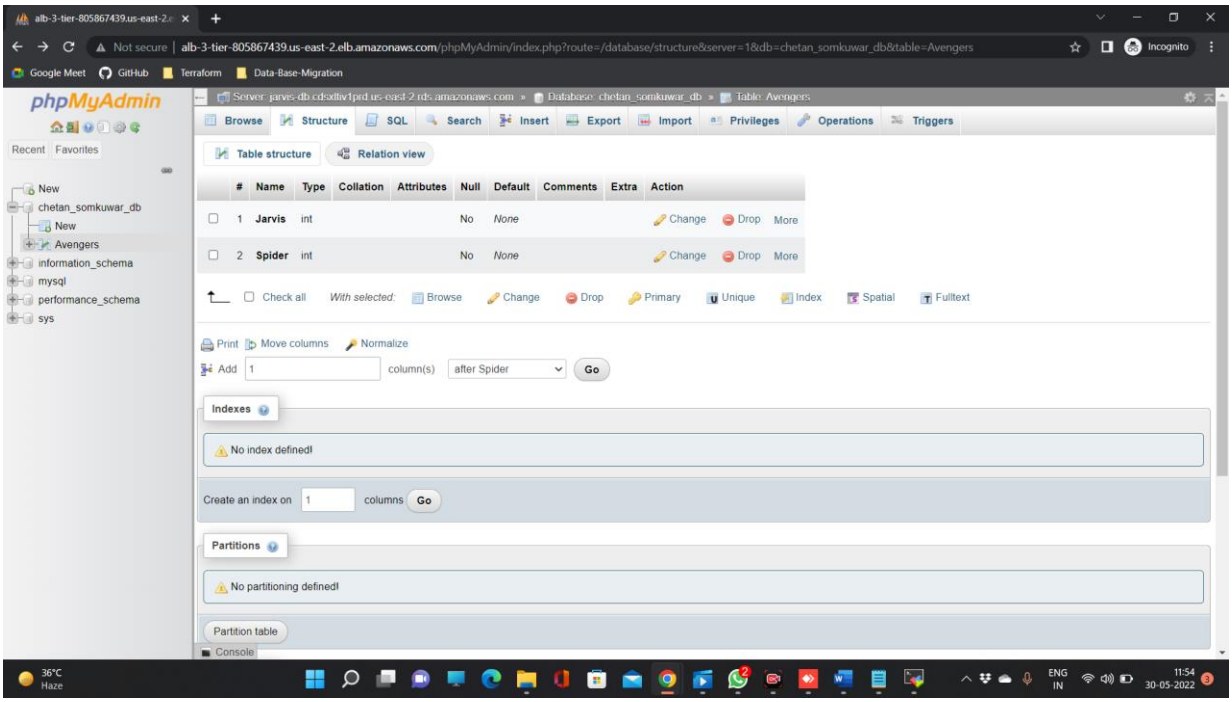


http://alb-3-tier-805867439.us-east-2.elb.amazonaws.com/phpMyAdmin/index.php

Hit DNS of Application load balancer/phpMyAdmin and login via masteruser of RDS Database

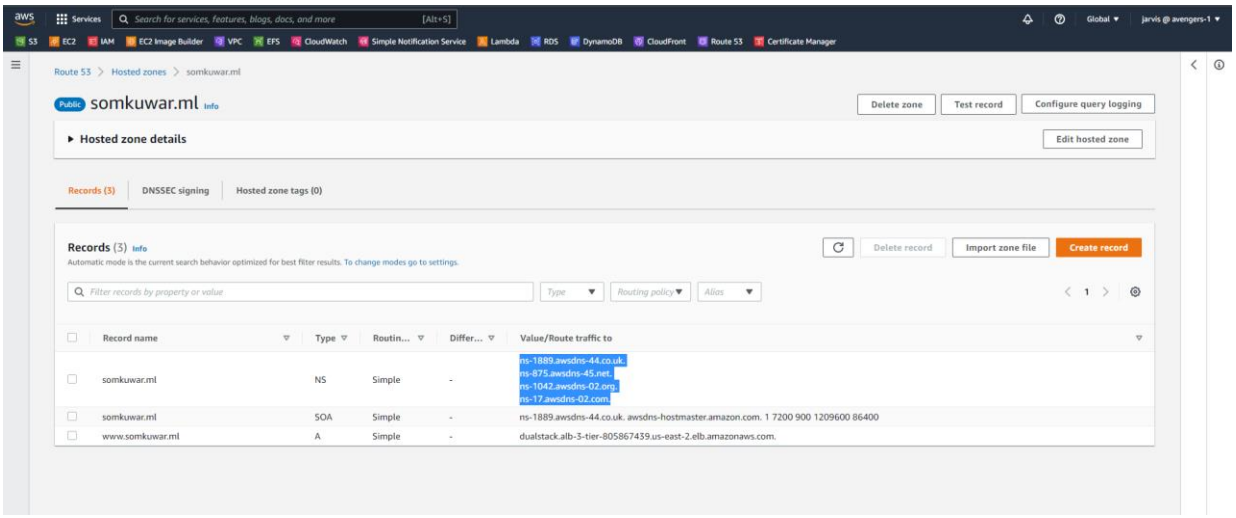


We can sucsesfully able to login and we can also able to create databse and theirt entries in private application web server.

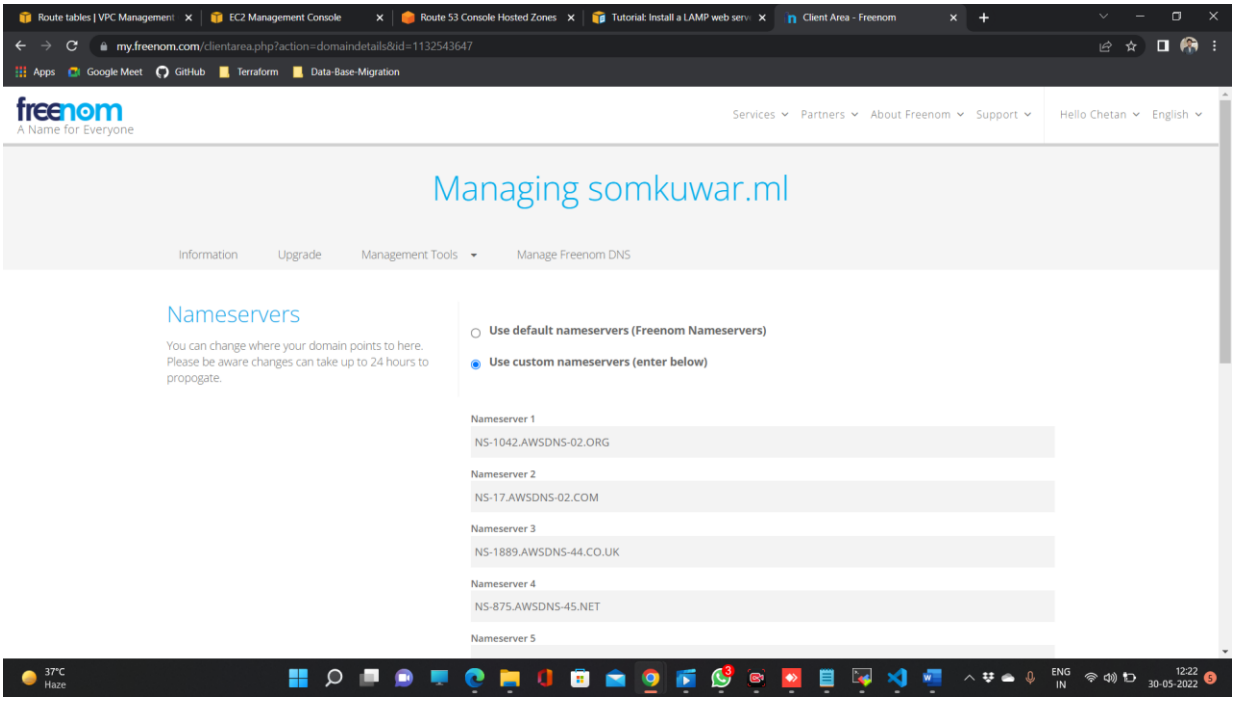


Step 18:- Create Hosted Zone in **Route53** with specified Domain Name referd in Freenom.com

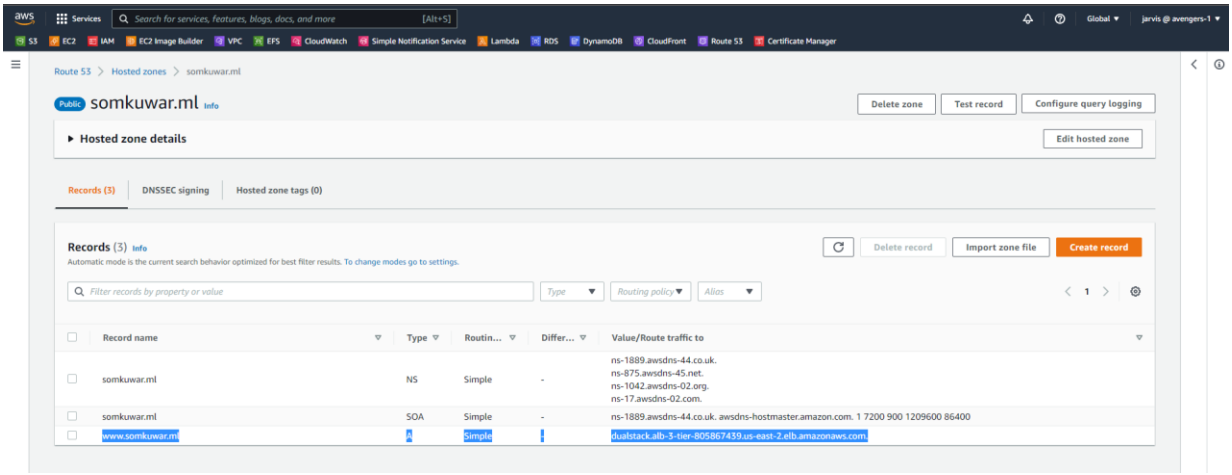
- As I get **somkuwar.ml**, so I created **somkuwar.ml** named hosted zone
- Do **NS records entry in your Somkuwa.ml** Domain in Freenom.com
- Create A record www/A-record/Alias/Application load balancer/Region/SimpleRouting



Do entry into purchased freenom domain

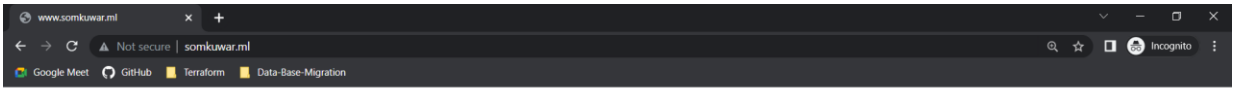


Create A record in somkuwar.ml hosted zone in Route53 serivce



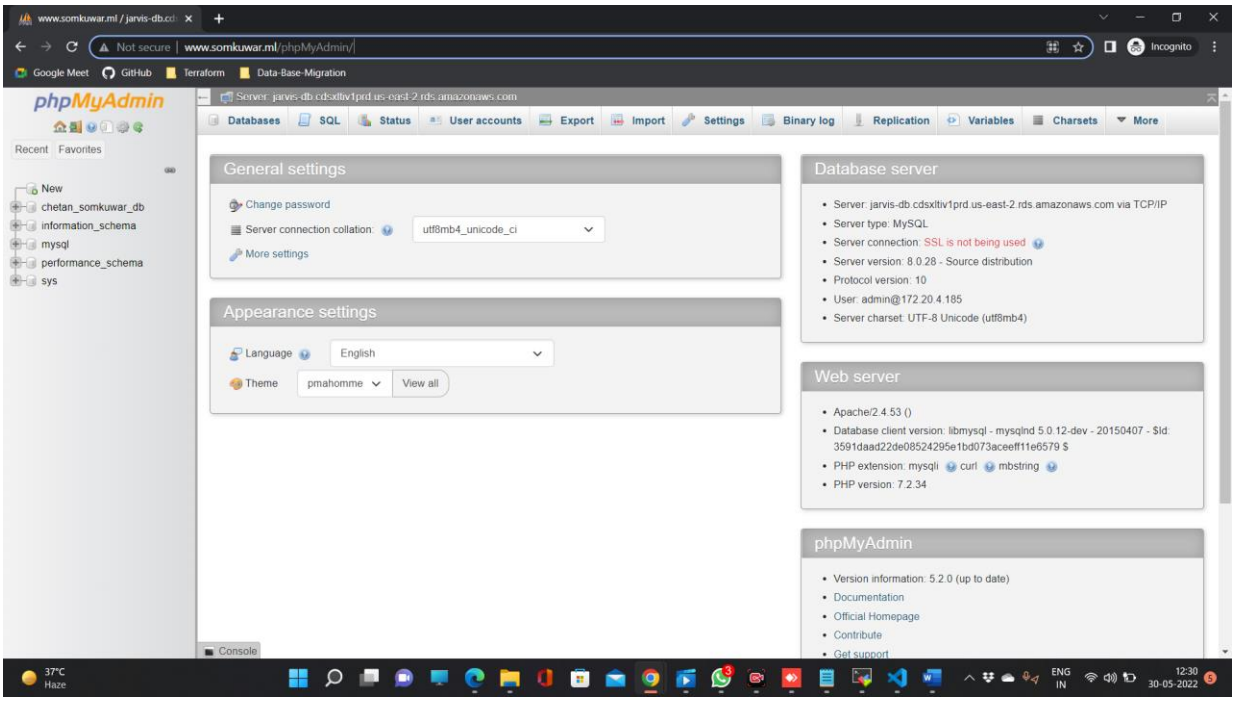
Now,

Hit www.somkuwar.ml → it shows my index page



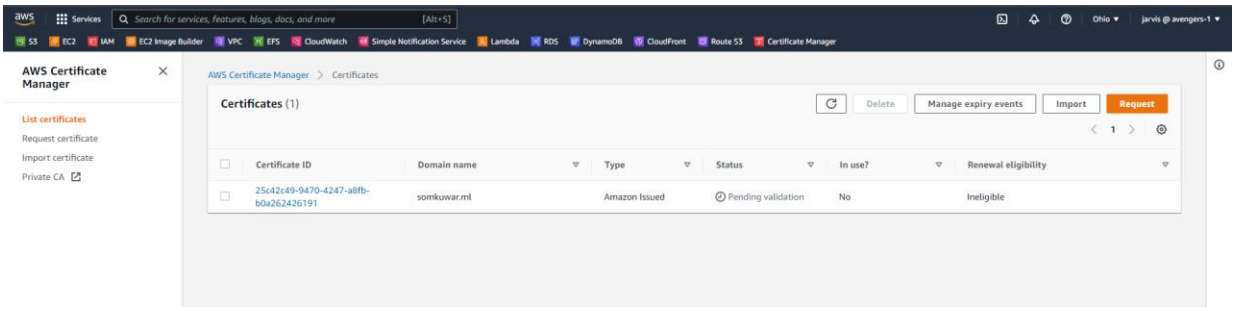
Hello chetan

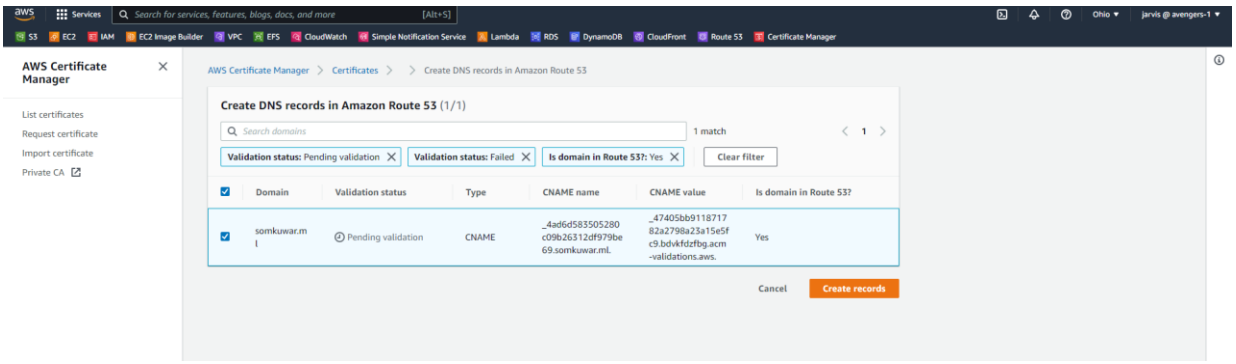
Hit www.somkuwar.ml/phpMyAdmin → it show my phpMyAdmin page and we also can also login their.

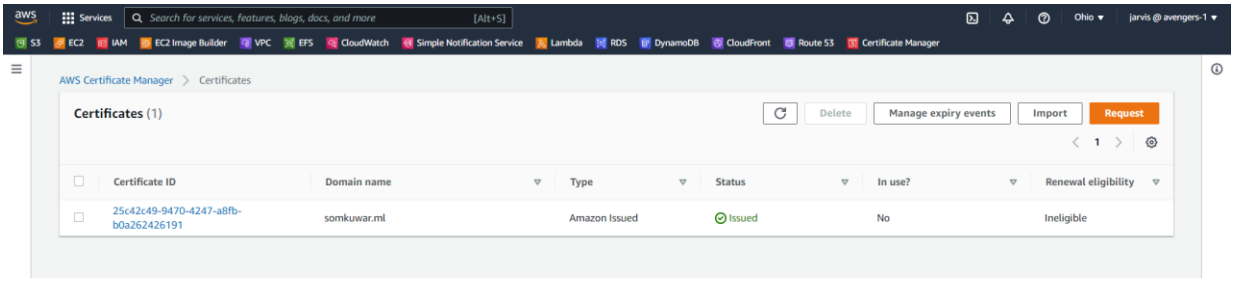


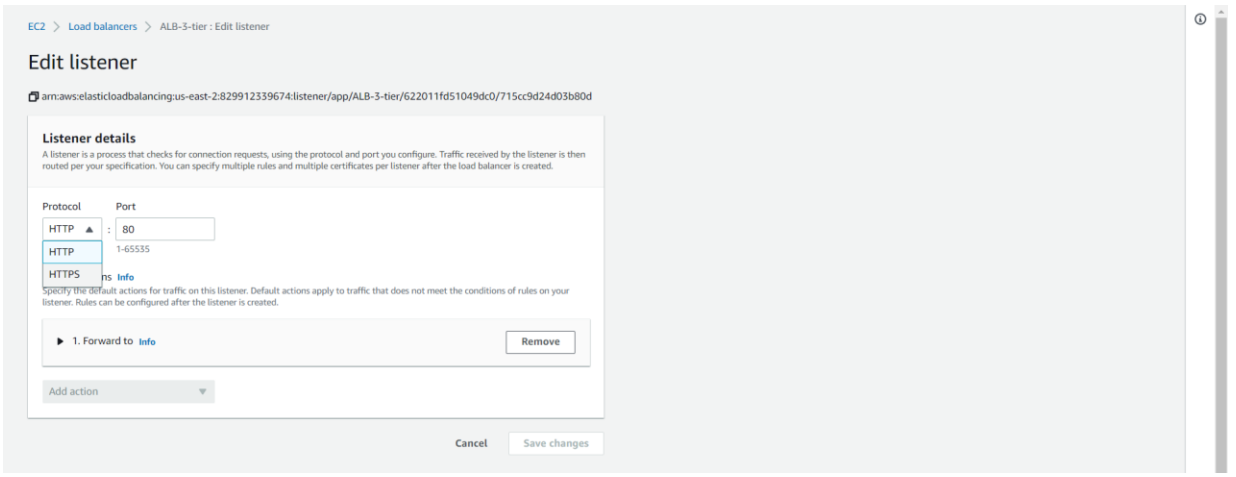
CURRETNLY IN TESTING

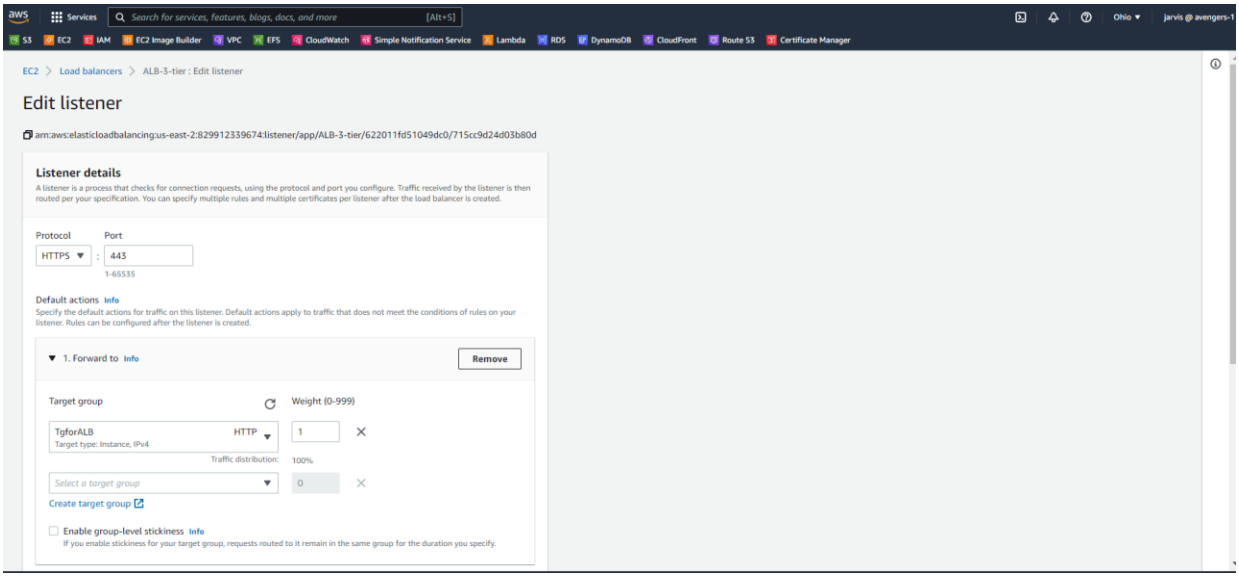
Step 19:- our website is not secured so now we issue SSL certificate from Amazon sertificate Manager and attach to our Domain then it becomes http → https

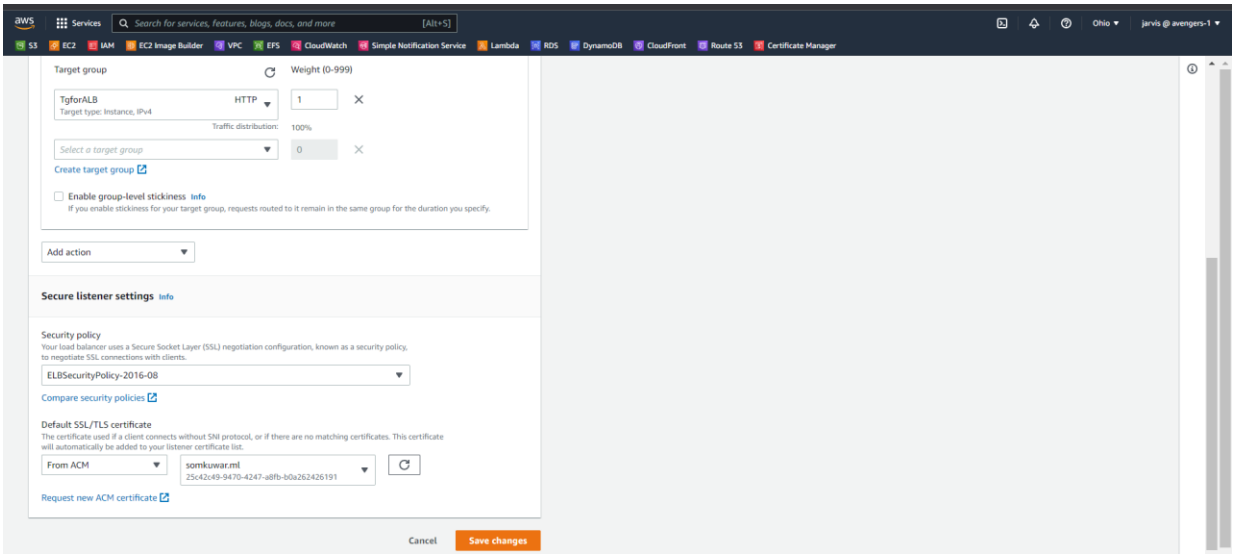







EC2 > Load balancers > ALB-3-tier : Edit listener
Edit listener
arn:aws:elasticloadbalancing:us-east-2:829912339674:listener/app/ALB-3-tier/622011fd51049dc0/715cc9d24d03b80d
Listener details
A listener is a process that checks for connection requests, using the protocol and port you configure. Traffic received by the listener is then routed per your specification. You can specify multiple rules and multiple certificates per listener after the load balancer is created.
Protocol
Port
HTTP
80
1-65535
HTTP
HTTPS
Info
Specify the default actions for traffic on this listener. Default actions apply to traffic that does not meet the conditions of rules on your listener. Rules can be configured after the listener is created.
1. Forward to Info
Remove
Add action
Cancel
Save changes


Services
Search for services, features, blogs, docs, and more
[Alt+S]
Ohio
jarvis @ avengers-1
S3
EC2
IAM
EC2 Image Builder
VPC
EFS
CloudWatch
Simple Notification Service
Lambda
RDS
DynamoDB
CloudFront
Route 53
Certificate Manager
EC2 > Load balancers > ALB-3-tier : Edit listener
Edit listener
arn:aws:elasticloadbalancing:us-east-2:829912339674:listener/app/ALB-3-tier/622011fd51049dc0/715cc9d24d03b80d
Listener details
A listener is a process that checks for connection requests, using the protocol and port you configure. Traffic received by the listener is then routed per your specification. You can specify multiple rules and multiple certificates per listener after the load balancer is created.
Protocol
Port
HTTPS
443
1-65535
Default actions Info
Specify the default actions for traffic on this listener. Default actions apply to traffic that does not meet the conditions of rules on your listener. Rules can be configured after the listener is created.
1. Forward to Info
Remove
Target group
Weight (0-999)
TgforALB
Target type: Instance, IPv4
HTTP
1
Traffic distribution:
100%
Select a target group
0
Create target group
Enable group-level stickiness Info
If you enable stickiness for your target group, requests routed to it remain in the same group for the duration you specify.


Services
Search for services, features, blogs, docs, and more
[Alt+S]
Ohio
jarvis @ avengers-1
S3
EC2
IAM
EC2 Image Builder
VPC
EFS
CloudWatch
Simple Notification Service
Lambda
RDS
DynamoDB
CloudFront
Route 53
Certificate Manager
Target group
Weight (0-999)
TgforALB
Target type: Instance, IPv4
HTTP
1
Traffic distribution:
100%
Select a target group
0
Create target group
Enable group-level stickiness Info
If you enable stickiness for your target group, requests routed to it remain in the same group for the duration you specify.
Add action
Secure listener settings Info
Security policy
Your load balancer uses a Secure Socket Layer (SSL) negotiation configuration, known as a security policy, to negotiate SSL connections with clients.
ELBSecurityPolicy-2016-08
Compare security policies
Default SSL/TLS certificate
The certificate used if a client connects without SNI protocol, or if there are no matching certificates. This certificate will automatically be added to your listener certificate list.
From ACM
somkuwar.ml
25c42c49-9470-4247-a8fb-b0a262426191
Request new ACM certificate
Cancel
Save changes