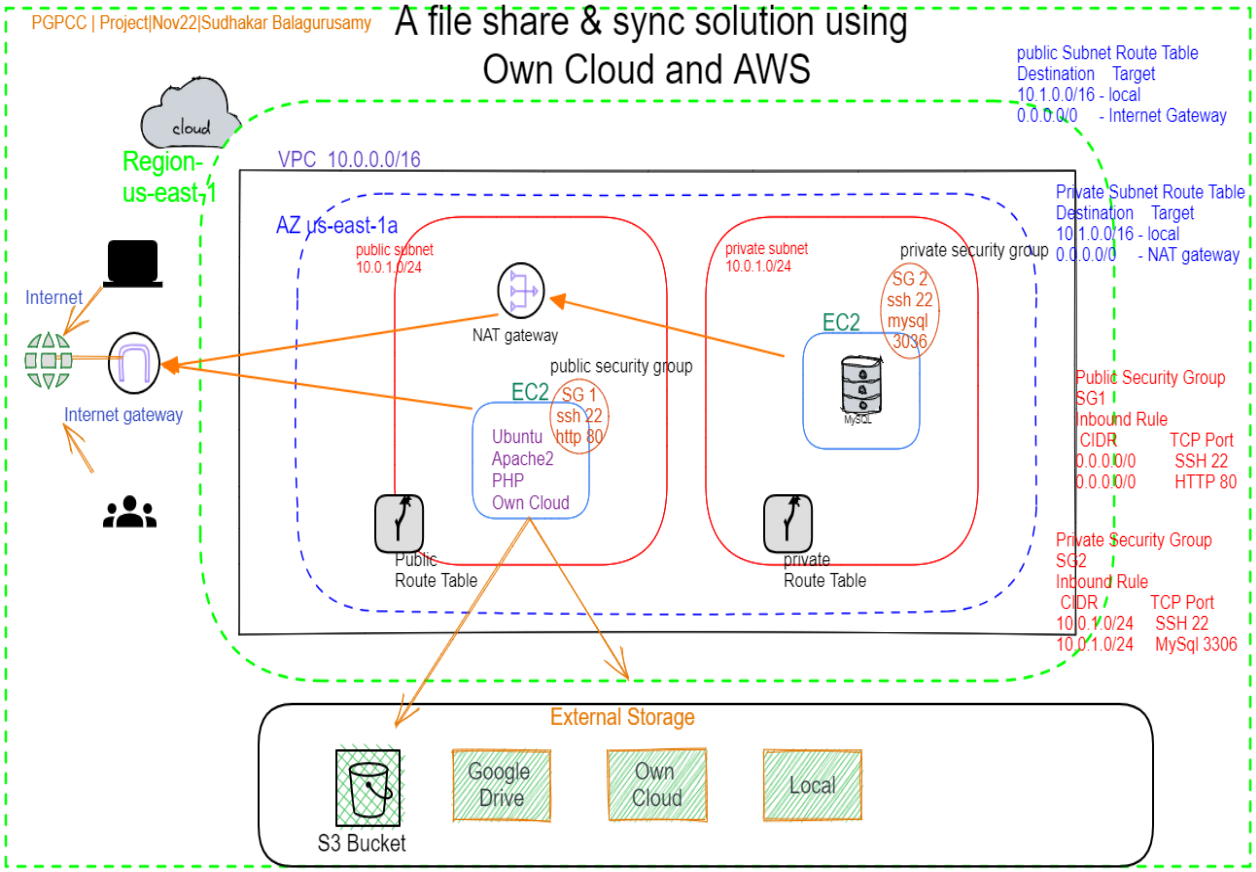
Architecture Diagram



Network Planning:

Create a VPC (10.0.0.0/16) with 2 subnets (10.0.1/24 and 10.0.2.0/24)

Public facing webserver will be in public the subnet (10.0.1/24) and database server will be in the private subnet (10.0.2.0/24)

Routing tables need to be created to direct the traffic to internet and NAT gateway and associated with subnets.

NAT gateway is created to provide internet access to private subnet.

Internet gateway is created for providing internet access to public subnet/VPC

Public Security group is created for SSH /HTTP access to EC2 instances in public subnet.

Private Security group is created for SSH /MYSQL access to EC2 instances in private subnet from public subnet.

The following scripts have been implemented for this project.

Location	Scripts	Description
CLI enabled bash terminal	Create_aws_resources_cli.bash	CLI commands to create the resources needed for this project. The region needs to be set as US-EAST-1 in the global configuration file.
DB Server (private subnet)	Install_mysql.bash	To install MySQL and create owncloud user in MYSQL
APP Server (public subnet)	Install_prerequisite.bash	Bash script to install Apache, PHP, Redis
APP Server (public subnet)	Config_apache2.bash	To configure apache2 for owncloud
APP Server (public subnet)	Install_owncloud.bash	To install own cloud component

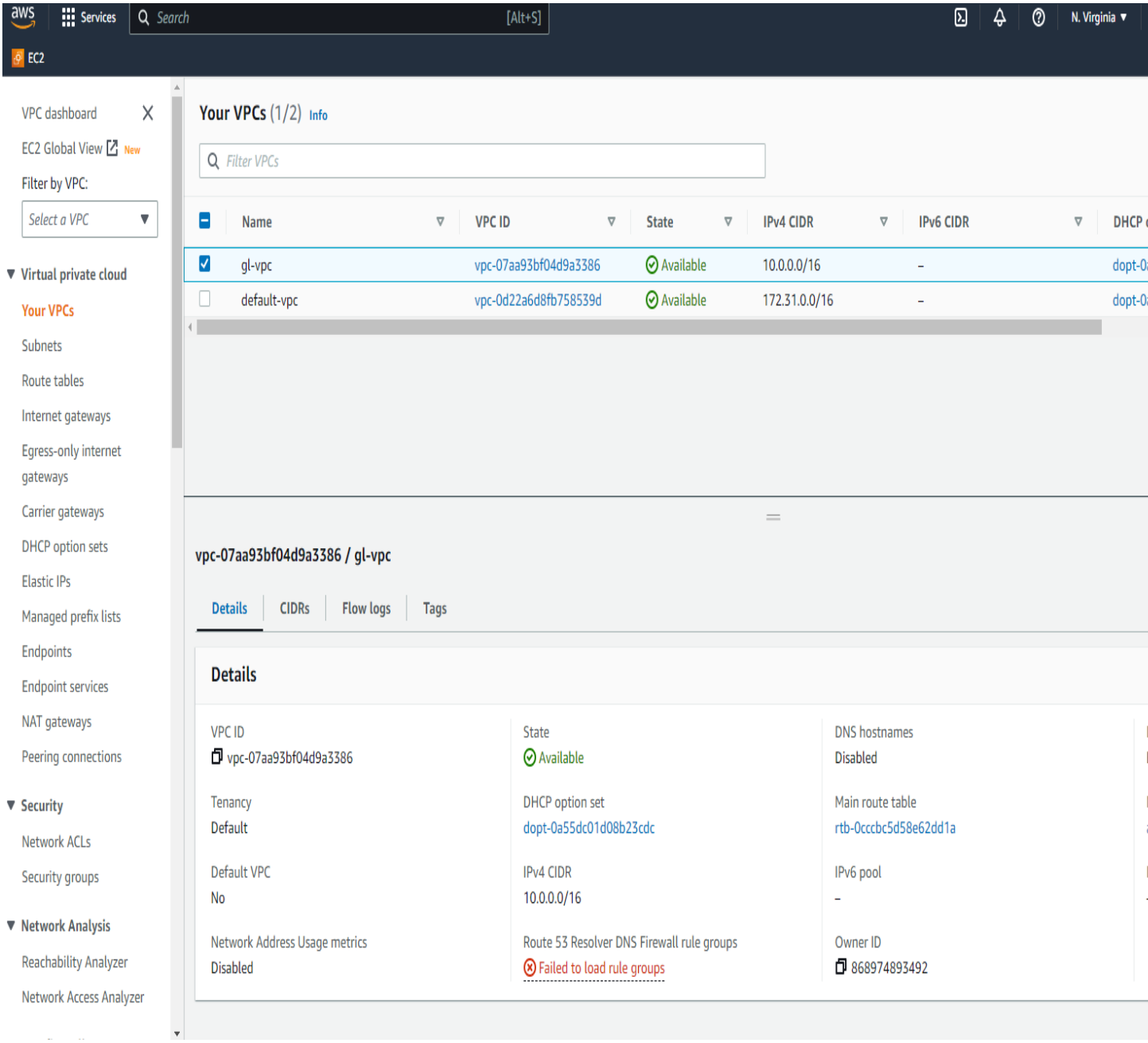
Workflow Steps

Step 1

Create a new VPC :gl-vpc with a CIDR block 10.0.0.0/16	
AWS CLI (bash)	gl_vpc=$(aws ec2 create-vpc \ --cidr-block 10.0.0.0/16 \ --query Vpc.VpcId \ --output text \

	--tag-specification "ResourceType=vpc,Tags=[{Key=Name,Value=gl-vpc}]")
AWS Console	1. VPC 2. Your VPCs 3. Create VPC

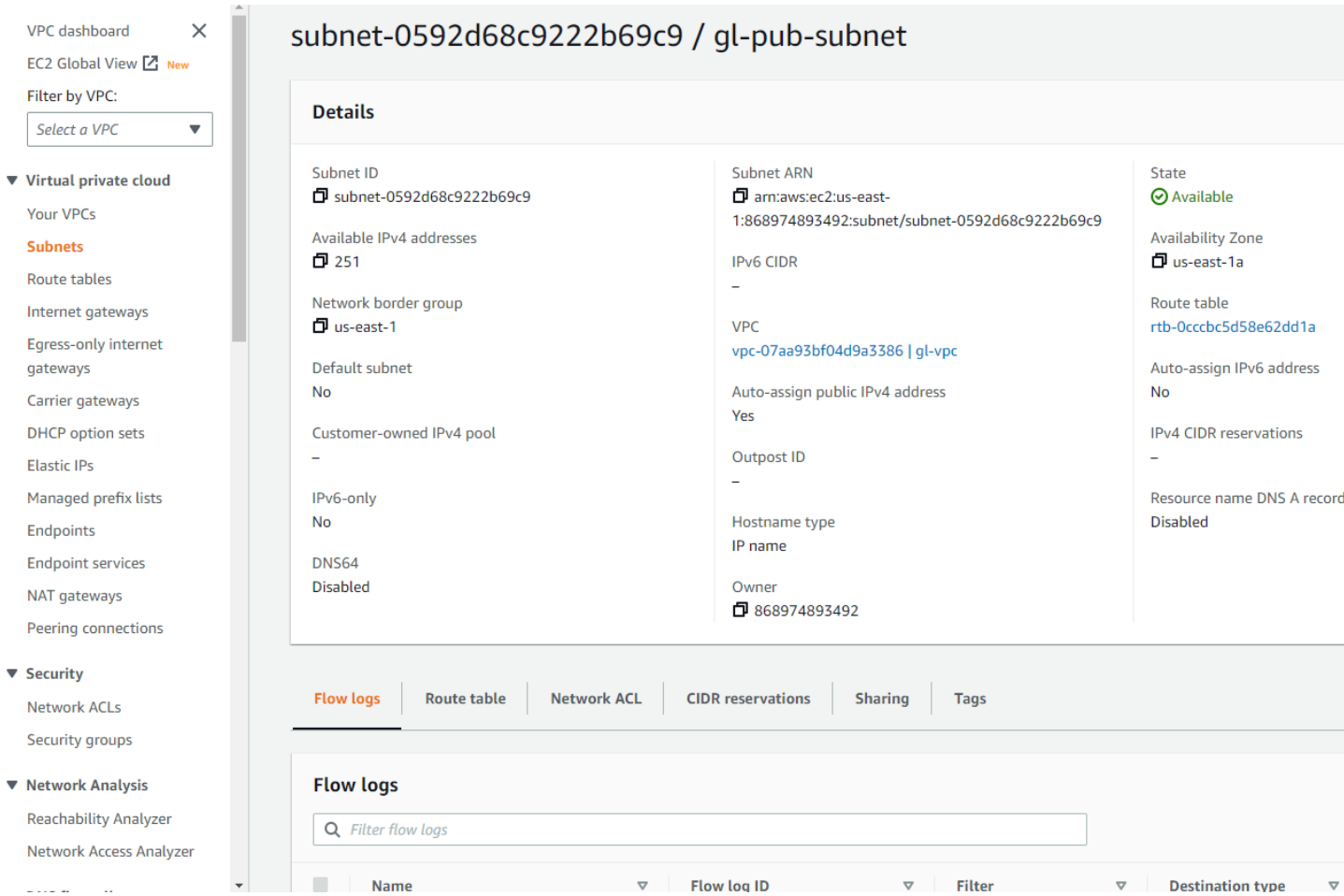
Screenshot



Step 2

Create a public subnet with a CIDR block 10.0.1.0/24 in gl-vpc Enable Public IP for the resources on this subnet	
AWS CLI (bash)	gl_pub_subnet=$(aws ec2 create-subnet \ --vpc-id $gl_vpc \ --availability-zone us-east-1a \ --cidr-block 10.0.1.0/24 \ --query Subnet.SubnetId \ --output text \ --tag-specifications "ResourceType=subnet,Tags=[{Key=Name,Value=gl-pub-subnet}]") aws ec2 modify-subnet-attribute --subnet-id $gl_pub_subnet --map-public-ip-on-launch
AWS Console	1. VPC 2. Subnets 3. Create subnet

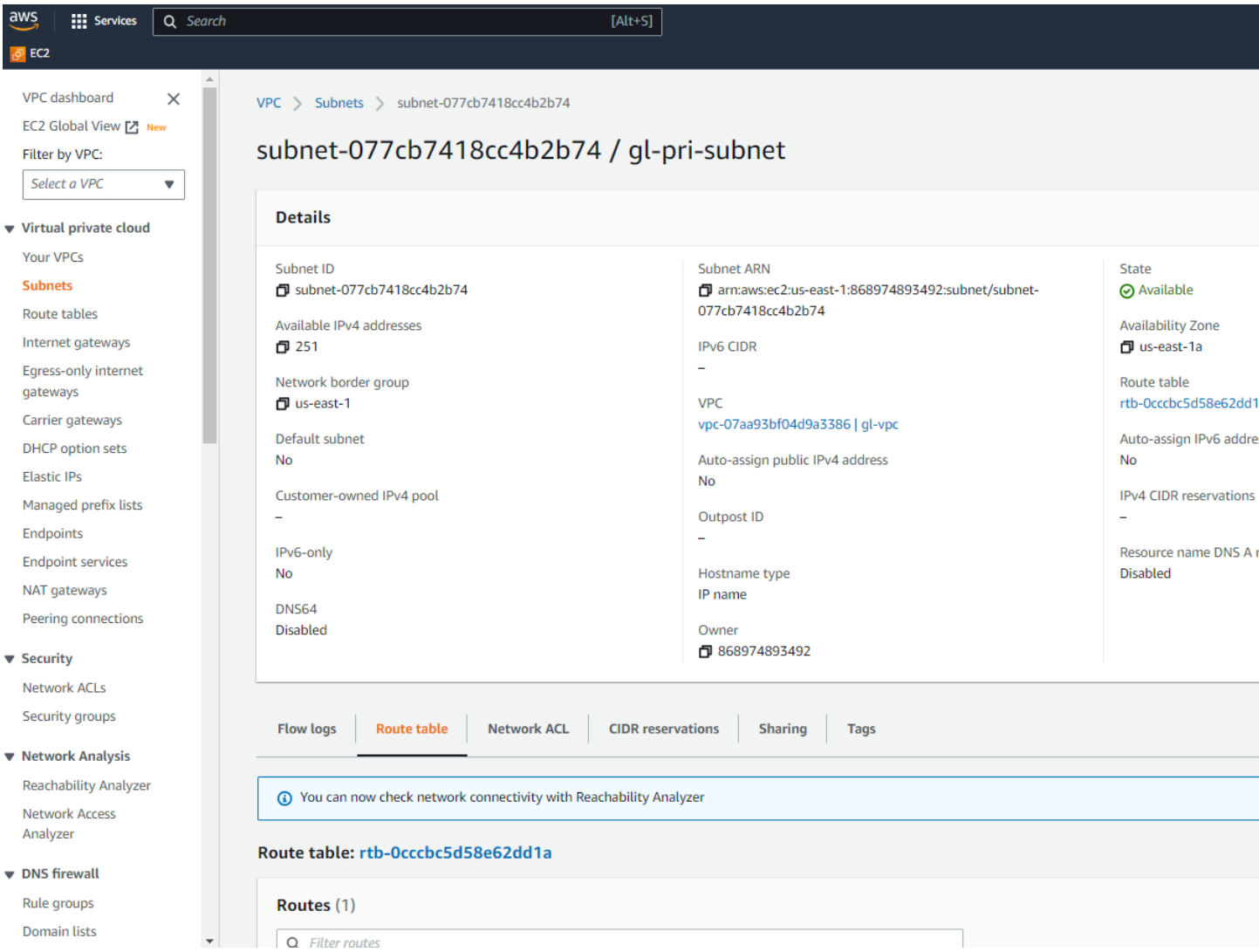
Screenshot



Step 3

Create a private subnet with a CIDR block 10.0.2.0/24 in gl-vpc	
AWS CLI (bash)	gl_pri_subnet=$(aws ec2 create-subnet \ --vpc-id $gl_vpc \ --availability-zone us-east-1a \ --cidr-block 10.0.2.0/24 \ --query Subnet.SubnetId \ --output text \ --tag-specifications "ResourceType=subnet,Tags=[{Key=Name,Value=gl-pri-subnet}]")
AWS Console	1. VPC 2. Subnets 3. Create subnet

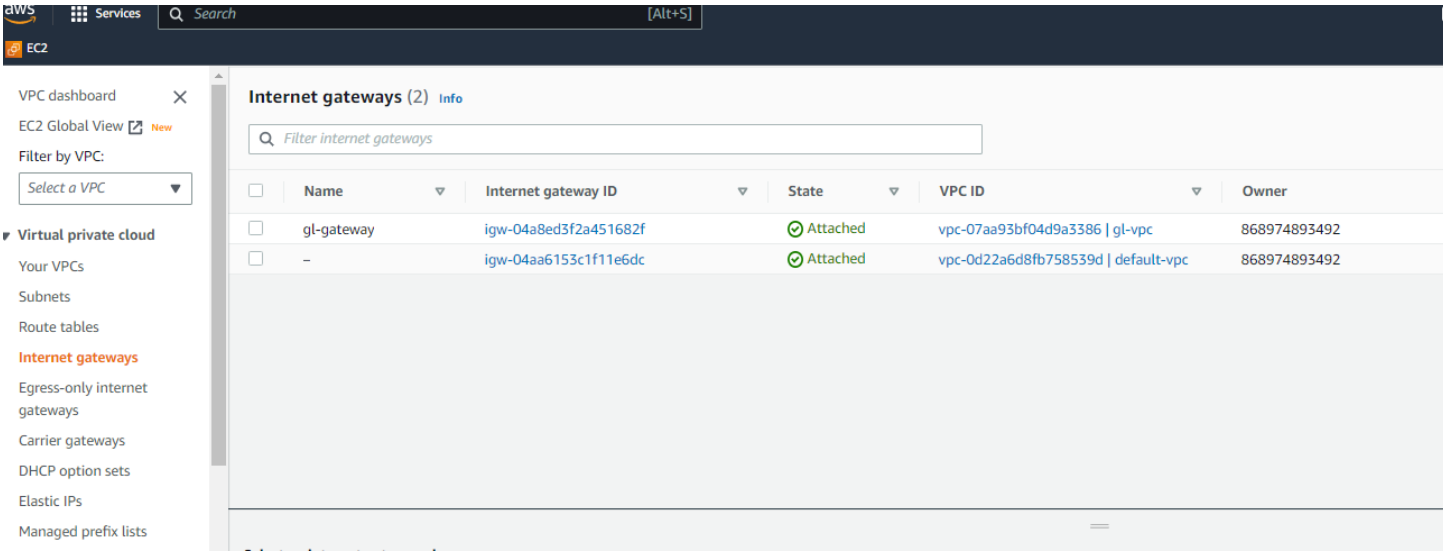
Screenshot



Step 4

Create an internet gateway gl-gateway and attach it to gl-vpc	
AWS CLI (bash)	gl_gateway=$(aws ec2 create-internet-gateway --query InternetGateway.InternetGatewayId \ --output text --tag-specifications "ResourceType=internet-gateway,Tags=[{Key=Name,Value=gl-gateway}]") aws ec2 attach-internet-gateway --vpc-id $gl_vpc --internet-gateway-id $gl_gateway
AWS Console	1. VPC 2. Internet gateways 3. Create internet gateway

Screenshot

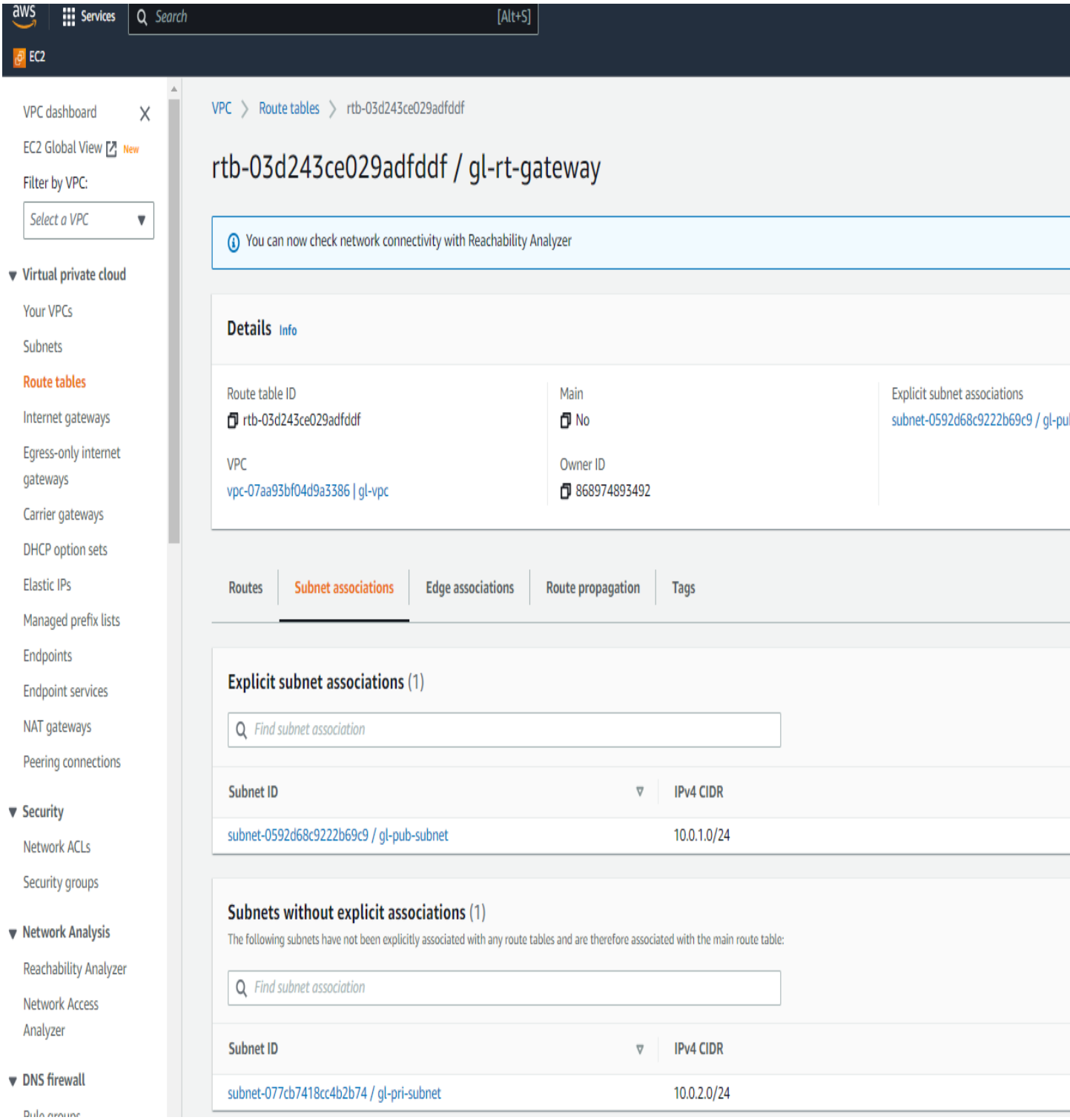


Step 5

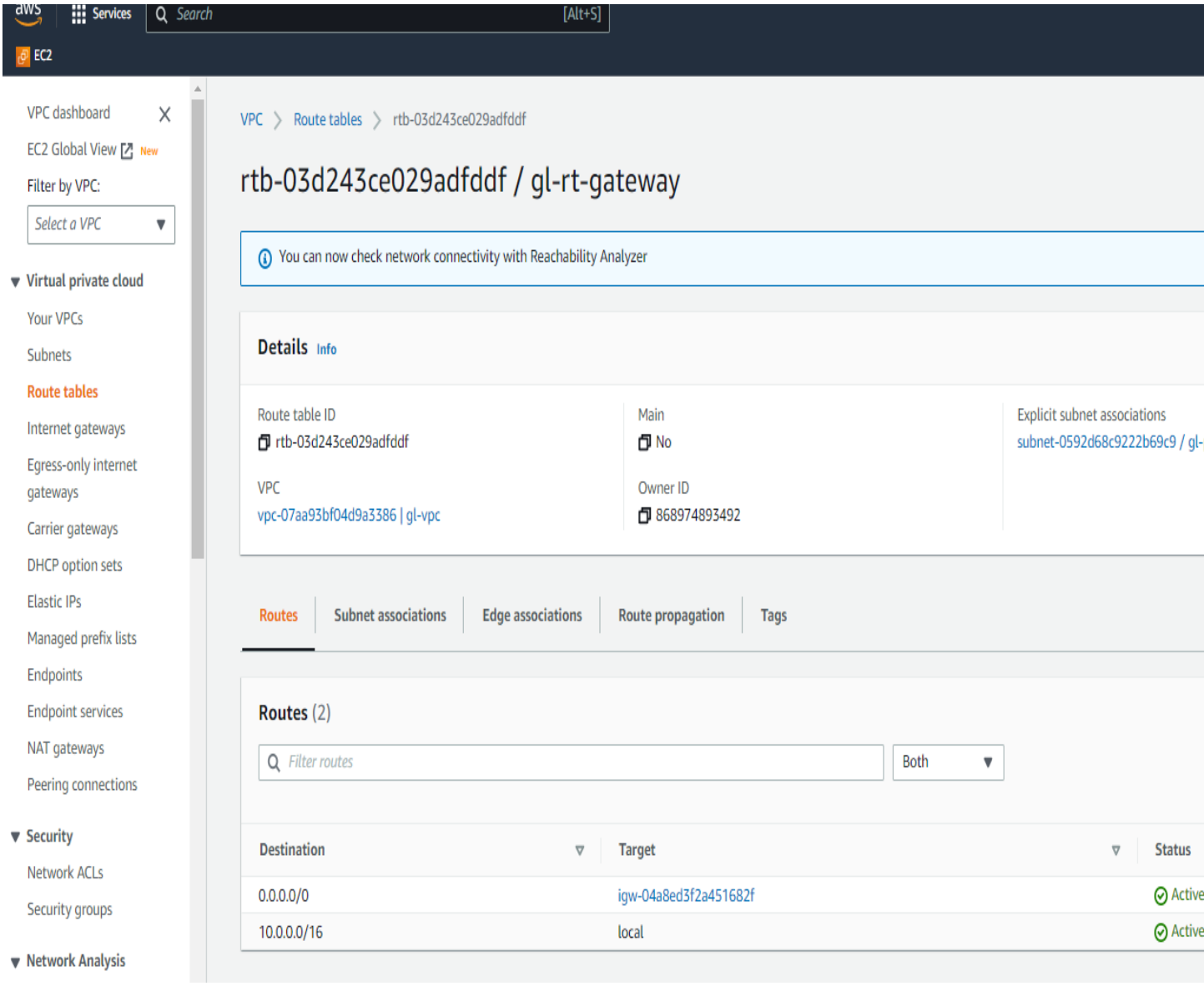
Create a route table for the public subnet to direct the traffic to gl-gateway And associate this to public subnet	
AWS CLI (bash)	gl_rt_gateway=$(aws ec2 create-route-table --vpc-id $gl_vpc --query RouteTable.RouteTableId --output text \ --tag-specifications "ResourceType=route-table,Tags=[{Key=Name,Value=gl-rt-gateway}]") #create an entry to route the 0.0.0.0/0 traffic to the gateway aws ec2 create-route --route-table-id $gl_rt_gateway --destination-cidr-block 0.0.0.0/0 --gateway-id $gl_gateway #associate the gl_rt_gateway to public subnet aws ec2 associate-route-table --subnet-id $gl_pub_subnet --route-table-id $gl_rt_gateway
AWS Console	1. VPC 2. Route tables 3. Create route table

Screenshot

Route table with subnet association



Routes in the public route table

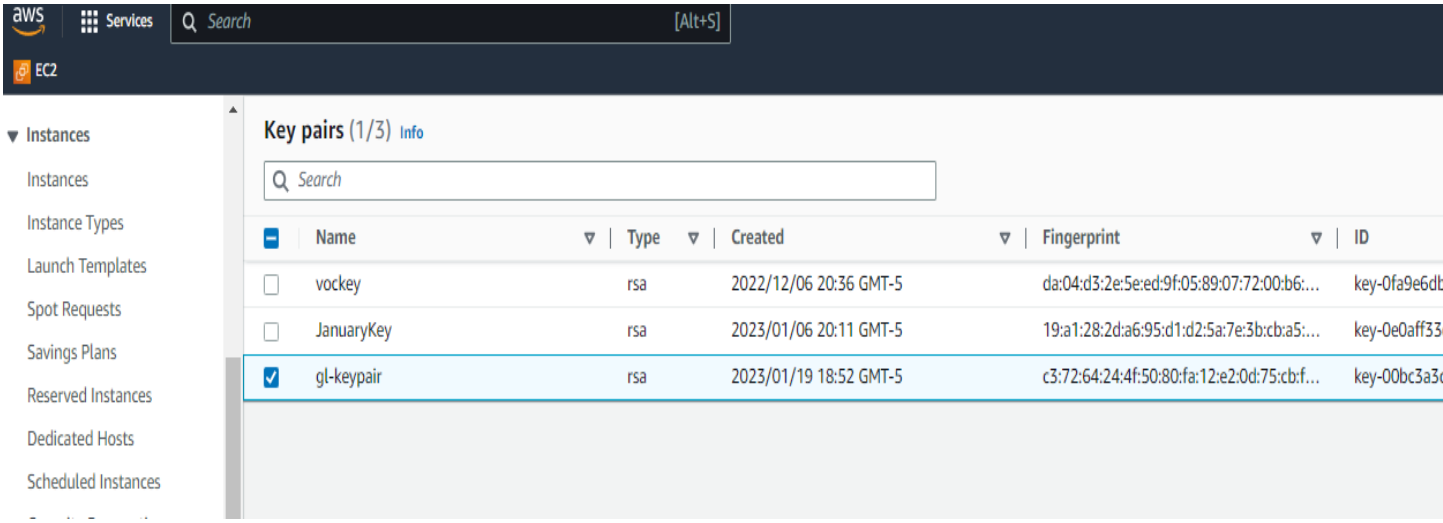


Step 6

Create a key pair gl-keypair	
AWS CLI (bash)	gl_keypair="gl-keypair" aws ec2 create-key-pair --key-name $gl_keypair --query "KeyMaterial" --output text > $gl_keypair.pem chmod 400 $gl_keypair
AWS Console	1. EC2 2. Key pairs

	3. Create key pair

Screenshot



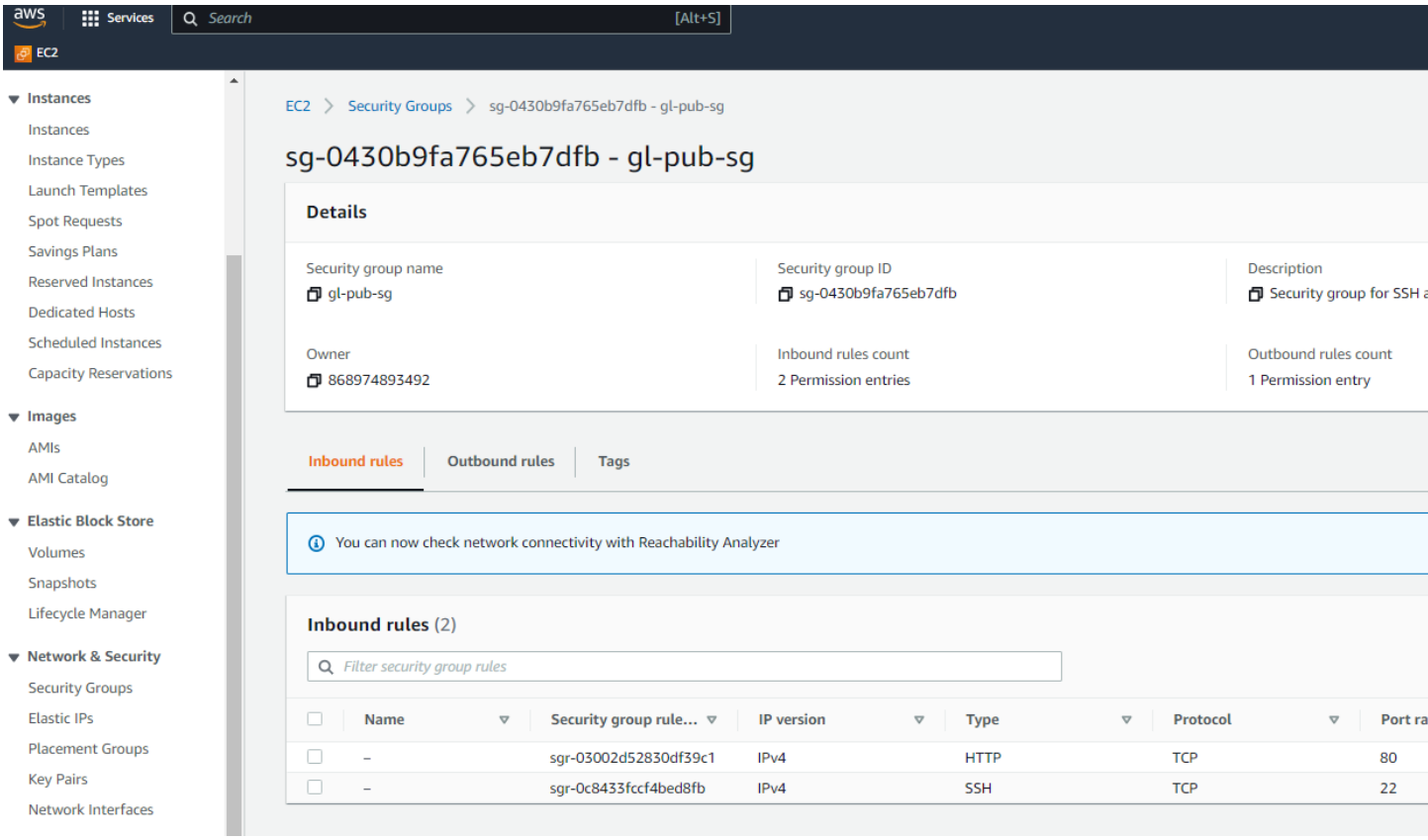
Step 7

Create a security group gl-pub-sg for public subnet EC2 instances. This allows SSH & HTTP ports to be accessed from internet	

AWS CLI (bash)	#create a security group for public subnet resources
	gl_pub_sg=$(aws ec2 create-security-group --group-name gl-pub-sg --description "Security group for SSH access" \
	--vpc-id $gl_vpc \
	--query GroupId --output text)
	
	#authorize the security group for ssh access
	aws ec2 authorize-security-group-ingress --group-id $gl_pub_sg --protocol tcp --port 22 --cidr 0.0.0.0/0
	aws ec2 authorize-security-group-ingress --group-id $gl_pub_sg --protocol tcp --port 80 --cidr 0.0.0.0/0

AWS Console	1. EC2 2. Security Groups 3. Create security group

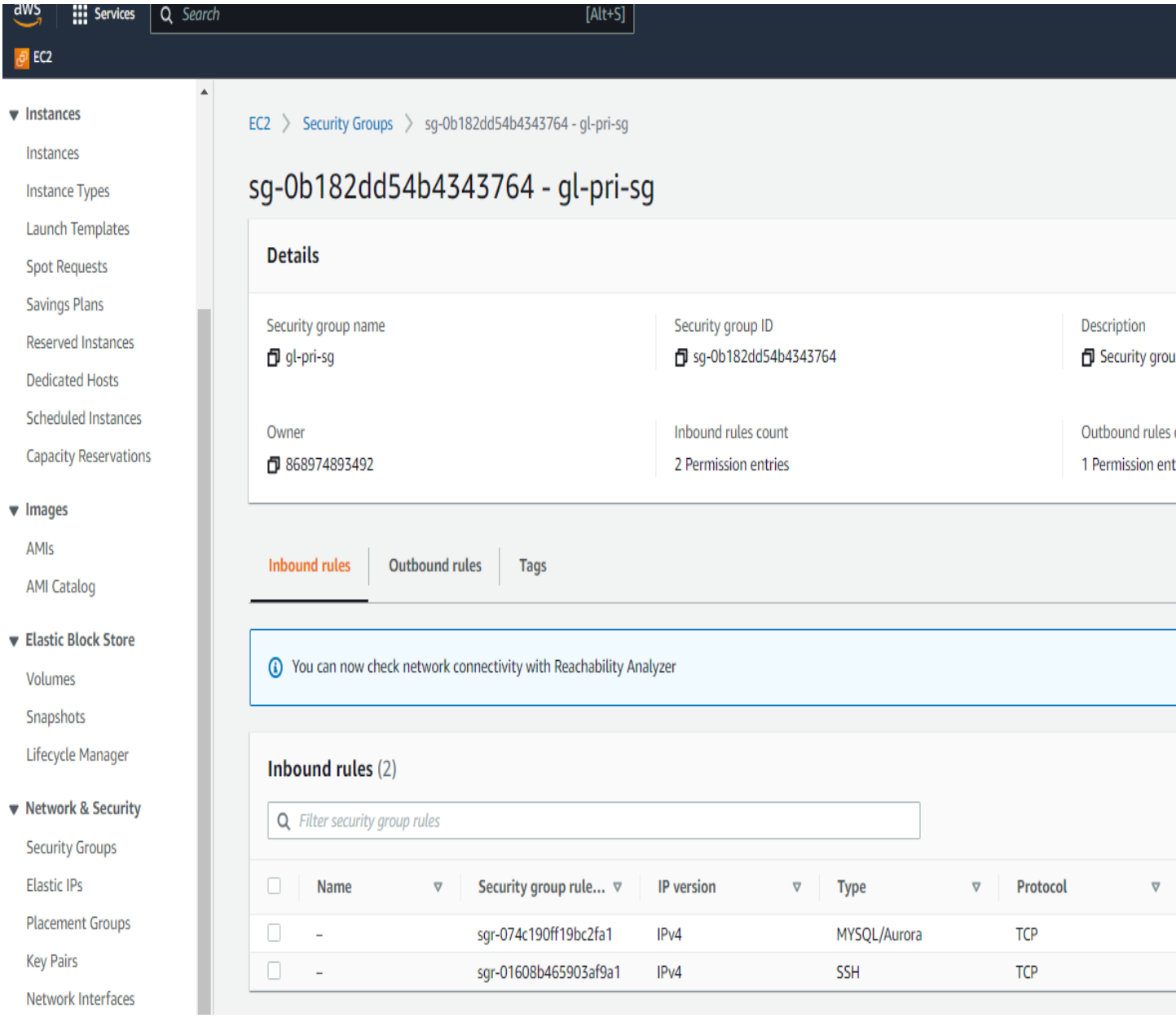
Screenshot



Step 8

Create a security group gl-pri-sg for private subnet EC2 instances. This allows SSH & MySQL ports to be accessed from the public subnet	
AWS CLI (bash)	gl_pri_sg=$(aws ec2 create-security-group --group-name \ gl-pri-sg --description "Security group for SSH & MYSQL access from public subnet" \ --vpc-id $gl_vpc \ --query GroupId --output text) aws ec2 authorize-security-group-ingress --group-id $gl_pri_sg --protocol tcp --port 22 --cidr 10.0.1.0/24 aws ec2 authorize-security-group-ingress --group-id $gl_pri_sg --protocol tcp --port 3306 --cidr 10.0.1.0/24
AWS Console	1. EC2 2. Security Groups 3. Create security group

Screenshot



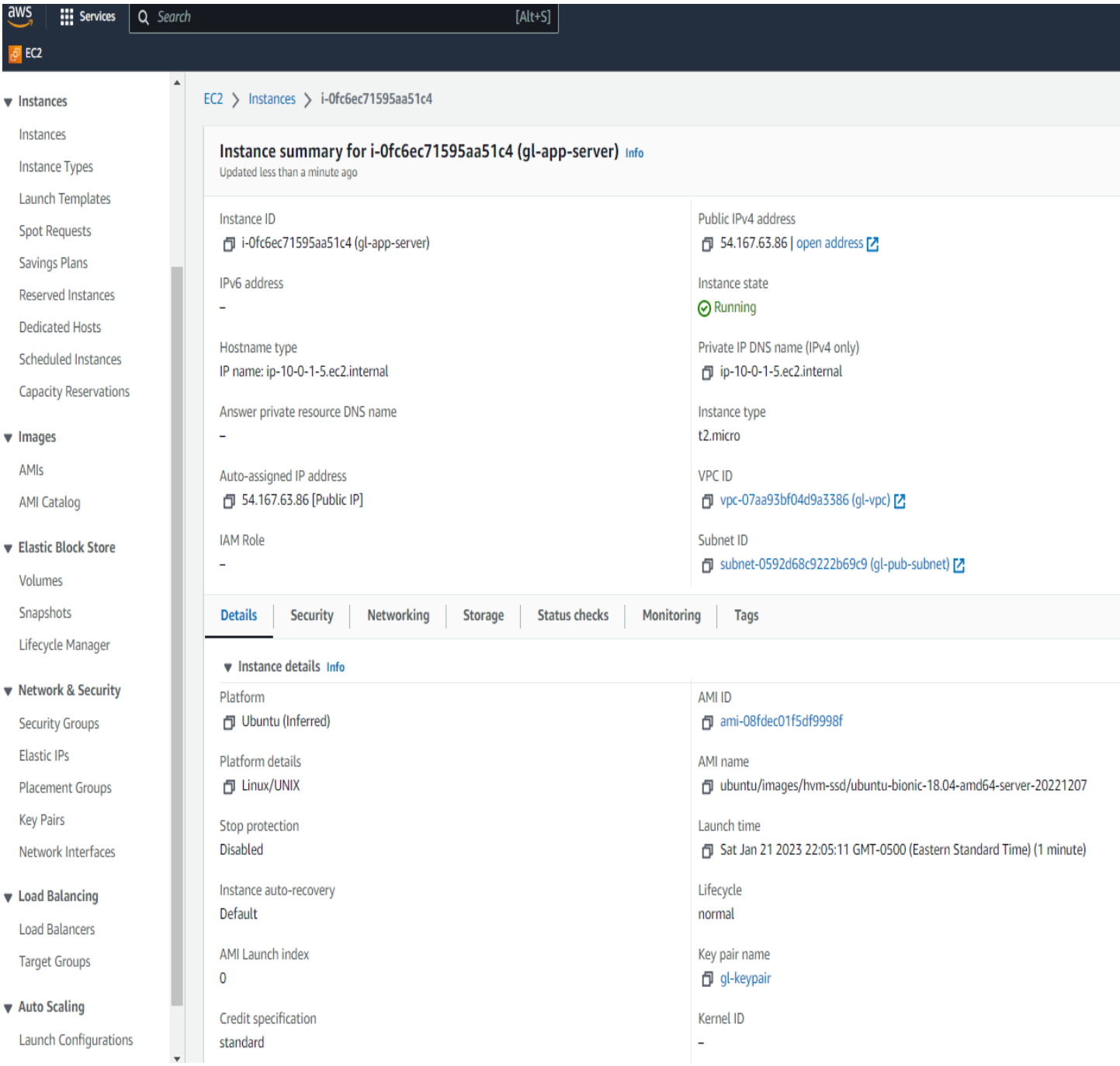
Step 9

Create an EC2 Ubuntu 18_04 instance (gl-app-server) to host own cloud application and webserver in the public subnet.
Associate the gl-pub-sg security group to it.
ami-08fdec01f5df9998f is used for 18_04 Ubuntu

AWS CLI (bash)	ubuntu_18_04="ami-08fdec01f5df9998f"
	
	gl_app_server=$(aws ec2 run-instances --image-id $ubuntu_18_04 --count 1 --instance-type t2.micro --key-name $gl_keypair \
	--security-group-ids $gl_pub_sg \
	--subnet-id $gl_pub_subnet \
	--associate-public-ip-address \
	--tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=gl-app-server}]' \
	--query Instances[0].InstanceId \
	--output text)
	
	#get the public IP address
	gl_app_server_ip=$(aws ec2 describe-instances --instance-id $gl_app_server \
	--query Reservations[0].Instances[0].PublicIpAddress \
	--output text)

AWS Console	1. EC2 2. Instances 3. Launch an instance

Screenshot



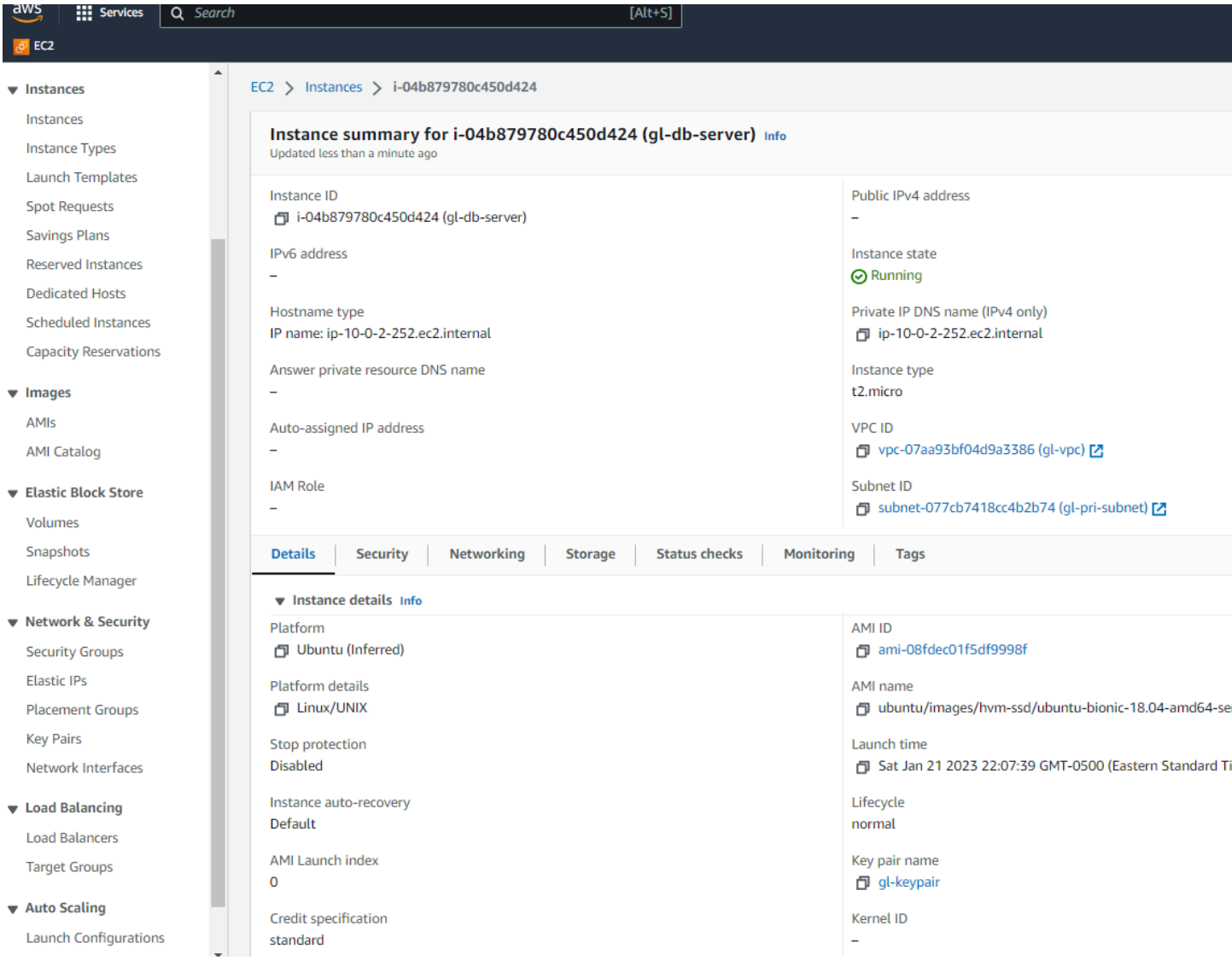
Step 10

Create an EC2 Ubuntu 18_04 instance (gl-db-server) to host My SQL server in the private subnet and associate the gl-pri-sg security group to it.

ami-08fdec01f5df9998f is used for 18_04 Ubuntu

AWS CLI (bash)	ubuntu_18_04="ami-08fdec01f5df9998f" gl_db_server=$(aws ec2 run-instances --image-id $ubuntu_18_04 --count 1 --instance-type t2.micro --key-name $gl_keypair \ --security-group-ids $gl_pri_sg \ --subnet-id $gl_pri_subnet \ --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=gl-db-server}]' \ --query Instances[0].InstanceId \ --output text)
AWS Console	1. EC2 2. Instances 3. Launch an instance

Screenshot



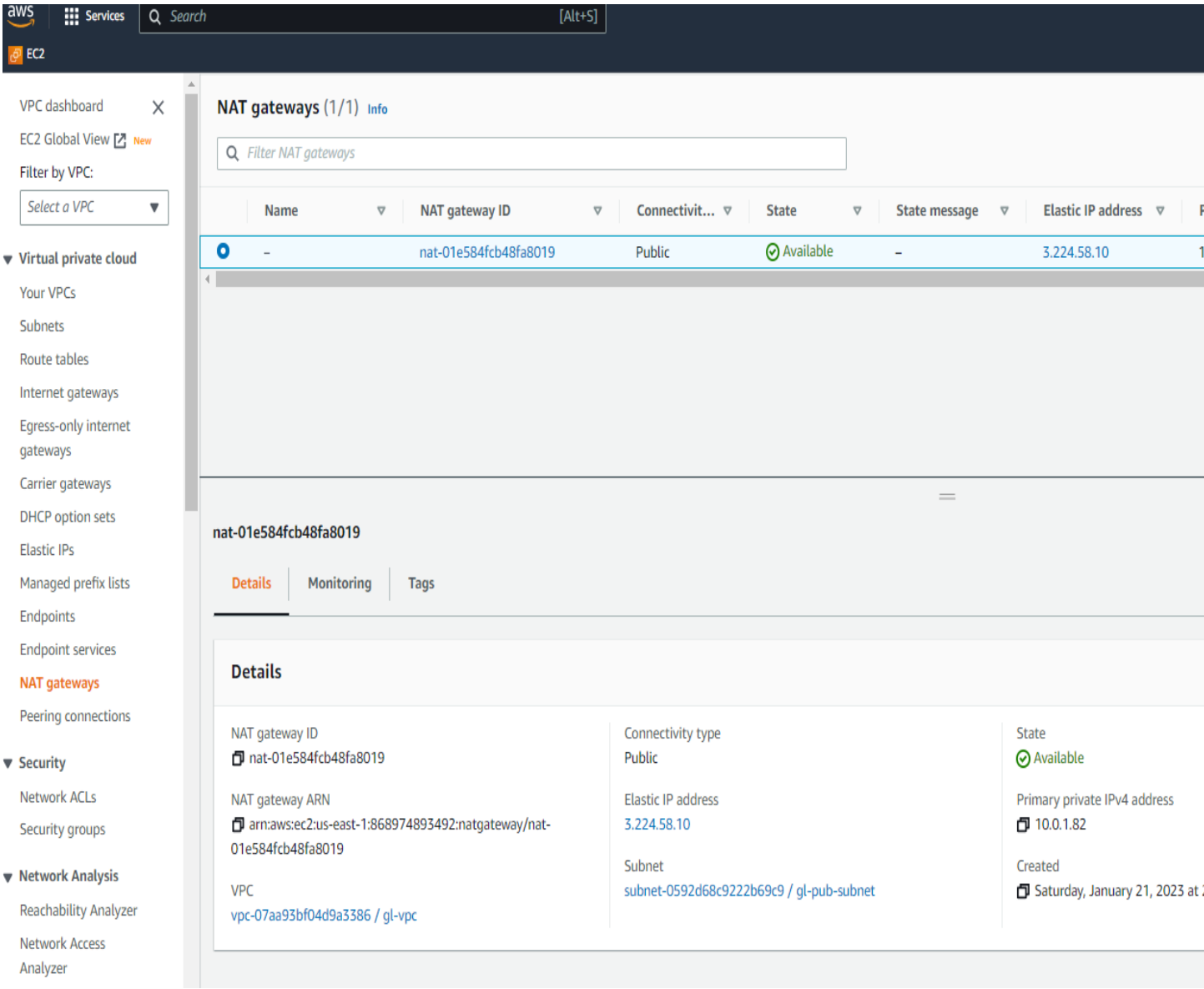
Step 11

Create a NAT gateway with a new elastic IP.	

AWS CLI (bash)	gl_nat_elastic_ip=$(aws ec2 allocate-address --query AllocationId --output text)
	
	#create NAT gateway on the public subnet with an Elastic IP
	gl_nat_gw=$(aws ec2 create-nat-gateway \
	--subnet-id $gl_pub_subnet \
	--allocation-id $gl_nat_elastic_ip \
	--query NatGateway.NatGatewayId \
	--output text)

AWS Console	1. EC2 2. Instances 3. Launch an instance

Screenshot



Step 12

Create a route table gl-rt-pri for private subnet.
Add a route to direct the internet traffic to NAT gateway

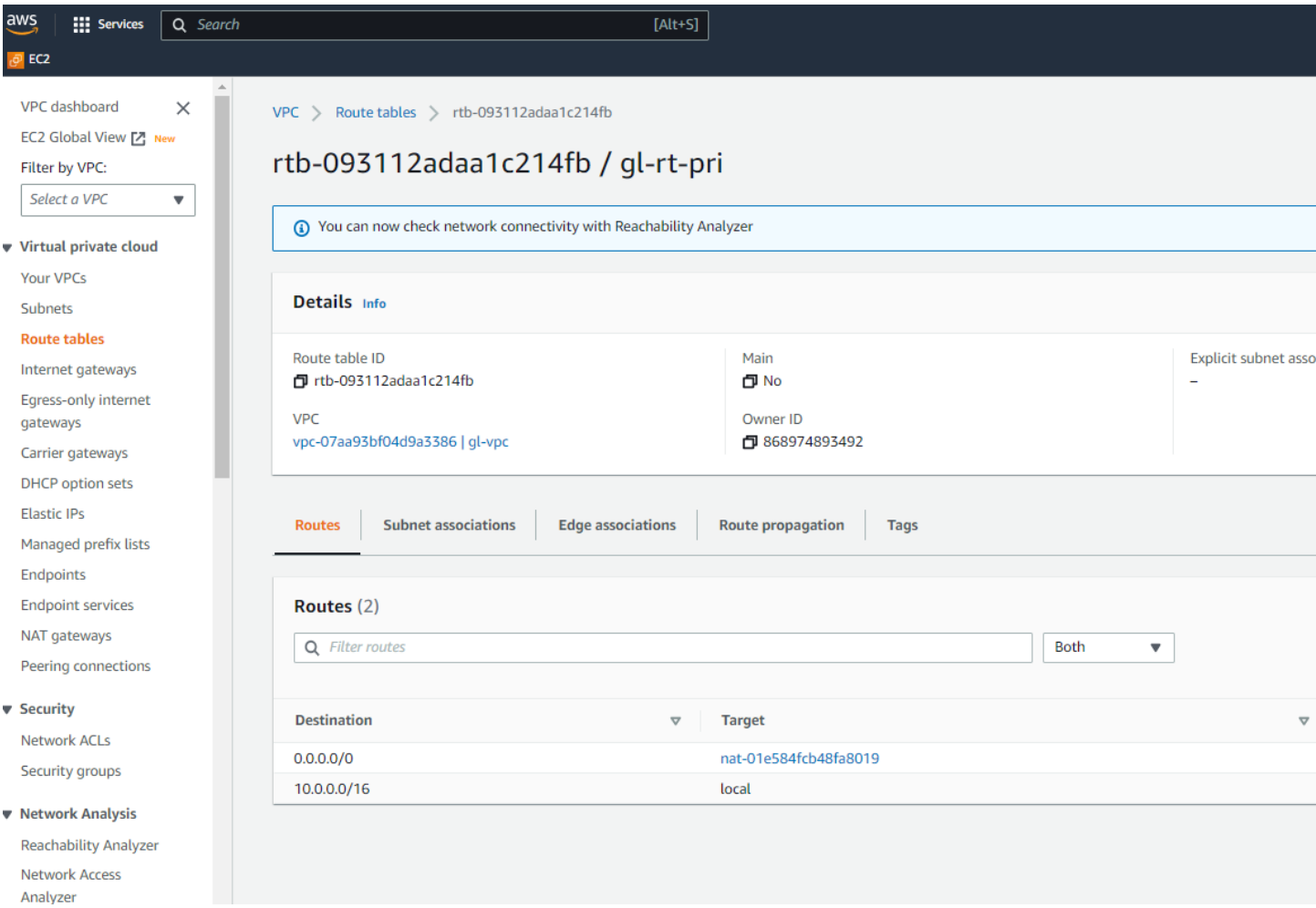
Associate this to private subnet	

AWS CLI (bash)	#create route table for private subnet
	gl_rt_pri=$(aws ec2 create-route-table --vpc-id $gl_vpc --query RouteTable.RouteTableId --output text \
	--tag-specifications "ResourceType=route-table,Tags=[{Key=Name,Value=gl-rt-pri}]")
	
	#create an entry to route the 0.0.0.0/0 traffic to the nat gateway
	aws ec2 create-route --route-table-id $gl_rt_pri --destination-cidr-block 0.0.0.0/0 --gateway-id $gl_nat_gw
	
	#assoicate
	aws ec2 associate-route-table --subnet-id $gl_pri_subnet --route-table-id $gl_rt_pri

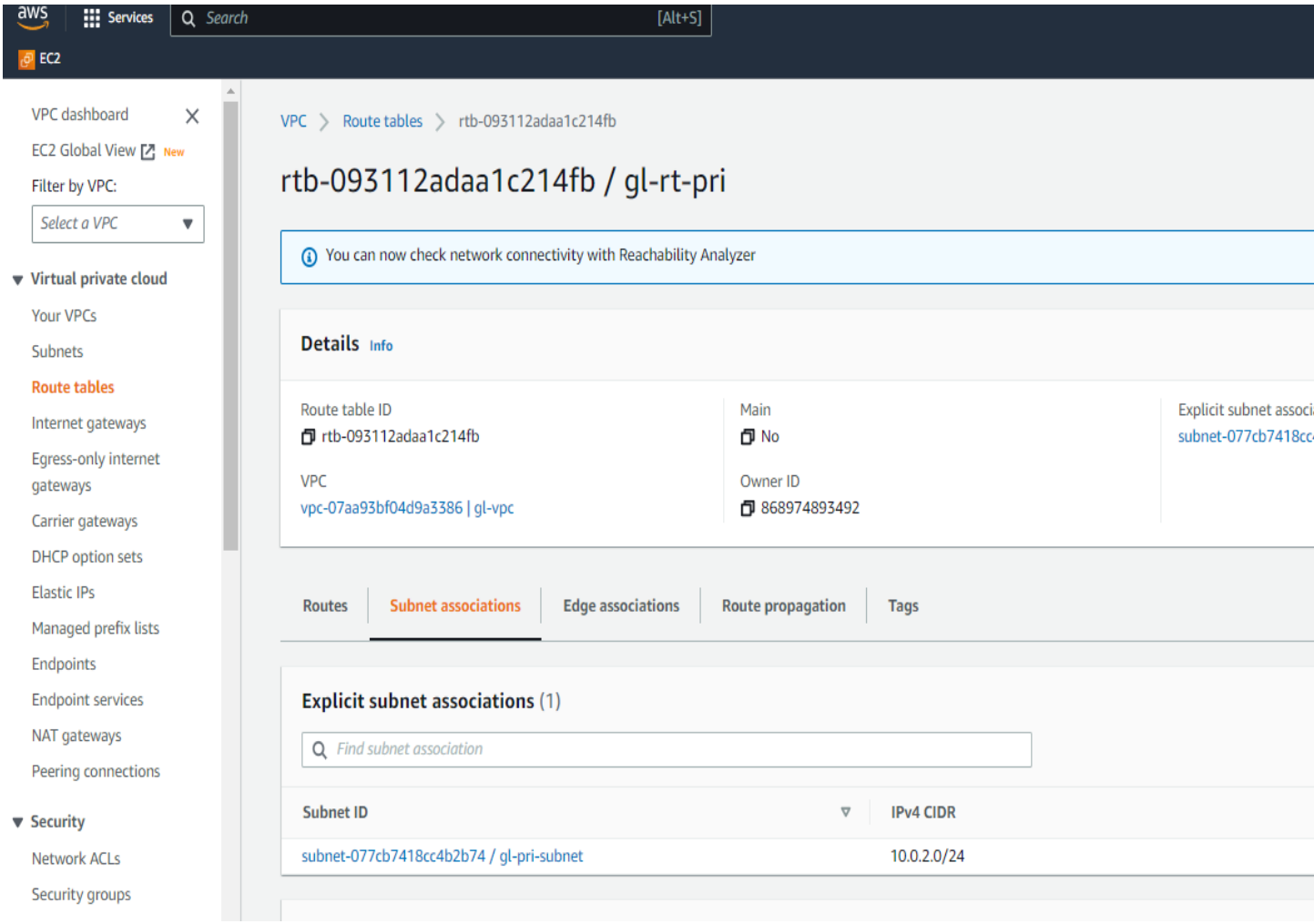
AWS Console	1. VPC 2. Route tables 3. Create route table

Screenshot

Route table with routes to direct traffic to NAT gateway



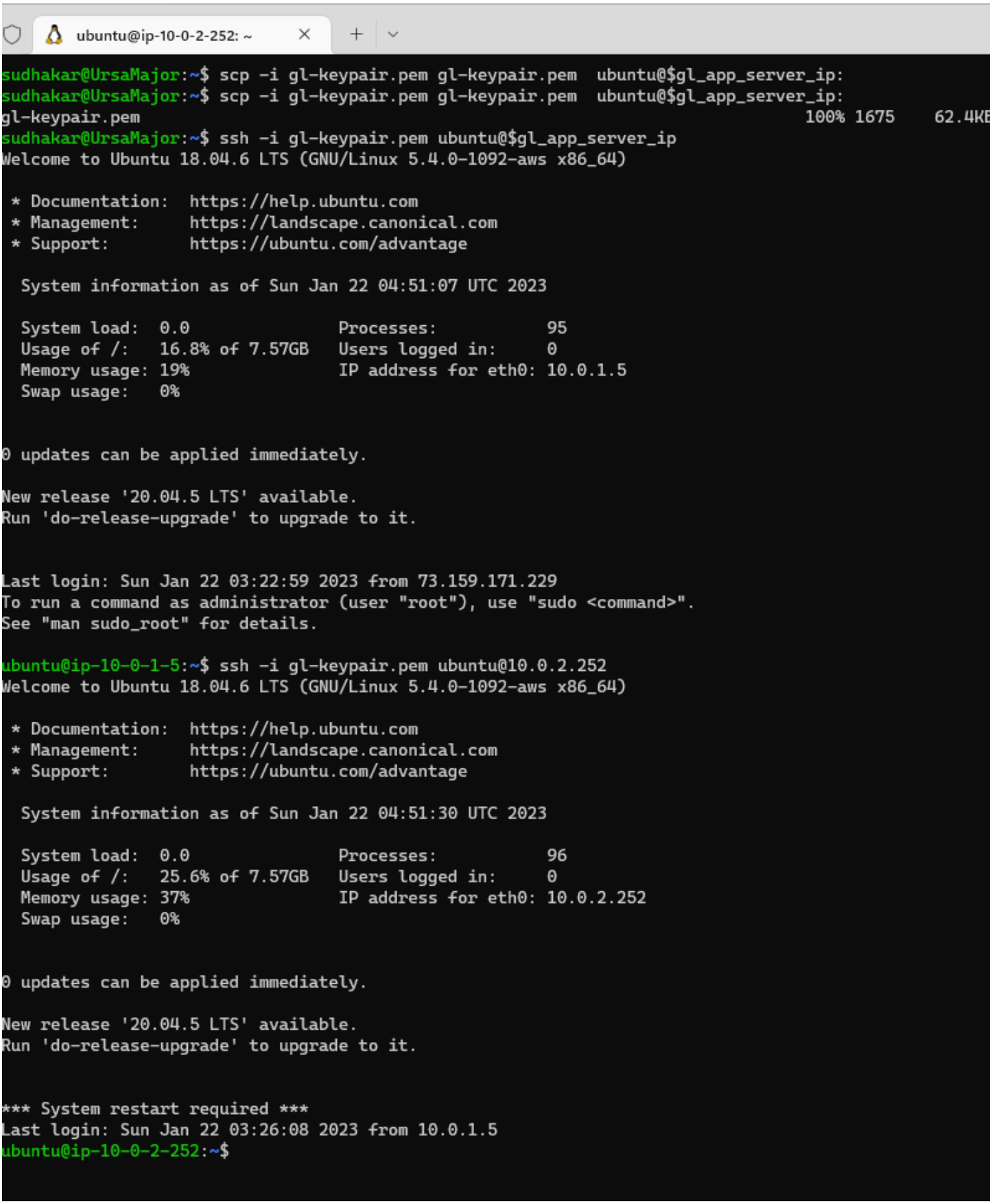
Route table with private subnet associated.



Step 13

Copy the keypair gl-keypair to app server EC2 instance in the public subnet. **SSH to app server from Local terminal.** **SSH to db server from app server**	
Commands	#copy the keypair file to app server scp -i ${gl_keypair}.pem ${gl_keypair}.pem ubuntu@$gl_app_server_ip:.ssh ssh -i gl-keypair.pem ubuntu@$gl_app_server_ip ssh -i gl-keypair.pem ubuntu@10.0.2.252

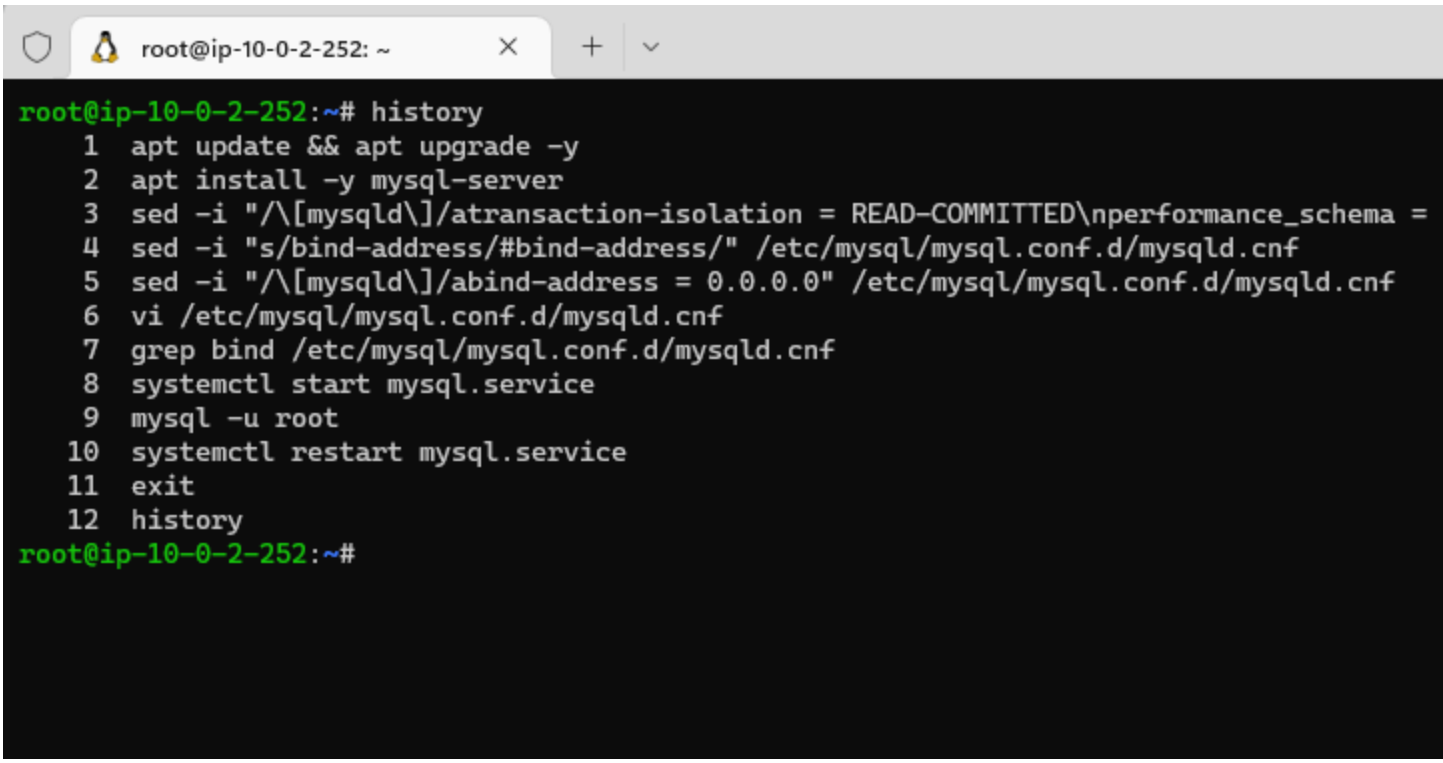
Screenshot for scp key , ssh to app server and DB server.


ubuntu@ip-10-0-2-252: ~
sudhakar@UrsaMajor:~$ scp -i gl-keypair.pem gl-keypair.pem ubuntu@$gl_app_server_ip:
sudhakar@UrsaMajor:~$ scp -i gl-keypair.pem gl-keypair.pem ubuntu@$gl_app_server_ip:
gl-keypair.pem 100% 1675 62.4KB
sudhakar@UrsaMajor:~$ ssh -i gl-keypair.pem ubuntu@$gl_app_server_ip
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1092-aws x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
System information as of Sun Jan 22 04:51:07 UTC 2023
System load: 0.0 Processes: 95
Usage of /: 16.8% of 7.57GB Users logged in: 0
Memory usage: 19% IP address for eth0: 10.0.1.5
Swap usage: 0%
0 updates can be applied immediately.
New release '20.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
Last login: Sun Jan 22 03:22:59 2023 from 73.159.171.229
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
ubuntu@ip-10-0-1-5:~$ ssh -i gl-keypair.pem ubuntu@10.0.2.252
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 5.4.0-1092-aws x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
System information as of Sun Jan 22 04:51:30 UTC 2023
System load: 0.0 Processes: 96
Usage of /: 25.6% of 7.57GB Users logged in: 0
Memory usage: 37% IP address for eth0: 10.0.2.252
Swap usage: 0%
0 updates can be applied immediately.
New release '20.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
*** System restart required ***
Last login: Sun Jan 22 03:26:08 2023 from 10.0.1.5
ubuntu@ip-10-0-2-252:~$

Step 14

Install My SQL Server on the gl-db-server based on the instructions from the following link. https://doc.owncloud.com/server/next/admin_manual/installation/quick_guides/ubuntu_22_04.html	
Commands	#!/usr/bin/env bash apt update && apt upgrade -y apt install -y mysql-server sed -i "/\[mysqld\]/atransaction-isolation = READ-COMMITTED\nperformance_schema = on" /etc/mysql/mysql.conf.d/mysqld.cnf sed -i "s/bind-address/#bind-address/" /etc/mysql/mysql.conf.d/mysqld.cnf sed -i "/\[mysqld\]/abind-address = 0.0.0.0" /etc/mysql/mysql.conf.d/mysqld.cnf systemctl start mysql.service mysql -u root -e "CREATE DATABASE IF NOT EXISTS owncloud; \ CREATE USER 'owncloud'@'localhost' IDENTIFIED BY 'pi=3.14159'; \ GRANT ALL PRIVILEGES ON *.* TO 'owncloud'@'localhost' WITH GRANT OPTION; \ FLUSH PRIVILEGES; \ CREATE USER 'owncloud'@'%' IDENTIFIED BY 'pi=3.14159'; \ GRANT ALL PRIVILEGES ON *.* TO 'owncloud'@'%' WITH GRANT OPTION; \ FLUSH PRIVILEGES; "

Screenshot for MySQL server installation



Step 15

Install Apache2, PHP 7.4 , Redis , Own Cloud on gl-appserver EC2 instance based on the instructions from the following link. https://doc.owncloud.com/server/next/admin_manual/installation/quick_guides/ubuntu_22_04.html	
Commands to install Prerequisite components	#!/usr/bin/env bash apt update && apt upgrade -y add-apt-repository ppa:ondrej/php -y apt update && apt upgrade -y apt install -y \ apache2 \ libapache2-mod-php7.4 \ openssl redis-server wget \ php7.4 php7.4-imagick php7.4-common php7.4-curl \ php7.4-gd php7.4-imap php7.4-intl php7.4-json \ php7.4-mbstring php7.4-gmp php7.4-bcmath php7.4-mysql \ php7.4-ssh2 php7.4-xml php7.4-zip php7.4-apcu \ php7.4-redis php7.4-ldap php-phpseclib apt-get install -y php7.4-smbclient

	echo "extension=smbclient.so" > /etc/php/7.4/mods-available/smbclient.ini
	phpenmod smbclient
	systemctl restart apache2
	
	apt install -y \
	  unzip bzip2 rsync curl jq \
	  inetutils-ping ldap-utils\
	  smbclient
	
	#install mysql-client
	apt-get install -y mysql-client

Commands to configure Apache	#!/usr/bin/env bash
	
	FILE="/etc/apache2/sites-available/owncloud.conf"
	cat <<EOM >$FILE
	<VirtualHost *:80>
	# uncommment the line below if variable was set
	#ServerName $my_domain
	DirectoryIndex index.php index.html
	DocumentRoot /var/www/owncloud
	<Directory /var/www/owncloud>
	  Options +FollowSymlinks -Indexes
	  AllowOverride All
	  Require all granted
	
	 <IfModule mod_dav.c>
	  Dav off
	 </IfModule>
	
	 SetEnv HOME /var/www/owncloud
	 SetEnv HTTP_HOME /var/www/owncloud
	</Directory>
	</VirtualHost>
	EOM
	
	a2dissite 000-default
	a2ensite owncloud.conf
	echo "Enabled $FILE in Apache"
	a2enmod dir env headers mime rewrite setenvif
	systemctl restart apache2
	echo "apache2 has been restarted"

Commands to install Own Cloud	#!/usr/bin/env bash #run this script as follows #dbserver_ip=xxxxx install-owncloud.bash if [$dbserver_ip = ""] ; then echo "exiting now. dbserver_ip is not defined"; exit; fi sec_admin_pwd="pi=3.14159" echo $sec_admin_pwd > /etc/.sec_admin_pwd.txt sec_db_pwd="pi=3.14159" echo $sec_db_pwd > /etc/.sec_db_pwd.txt FILE="/usr/local/bin/occ" cat <<EOM >$FILE #! /bin/bash cd /var/www/owncloud sudo -E -u www-data /usr/bin/php /var/www/owncloud/occ "\$@" EOM chmod +x $FILE echo "Created the helper file: $FILE" cd /var/www/ wget https://download.owncloud.com/server/stable/owncloud-complete-latest.tar.bz2 && \ tar -xjf owncloud-complete-latest.tar.bz2 && \ chown -R www-data. owncloud echo "Downloaded the OwnCloud repo" occ maintenance:install \ --database "mysql" \ --database-host ${dbserver_ip} \ --database-name "owncloud" \ --database-user "owncloud" \ --database-pass ${sec_db_pwd} \ --data-dir "/var/www/owncloud/data" \ --admin-user "admin" \ --admin-pass ${sec_admin_pwd} app_server_ip=$(curl -s http://169.254.169.254/latest/meta-data/public-ipv4)

```
local_ip=$(curl -s http://169.254.169.254/latest/meta-data/local-ipv4)
client_ip=$(echo $SSH_CLIENT | awk '{ print $1}')

occ config:system:set trusted_domains 1 --value="$client_ip"
occ config:system:set trusted_domains 2 --value="$app_server_ip"
occ config:system:set trusted_domains 3 --value="$local_ip"

occ config:system:set files_external_allow_create_new_local --value 'true'

occ background:cron
echo "*/15  *  *  *  * /var/www/owncloud/occ system:cron" \
  | sudo -u www-data -g crontab tee -a \
  /var/spool/cron/crontabs/www-data

echo "0  2  *  *  * /var/www/owncloud/occ dav:cleanup-chunks" \
  | sudo -u www-data -g crontab tee -a \
  /var/spool/cron/crontabs/www-data

echo "Configuring Memcache\APCu"

occ config:system:set \
   memcache.local \
   --value '\OC\Memcache\APCu'

   echo "Configuring Memcache\Redis"
occ config:system:set \
   memcache.locking \
   --value '\OC\Memcache\Redis'
occ config:system:set \
   redis \
   --value '{"host": "127.0.0.1", "port": "6379"}' \
   --type json

FILE="/etc/logrotate.d/owncloud"
cat <<EOM >$FILE
/var/www/owncloud/data/owncloud.log {
  size 10M
  rotate 12
  copytruncate
  missingok
  compress
  compresscmd /bin/gzip
}
EOM

cd /var/www/
chown -R www-data. owncloud
```

	systemctl restart apache2 echo "apache2 has been restarted"

Screenshot for Apache2, PHP 7.4, Redis, Own Cloud Installation

root@ip-10-0-1-5: /var/www

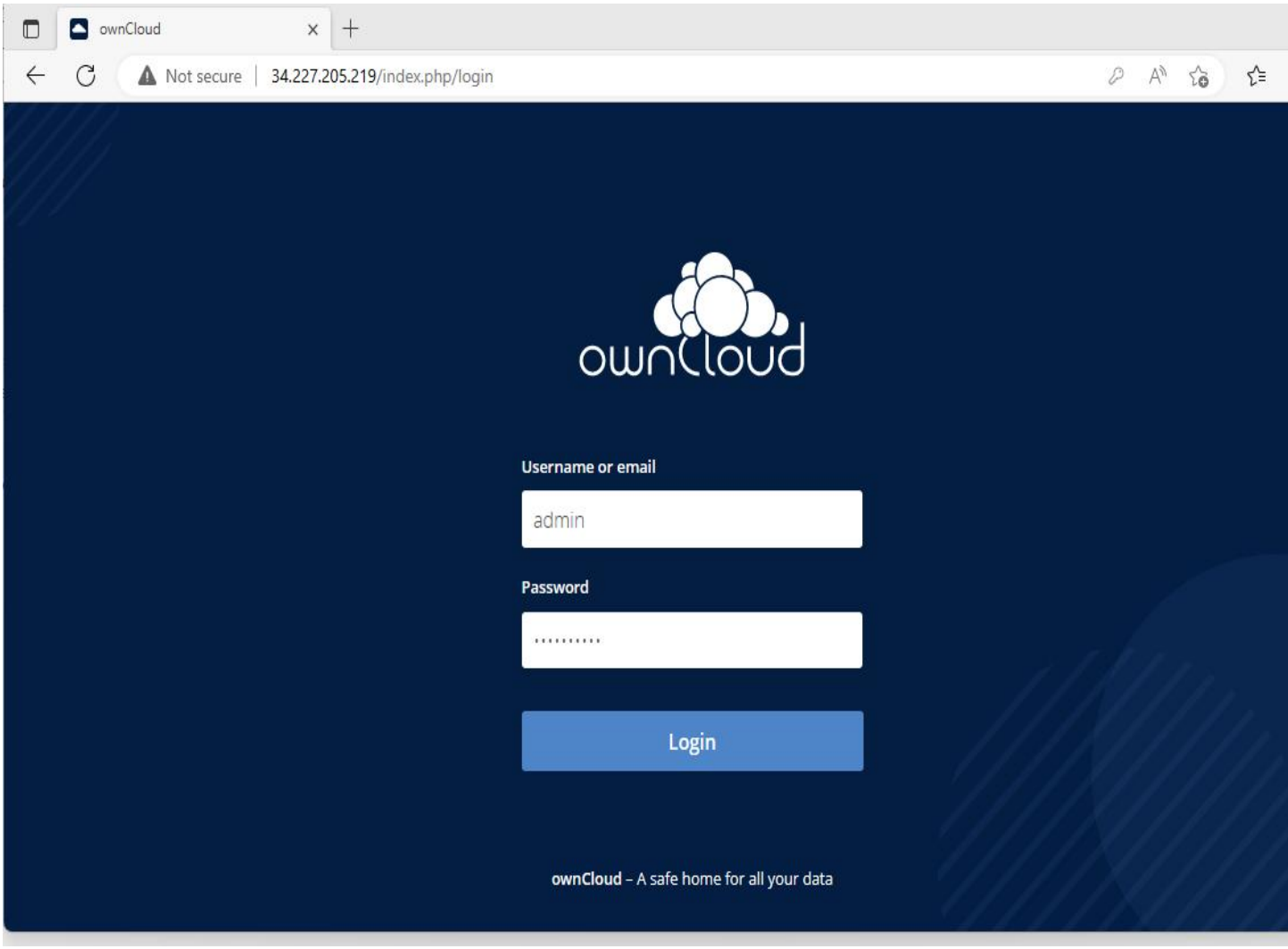
```
   65  history
   66  history 30
root@ip-10-0-1-5:/var/www# history
    1  ll
    2  ll .ssh/
    3  ssh -i .ssh/gl-keypair.pem ubuntu@10.0.2.252
    4  exit
    5  apt update && apt upgrade -y
    6  add-apt-repository ppa:ondrej/php -y
    7  apt update && apt upgrade -y
    8  apt install -y   apache2   libapache2-mod-php7.4   openssl redis-server wget   php7.4 php7.
  php7.4-gd php7.4-imap php7.4-intl php7.4-json   php7.4-mbstring php7.4-gmp php7.4-bcmath php7.4-
4-zip php7.4-apcu   php7.4-redis php7.4-ldap php-phpseclib
    9  apt-get install -y php7.4-smbclient
   10  echo "extension=smbclient.so" > /etc/php/7.4/mods-available/smbclient.ini
   11  phpenmod smbclient
   12  systemctl restart apache2
   13  php -m |grep smb
   14  systemctl restart apache2
   15  apt install -y   unzip bzip2 rsync curl jq   inetutils-ping  ldap-utils  smbclient
   16  apt-get install -y mysql-client
   17  FILE="/etc/apache2/sites-available/owncloud.conf"
   18  cat <<EOM >$FILE
<VirtualHost *:80>
# uncommment the line below if variable was set
#ServerName $my_domain
DirectoryIndex index.php index.html
DocumentRoot /var/www/owncloud
<Directory /var/www/owncloud>
  Options +FollowSymlinks -Indexes
  AllowOverride All
  Require all granted

 <IfModule mod_dav.c>
  Dav off
 </IfModule>

 SetEnv HOME /var/www/owncloud
 SetEnv HTTP_HOME /var/www/owncloud
</Directory>
</VirtualHost>
EOM

   19  a2dissite 000-default
   20  a2ensite owncloud.conf
   21  a2enmod dir env headers mime rewrite setenvif
   22  systemctl restart apache2
   23  mysql
   24  mysql -u owncloud -h 10.0.2.252 -p
   25  history
   26  export dbserver_ip=10.0.2.252
   27  env|grep -i  dbser
   28  sec_admin_pwd="pi=3.14159"
   29  echo $sec_admin_pwd > /etc/.sec_admin_pwd.txt
   30  sec_db_pwd="pi=3.14159"
   31  echo $sec_db_pwd > /etc/.sec_db_pwd.txt
   32  FILE="/usr/local/bin/occ"
```

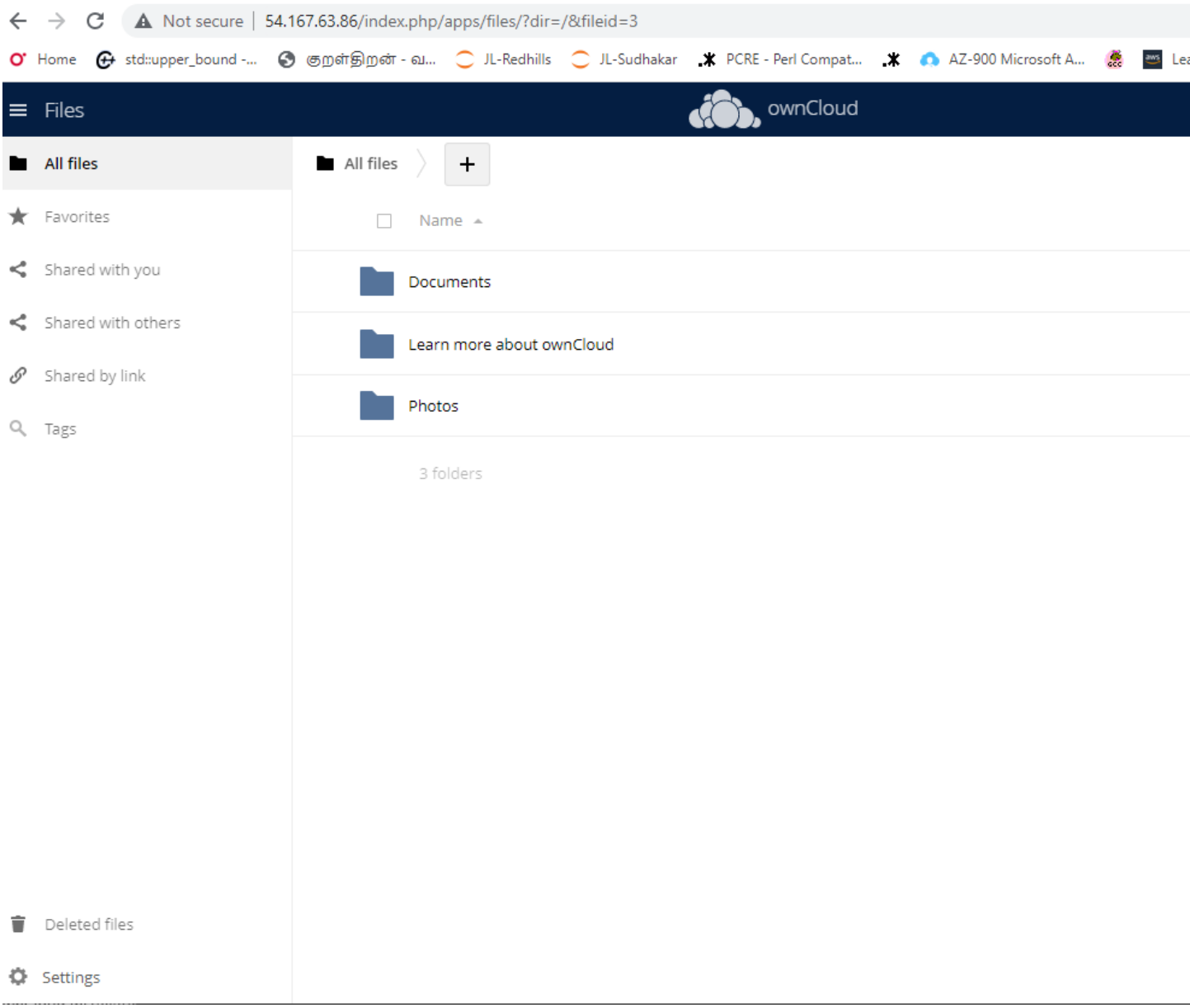
Screenshot2 for Apache2, PHP 7.4, Redis, Own Cloud Installation


root@ip-10-0-1-5: /var/www
30 sec_db_pwd="pi=3.14159"
31 echo $sec_db_pwd > /etc/.sec_db_pwd.txt
32 FILE="/usr/local/bin/occ"
33 cat <<EOM >$FILE
#! /bin/bash
cd /var/www/owncloud
sudo -E -u www-data /usr/bin/php /var/www/owncloud/occ "\$@"
EOM
34 chmod +x $FILE
35 cd /var/www/
36 wget https://download.owncloud.com/server/stable/owncloud-complete-latest.tar.bz2 && tar -xj
&& chown -R www-data. owncloud
37 occ -h
38 env|grep -i sec_a
39 history
40 echo $sec_admin_pwd
41 app_server_ip=$(curl -s http://169.254.169.254/latest/meta-data/public-ipv4)
42 local_ip=$(curl -s http://169.254.169.254/latest/meta-data/local-ipv4)
43 client_ip=$(echo $SSH_CLIENT | awk '{ print $1}')
44 occ config:system:set trusted_domains 1 --value=""
45 occ config:system:set trusted_domains 2 --value="$client_ip"
46 echo $client_ip
47 echo $local_ip
48 echo $app_server_ip
49 occ config:system:set trusted_domains 2 --value="$app_server_ip"
50 occ config:system:set trusted_domains 3 --value="$local_ip"
51 occ config:system:set files_external_allow_create_new_local --value 'true'
52 occ background:cron
53 echo "*/15 * * * * /var/www/owncloud/occ system:cron" | sudo -u www-data -g crontab te
-data
54 echo "0 2 * * * /var/www/owncloud/occ dav:cleanup-chunks" | sudo -u www-data -g cronta
/www-data
55 occ config:system:set memcache.local --value '\OC\Memcache\APCu'
56 occ config:system:set memcache.locking --value '\OC\Memcache\Redis'
57 occ config:system:set redis --value '{"host": "127.0.0.1", "port": "6379"}' --type
58 FILE="/etc/logrotate.d/owncloud"
59 cat <<EOM >$FILE
/var/www/owncloud/data/owncloud.log {
size 10M
rotate 12
copytruncate
missingok
compress
compresscmd /bin/gzip
}
EOM
60 cd /var/www/
61 chown -R www-data. owncloud
62 systemctl restart apache2
63 history
64 history -h
65 history
66 history 30
67 history
root@ip-10-0-1-5:/var/www#

STEP 16 Login to OwnCloud (user = admin)

Screenshot





STEP 17: Install S3 External storage app in Own Cloud

After logging in as admin, navigate to Market/Storage/External Storage S3.

Install the external storage S3 plugin.

Screenshot

Not secure | 54.167.63.86/index.php/apps/market/#/app/files_external_s3

Home | std::upper_bound -... | குறள்திறன் - வ... | JL-Redhills | JL-Sudhakar | PCRE - Perl Compat... | AZ-900 Microsoft A... | Learner Lab | How To Install Linux... | Other boo

☰ Market ownCloud ad

Market

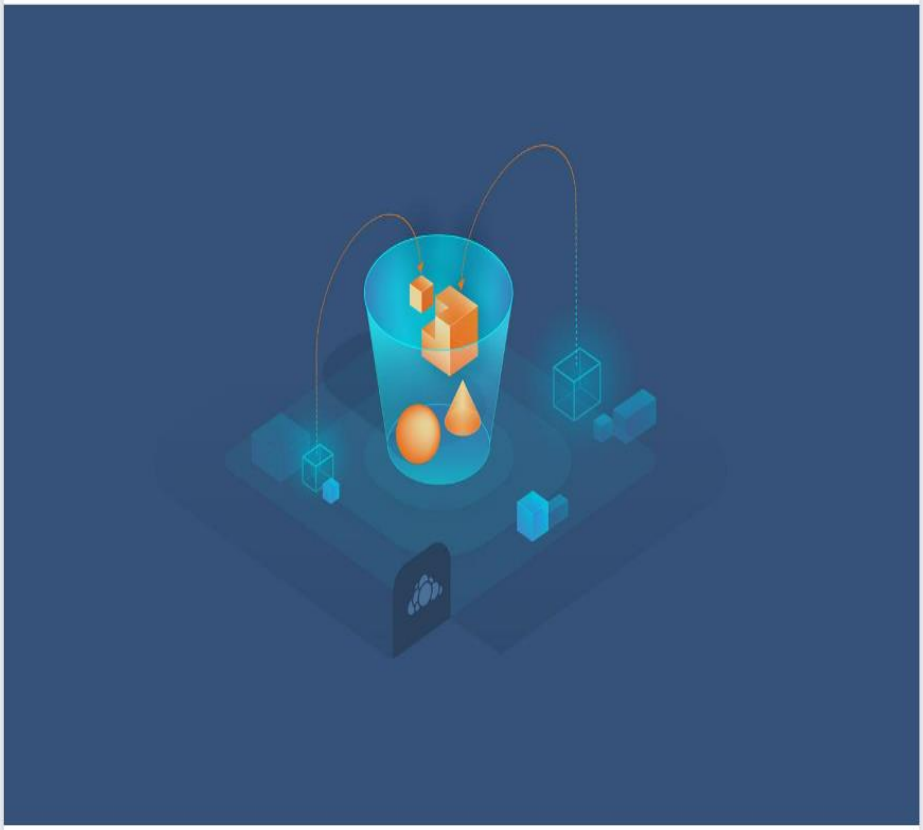
Show all
App Bundles

CATEGORIES
Automation
Collaboration
Customization
External plugins
Games
Integration
Multimedia
Productivity
Security
Storage
Tools

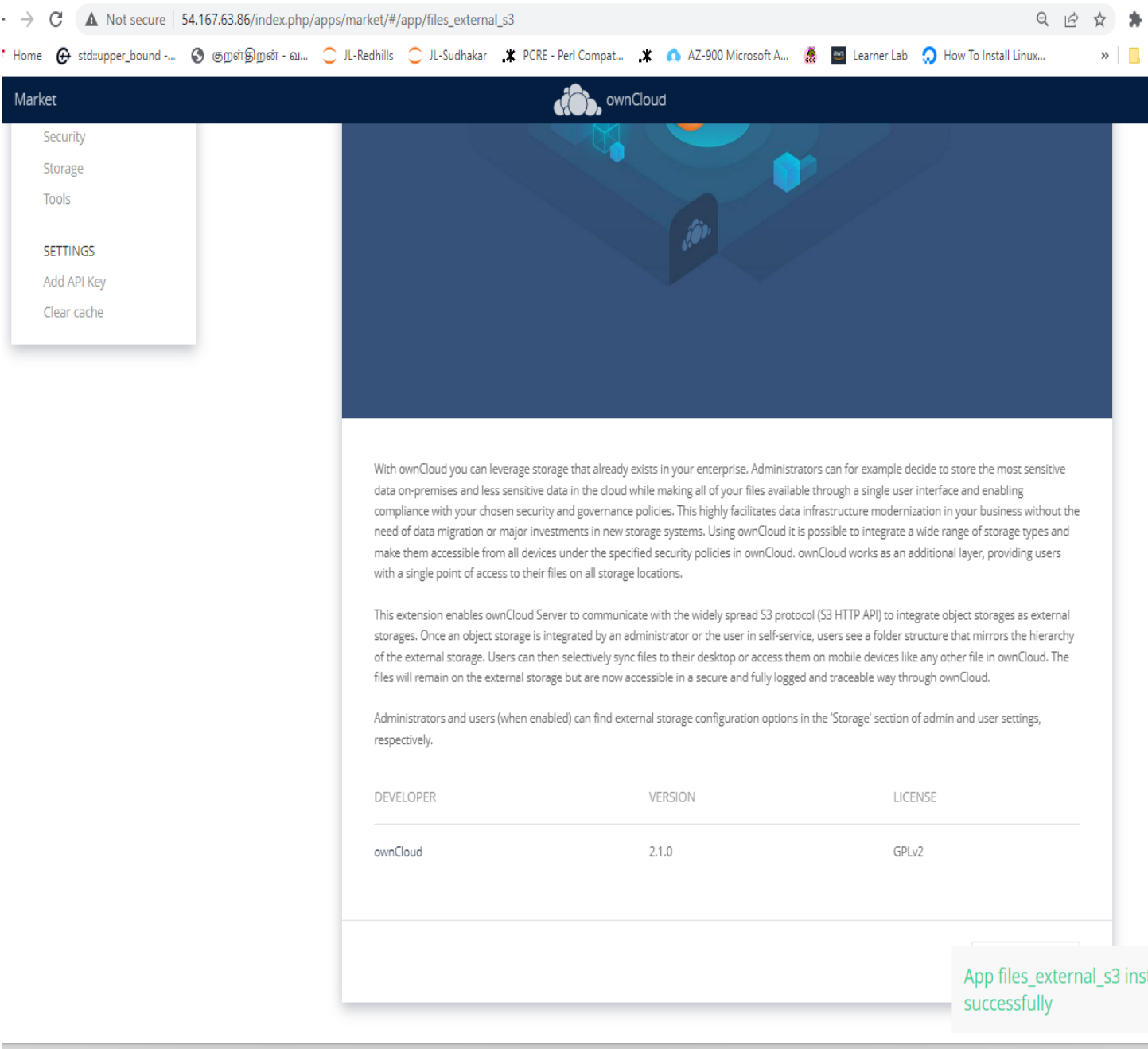
SETTINGS
Add API Key
Clear cache

External Storage: S3

storage

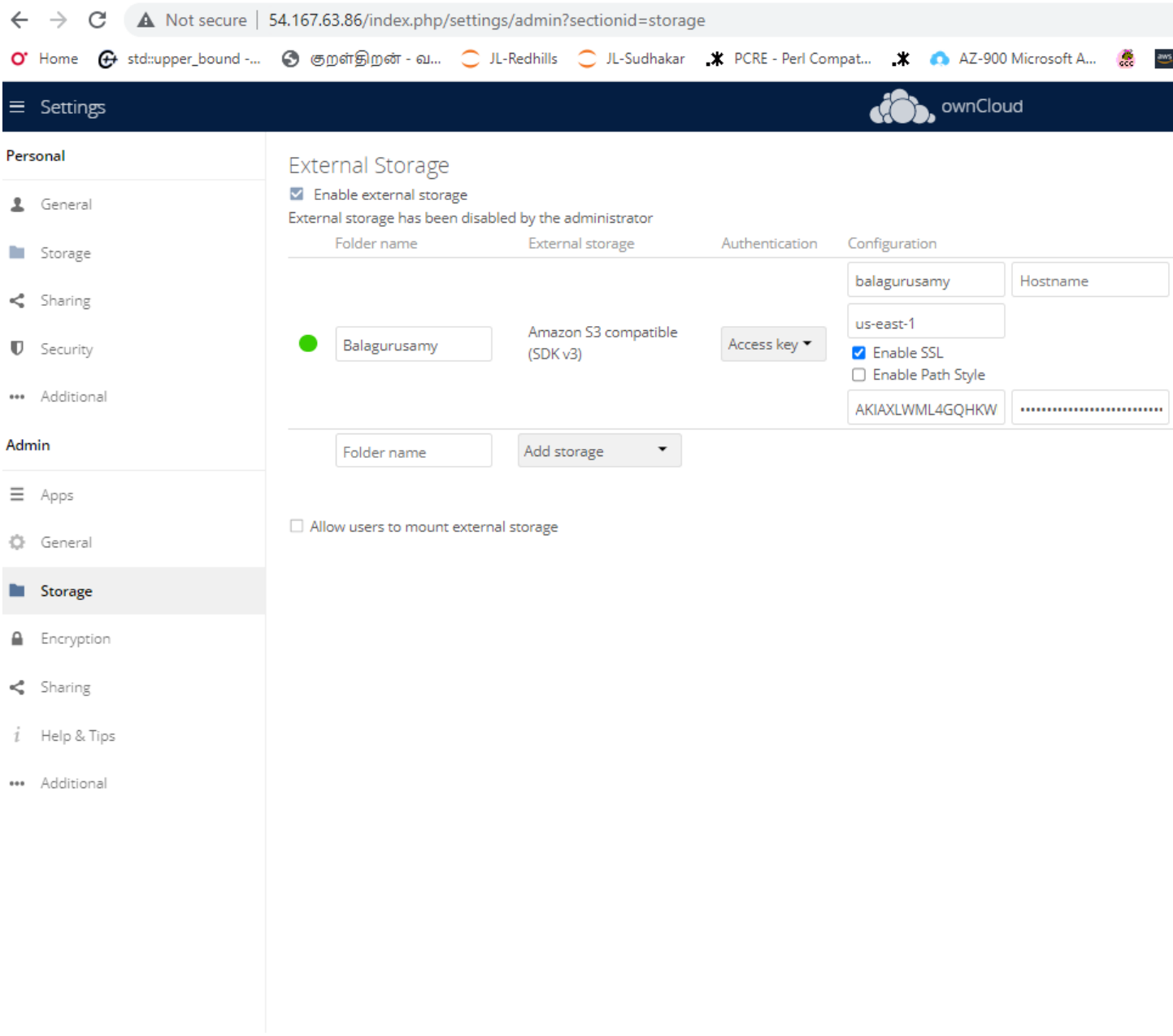


With ownCloud you can leverage storage that already exists in your enterprise. Administrators can for example decide to store the most sensitive data on-premises and less sensitive data in the cloud while making all of your files available through a single user interface and enabling compliance with your chosen security and governance policies. This highly facilitates data infrastructure modernization in your business without the need of data migration or major investments in new storage systems. Using ownCloud it is possible to integrate a wide range of storage types and



Step 18: Add S3 buckets to External Storage using and aws_access_key_id and aws_secret_access_key

Screenshot

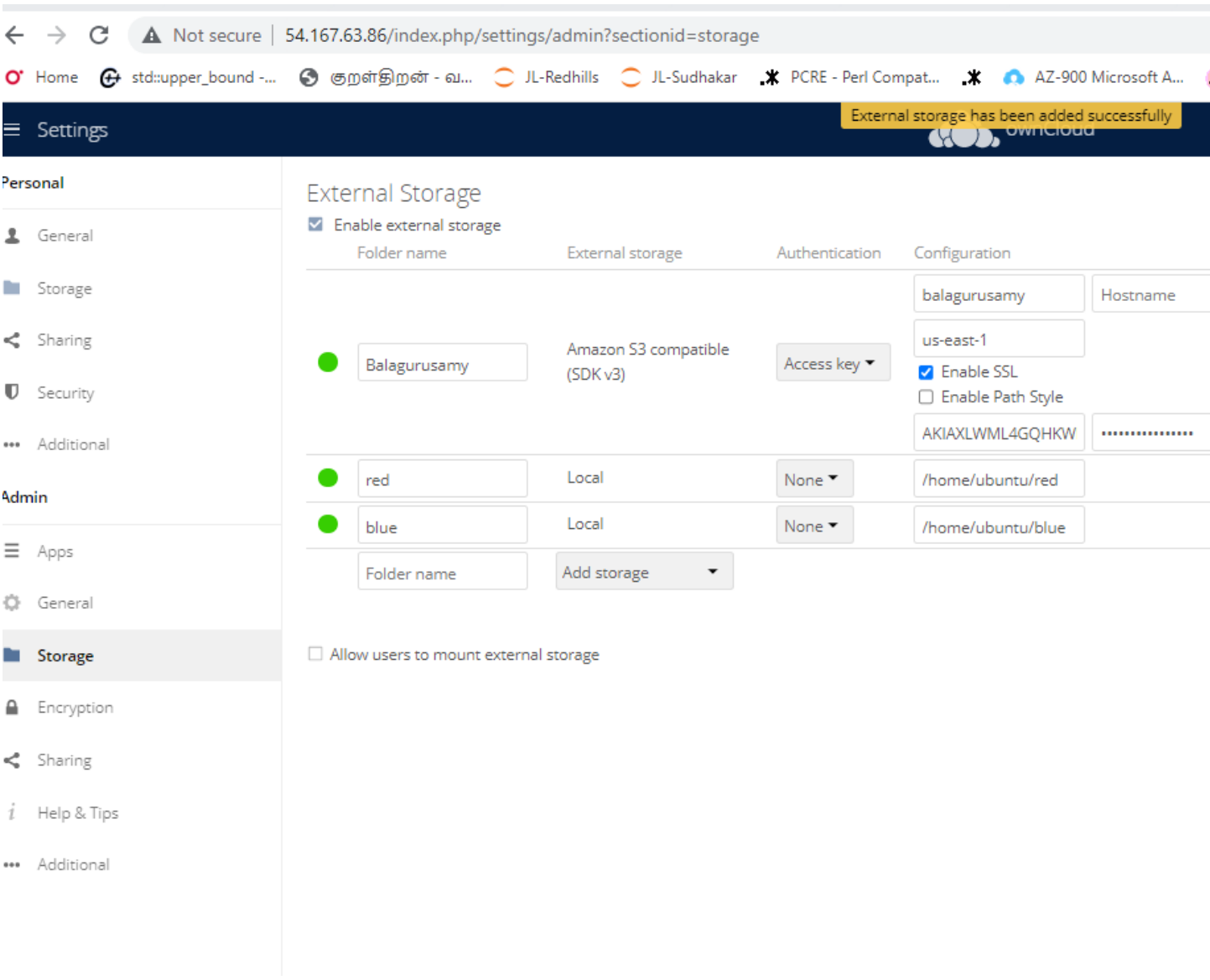


STEP 19: Create users & groups and grant access to S3 bucket.

Users; bluehills -Group blue

Redhills- Group red

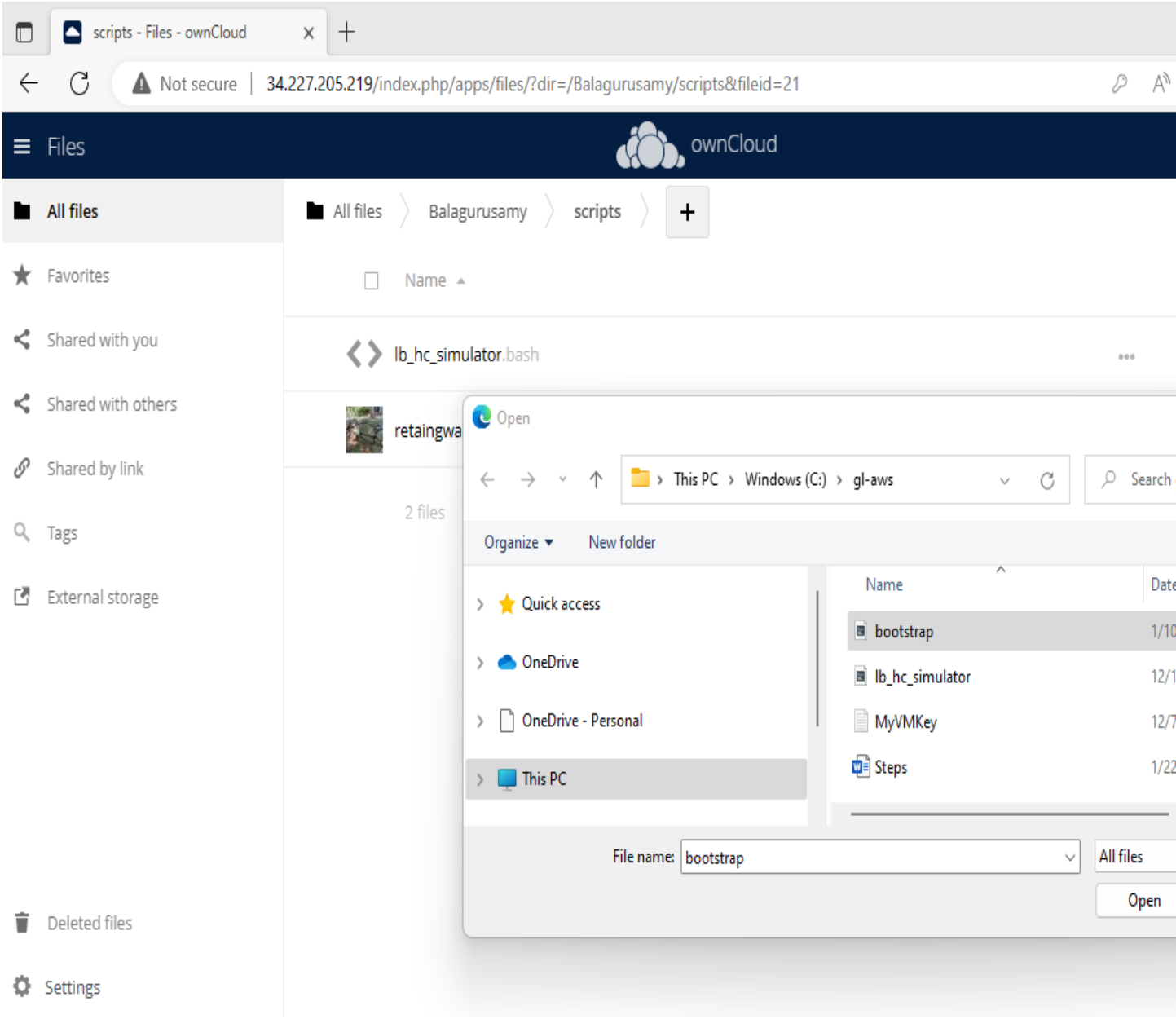
Screenshot



STEP 20: login in using user account : bluehills and upload some files to S3 bucket

Navigate to All Files > S3 Bucket (Balagurusamy) and click + button to upload.

Screenshot

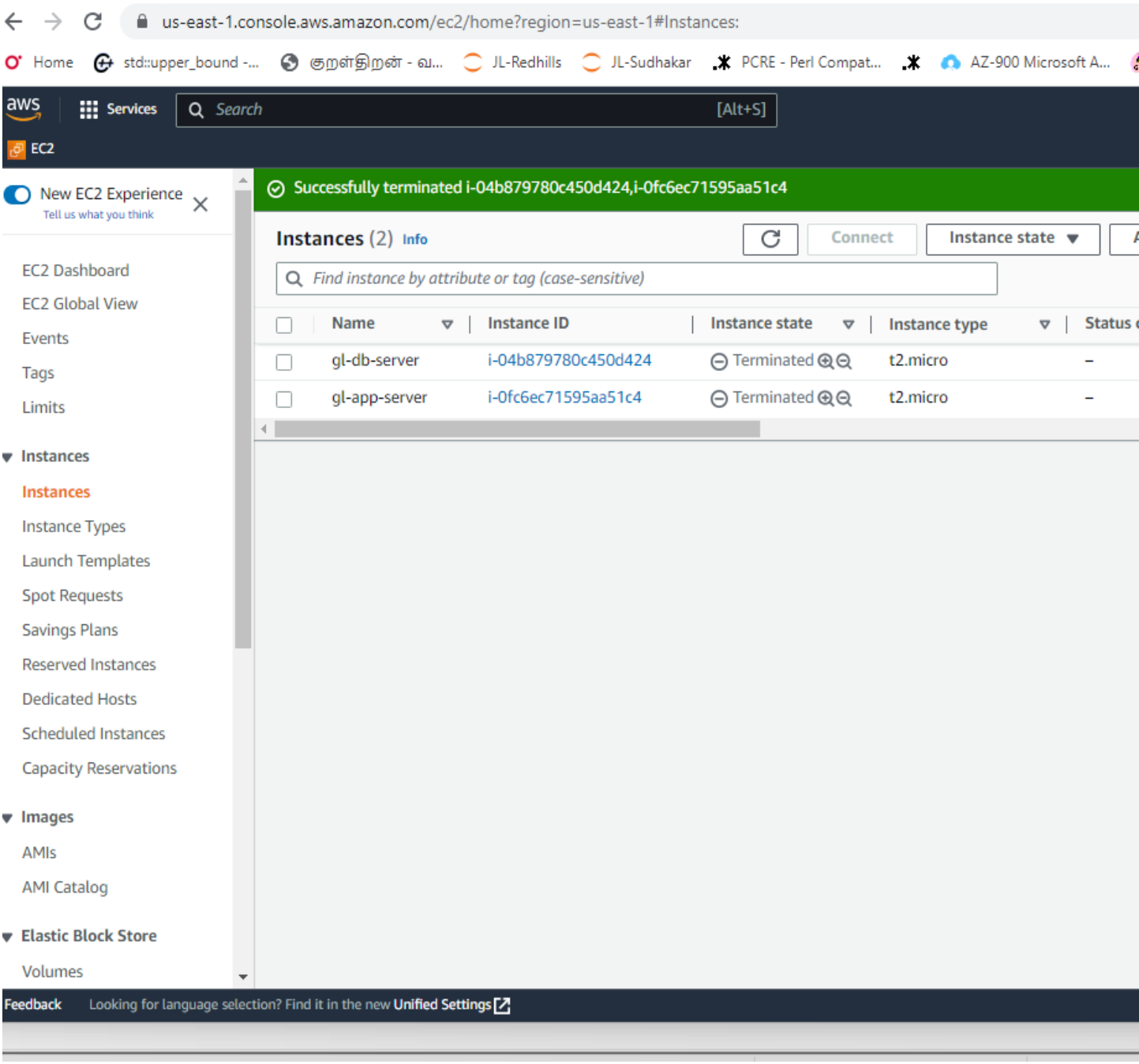


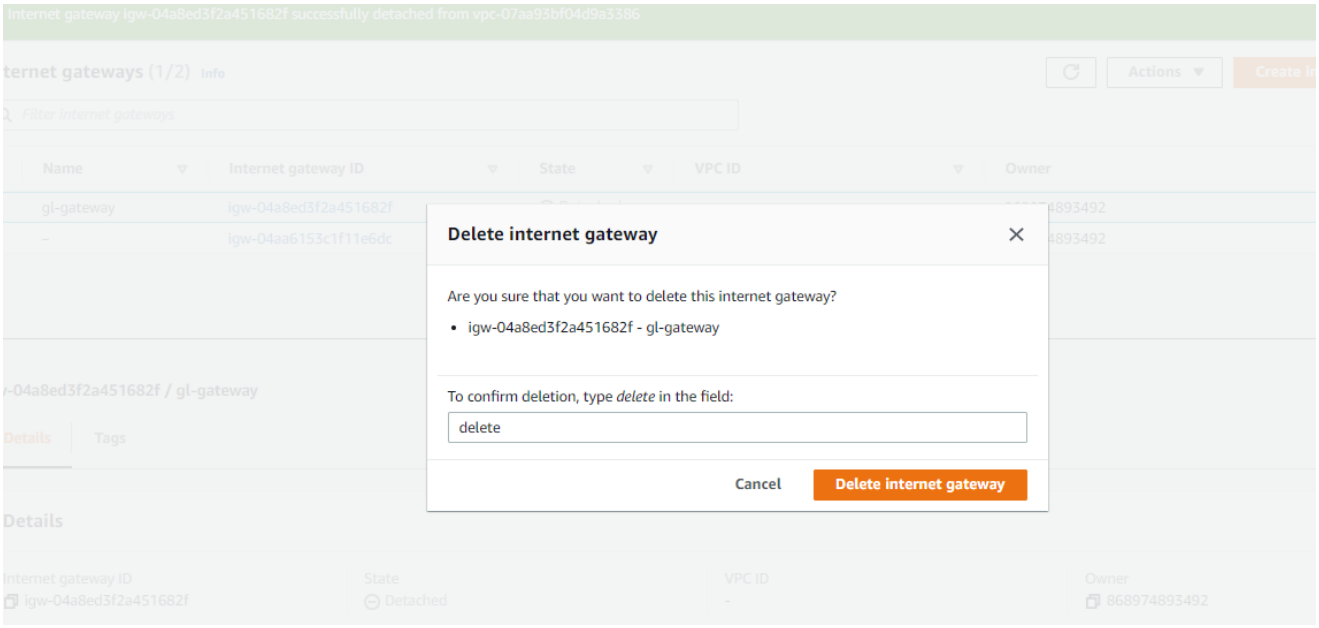
STEP 21: Cleanup

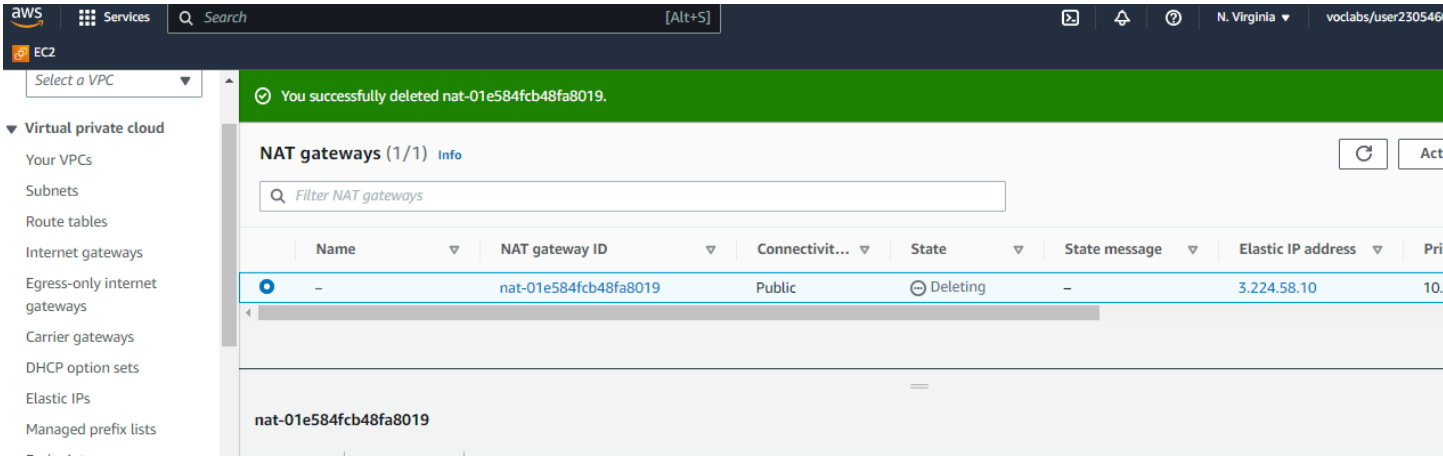
Create a key pair gl-keypair

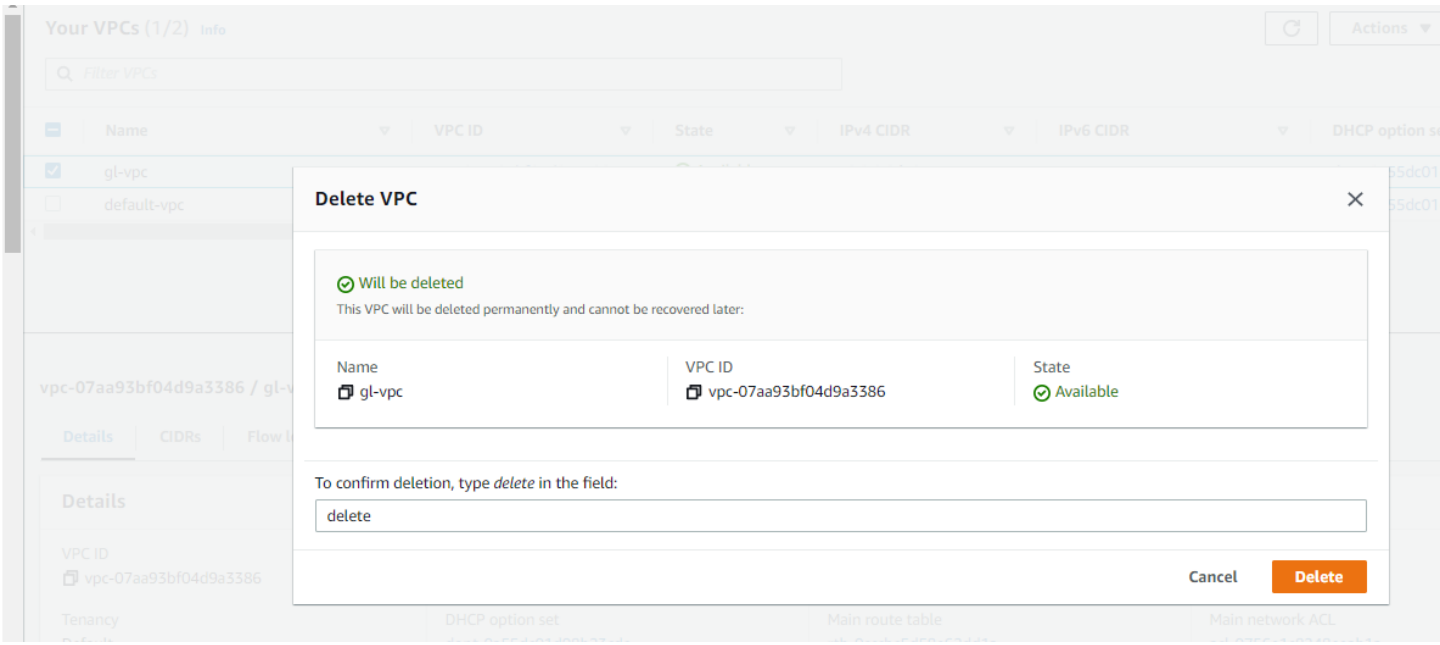
AWS CLI (bash)	#terminate the instance both app & db servers aws ec2 terminate-instances --instance-ids $gl_app_server aws ec2 terminate-instances --instance-ids $gl_db_server #delete NAT gateway aws ec2 delete-nat-gateway --nat-gateway-id $gl_nat_gw #delete security groups aws ec2 delete-security-group --group-id $gl_pub_sg aws ec2 delete-security-group --group-id $gl_pri_sg #delete subnets aws ec2 delete-subnet --subnet-id $gl_pub_subnet aws ec2 delete-subnet --subnet-id $gl_pri_subnet #delete EC2 route tables aws ec2 delete-route-table --route-table-id $gl_rt_gateway aws ec2 delete-route-table --route-table-id $gl_rt_pri #detach internet-gateway aws ec2 detach-internet-gateway --internet-gateway-id $gl_gateway --vpc-id $gl_vpc #delete internet-gateway aws ec2 delete-internet-gateway --internet-gateway-id $gl_gateway #delete vpc aws ec2 delete-vpc --vpc-id $gl_vpc #release address aws ec2 release-address --allocation-id $gl_nat_elastic_ip
AWS Console	1. Terminate Instances 2. Delete NAT GATEWAy 3. Delete security groups 4. Delete Subnets 5. Delete route tables 6. Detach and delete Internet gateways 7. Delete vpc 8. Release elastic IP

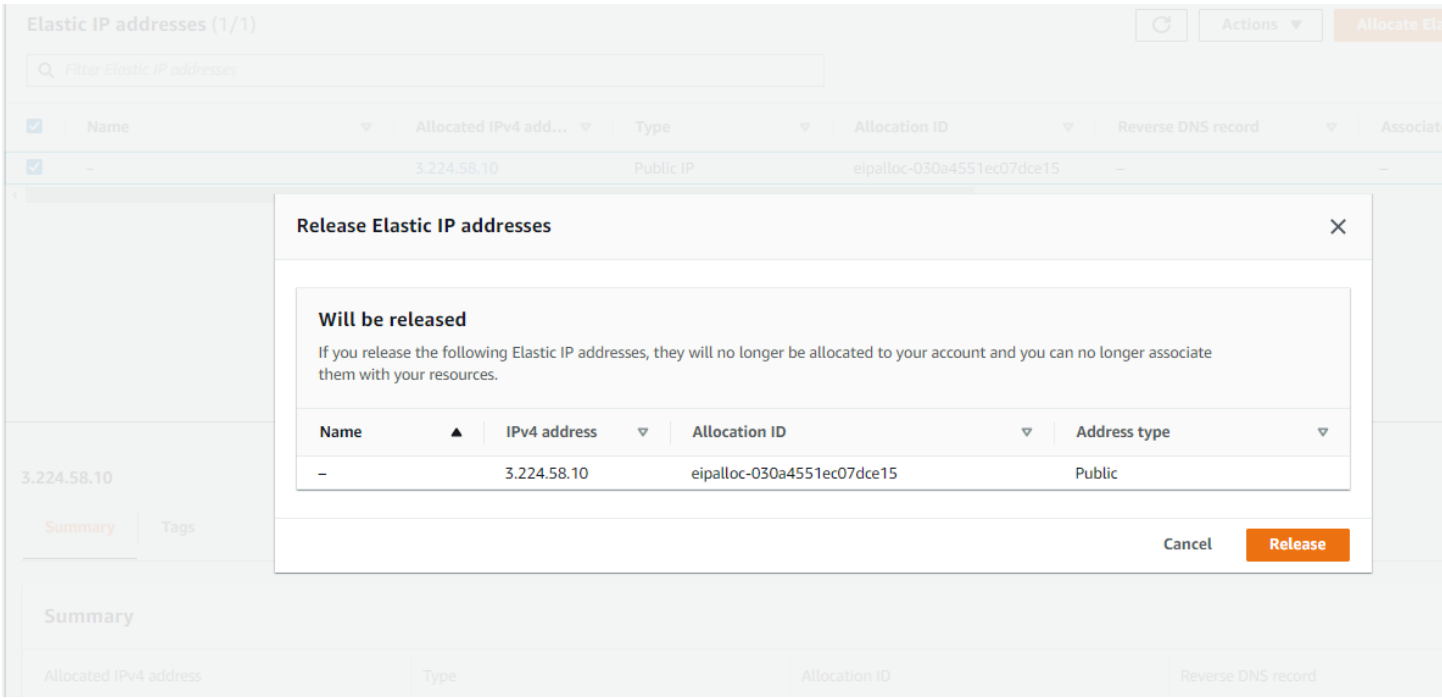
Screenshots


us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#Instances:
Home
std::upper_bound -...
குறள்திறன் - வ...
JL-Redhills
JL-Sudhakar
PCRE - Perl Compat...
AZ-900 Microsoft A...
aws
Services
Search
[Alt+S]
EC2
New EC2 Experience
Tell us what you think
EC2 Dashboard
EC2 Global View
Events
Tags
Limits
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Scheduled Instances
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Successfully terminated i-04b879780c450d424,i-0fc6ec71595aa51c4
Instances (2) Info
Connect
Instance state
Find instance by attribute or tag (case-sensitive)
Name
Instance ID
Instance state
Instance type
Status
gl-db-server
i-04b879780c450d424
Terminated
t2.micro
–
gl-app-server
i-0fc6ec71595aa51c4
Terminated
t2.micro
–
Feedback
Looking for language selection? Find it in the new Unified Settings


Internet gateway igw-04a8ed3f2a451682f successfully detached from vpc-07aa93bf04d9a3386
Delete internet gateway
Are you sure that you want to delete this internet gateway?
igw-04a8ed3f2a451682f - gl-gateway
To confirm deletion, type delete in the field:
delete
Cancel
Delete internet gateway


Services
Search
[Alt+S]
N. Virginia
EC2
Select a VPC
You successfully deleted nat-01e584fcb48fa8019.
Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs
Managed prefix lists
NAT gateways (1/1) Info
Filter NAT gateways
Name
NAT gateway ID
Connectivit...
State
State message
Elastic IP address
nat-01e584fcb48fa8019
Public
Deleting
3.224.58.10
nat-01e584fcb48fa8019


Delete VPC
Will be deleted
This VPC will be deleted permanently and cannot be recovered later:
Name
gl-vpc
VPC ID
vpc-07aa93bf04d9a3386
State
Available
To confirm deletion, type delete in the field:
delete
Cancel
Delete


Release Elastic IP addresses
Will be released
If you release the following Elastic IP addresses, they will no longer be allocated to your account and you can no longer associate them with your resources.
Name
IPv4 address
Allocation ID
Address type
–
3.224.58.10
eipalloc-030a4551ec07dce15
Public
Cancel
Release