PGPCC - PROJECT

Creating a file share & sync solution using OwnCloud and AWS

Scenario - Solving the Dropbox Problem

According to recent research, 40-75% of employees are using Dropbox to share files inside and outside of their businesses. Half of those Dropbox users do this even though they know it's against the rules. More than 40% of businesses have experienced the exposure of confidential information and the estimated average cost of a data breach equaled $5.5 Million in 2011.

These files, containing sensitive company and customer data, are stored in a public cloud outside of the businesses' control - possibly even outside of the country. The potential for data leakage and security breaches is enormous and companies need to stay compliant with their own policies and procedures for security and governance.

The Solution

OwnCloud is an open source secure file sync and share solution which can help you gain control of this situation and enable you to create and deploy an enterprise scale file solution. ownCloud can run in your data center or on a public cloud, with its servers, storage etc completely managed and controlled by your IT team and management in accordance with your company's governance and security requirements.

You will implement the ownCloud solution for a small workgroup, which can cater upto 150 users by using various AWS services. Your solution will be completely deployed on public cloud.

What are you expected to do?

1. Phase 1 - Architecture

Create an architecture diagram for the final implementation.

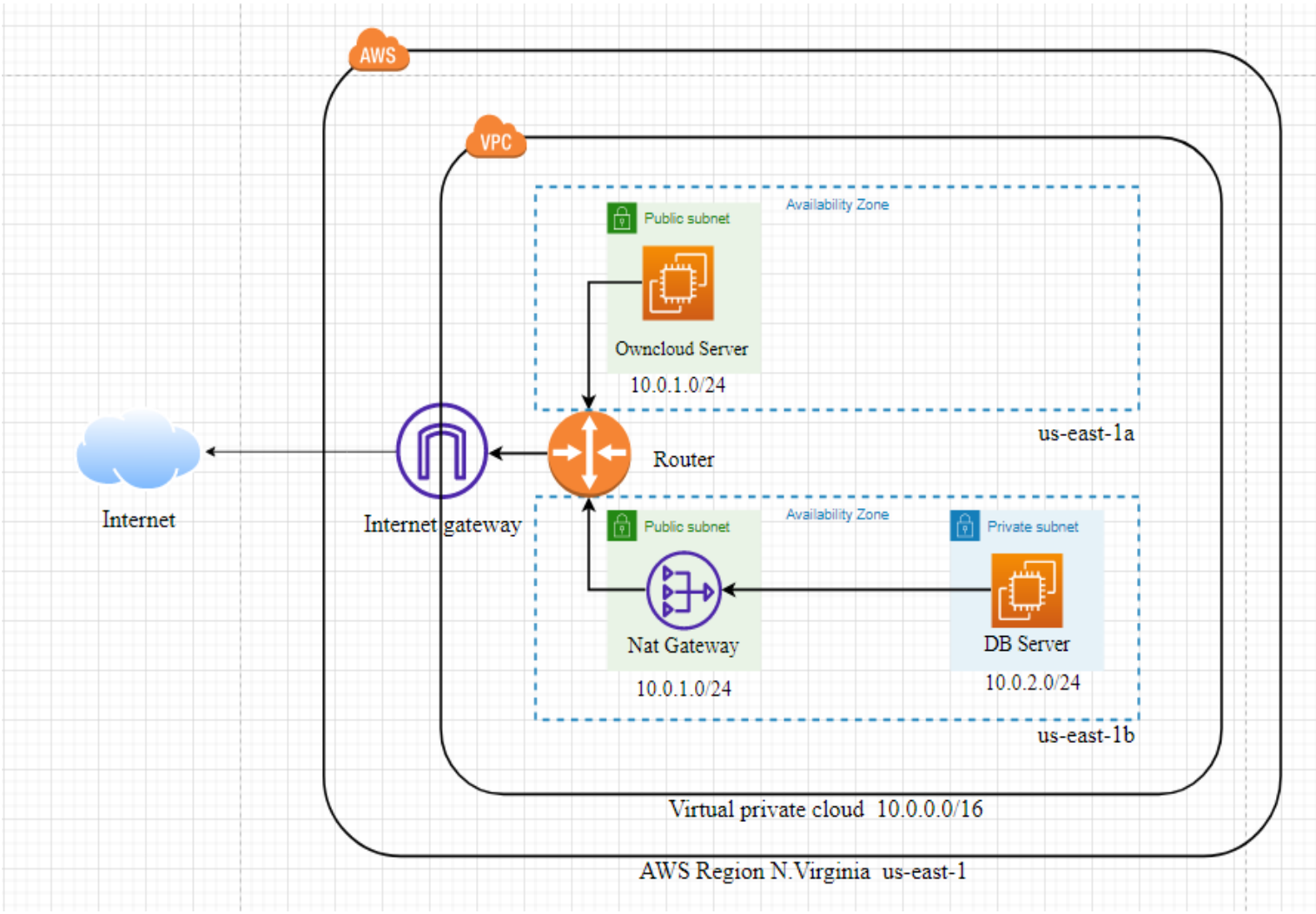
2. Phase 2 – Implementation

A. Implement 2 different subnets (one public and the other private) in a custom VPC called owncloud-vpc.

B. Install and configure MySQL database to run on the private subnet. This subnet should be associated with a security group that allows traffic to private subnet only from the public subnet.

C. The ownCloud app should be installed in public subnet and MUST be configured to access a new database called owncloud-db (created by you) in the private subnet.
Apache HTTP server should host ownCloud application in this subnet and must be configured with required PHP modules for ownCloud.

1. Phase 1 – Architecture:

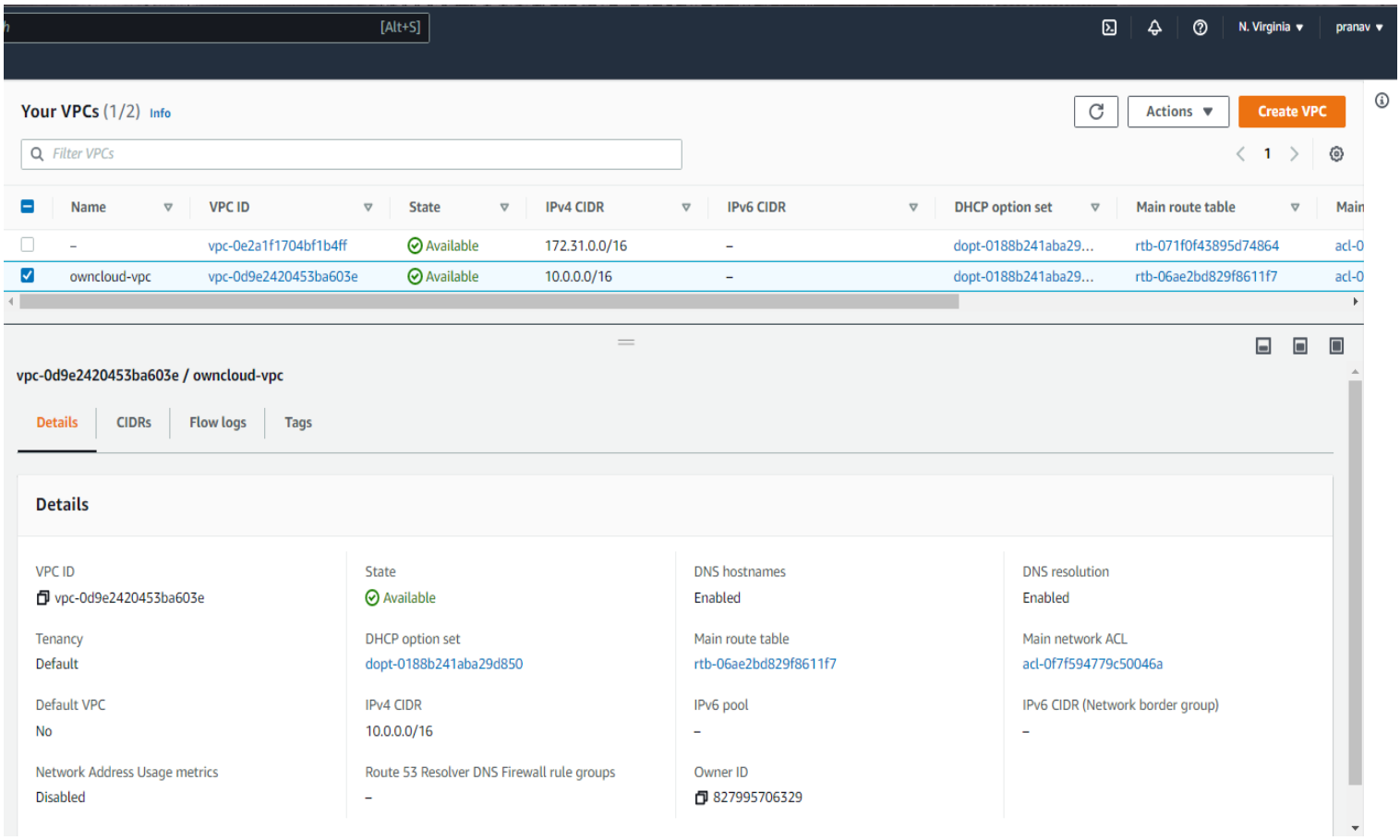


2. Phase 2 – Implementation:

Implementing 2 different subnets (one public and the other private) in a custom VPC(owncloud-vpc)

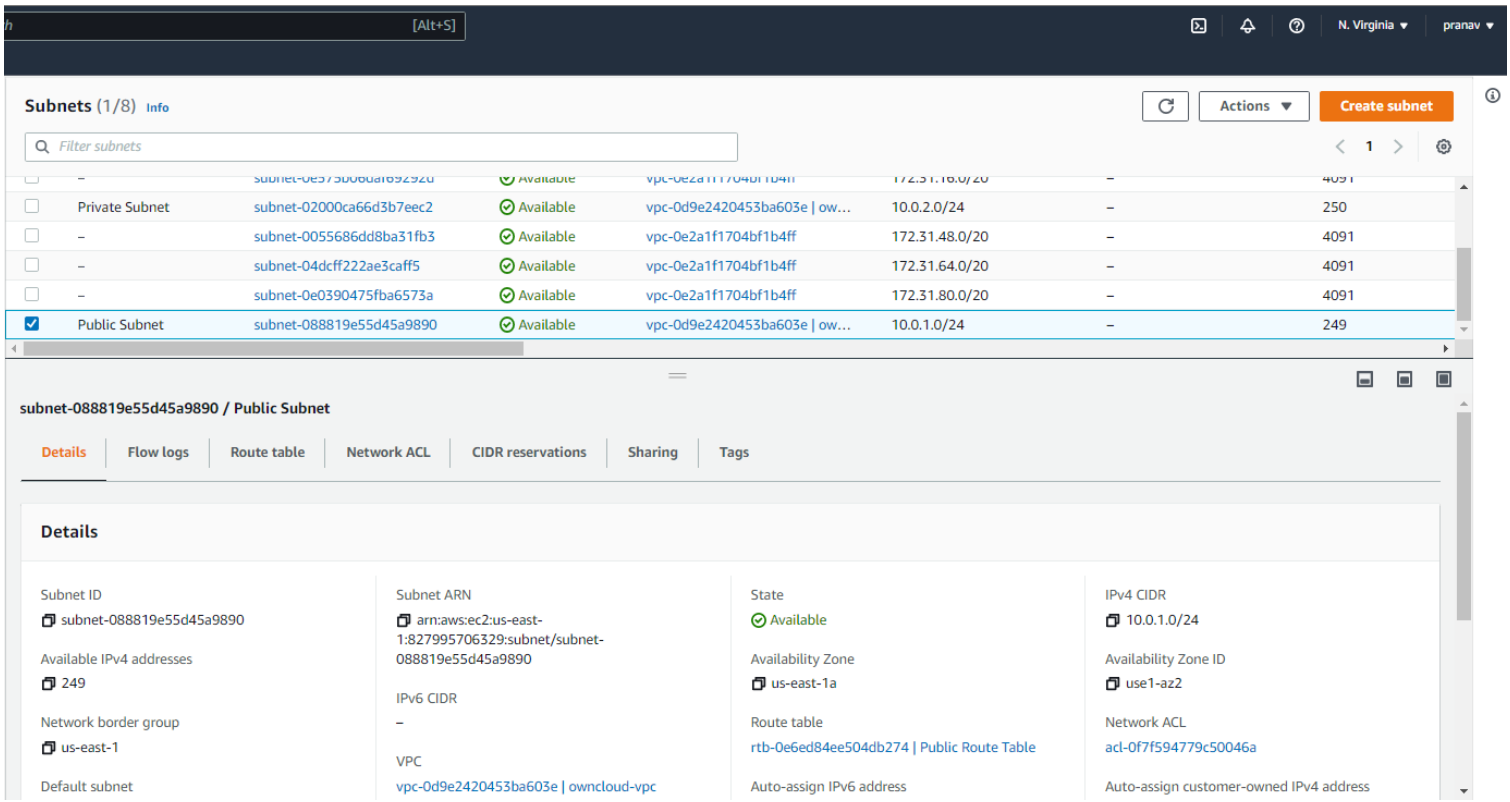
Step 1: VPC and Subnet Creation

Step number : a
Step name : Creation of **VPC**
Instructions : 1) Navigate to VPC using the Services button at the top of the screen
2) Select "Your VPCs" on the left side of the screen
3) Click on "Create VPC"
4) Enter the following fields :
Name: **owncloud-vpc**
IPv4 CIDR Block : 10.0.0.0/16
The rest of the options can be ignored
5) Select "Create VPC"



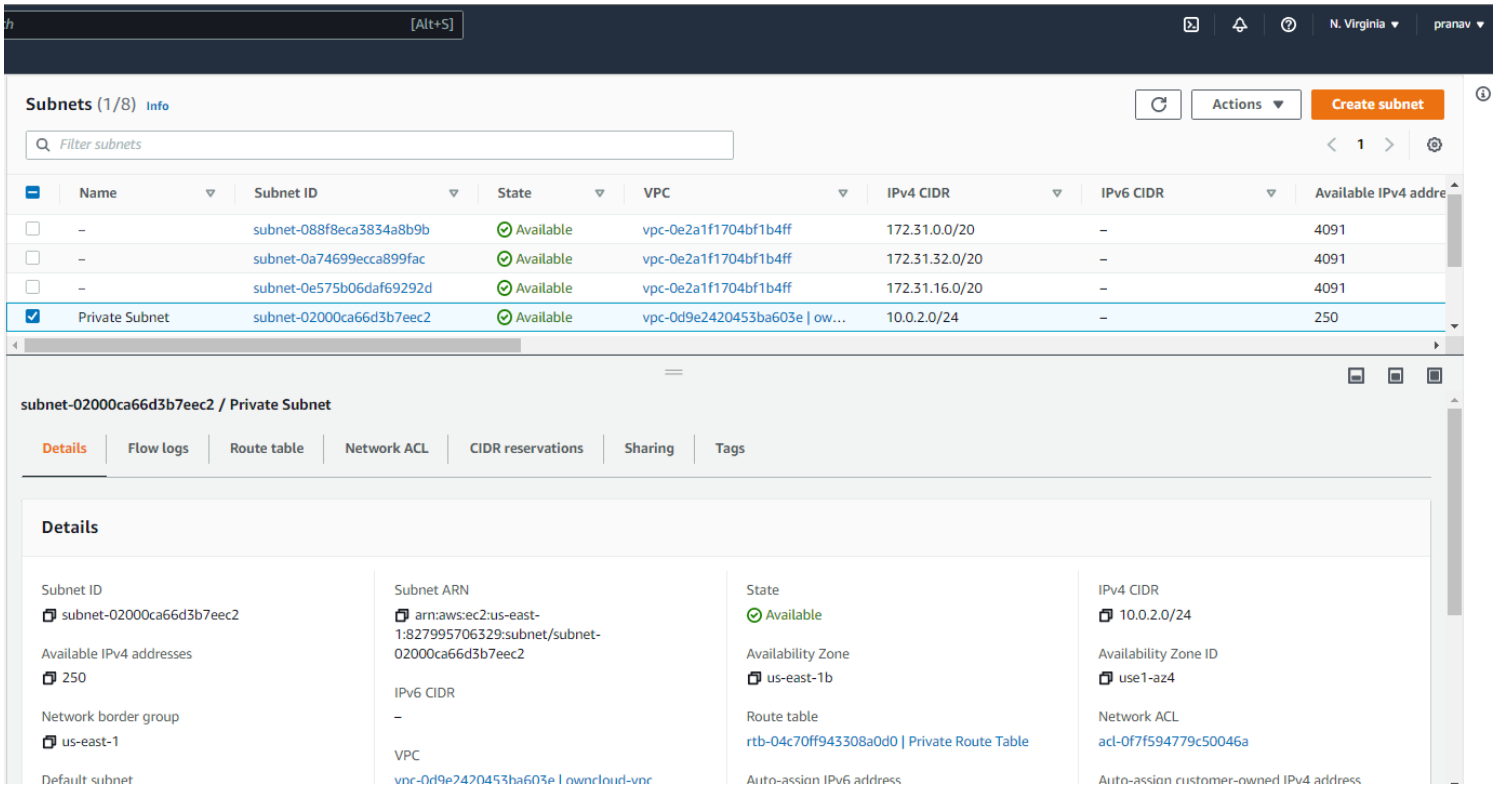
Owncloud-vpc created

Step number : b
Step name : Creation of **public subnet**
Instructions : 1) Navigate to VPC->Subnets
2) Click on "Create Subnet"
3) Enter the following fields
Name tag : Public Subnet
VPC : Select the owncloud-vpc
IPv4 CIDR block : 10.0.1.0/24
4) Click on Create
5) Once the subnet has been created, select the subnet and click on Actions->Modify Auto-assign IP settings
6) Enable the option "Auto assign IPv4" and select Save



Public Subnet created

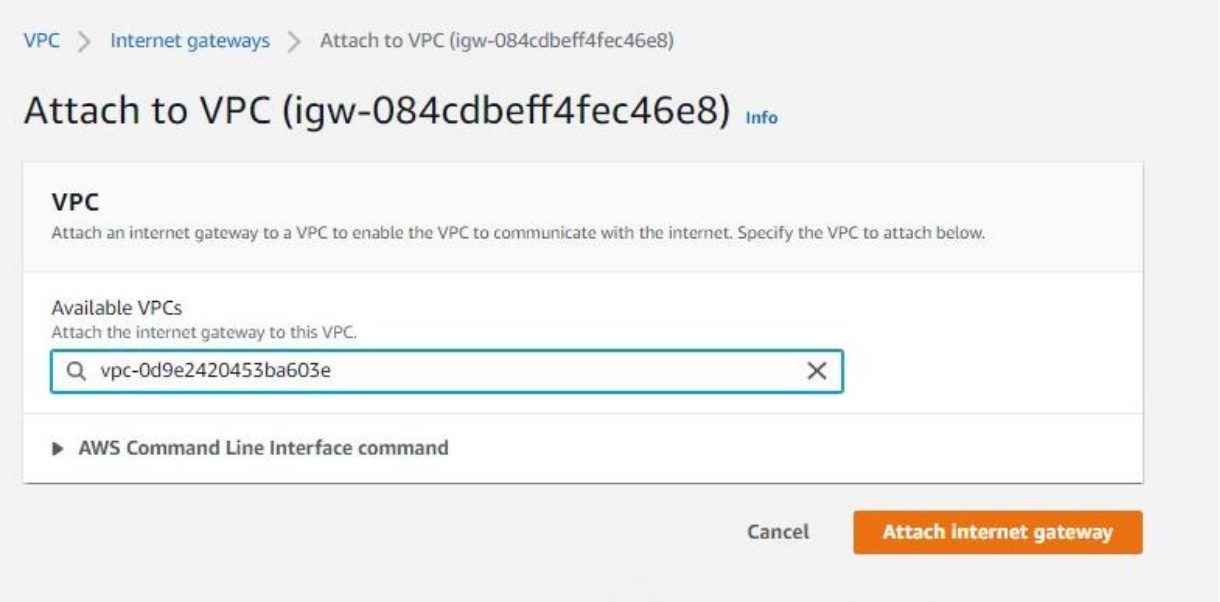
Step number : c
Step name : Creation of **private subnet**
Instructions : 1) Navigate to VPC->Subnets
2) Click on "Create Subnet"
3) Enter the following fields
Name tag : Private Subnet
VPC : Select the owncloud-vpc
IPv4 CIDR block : 10.0.2.0/24
4) Click on Create



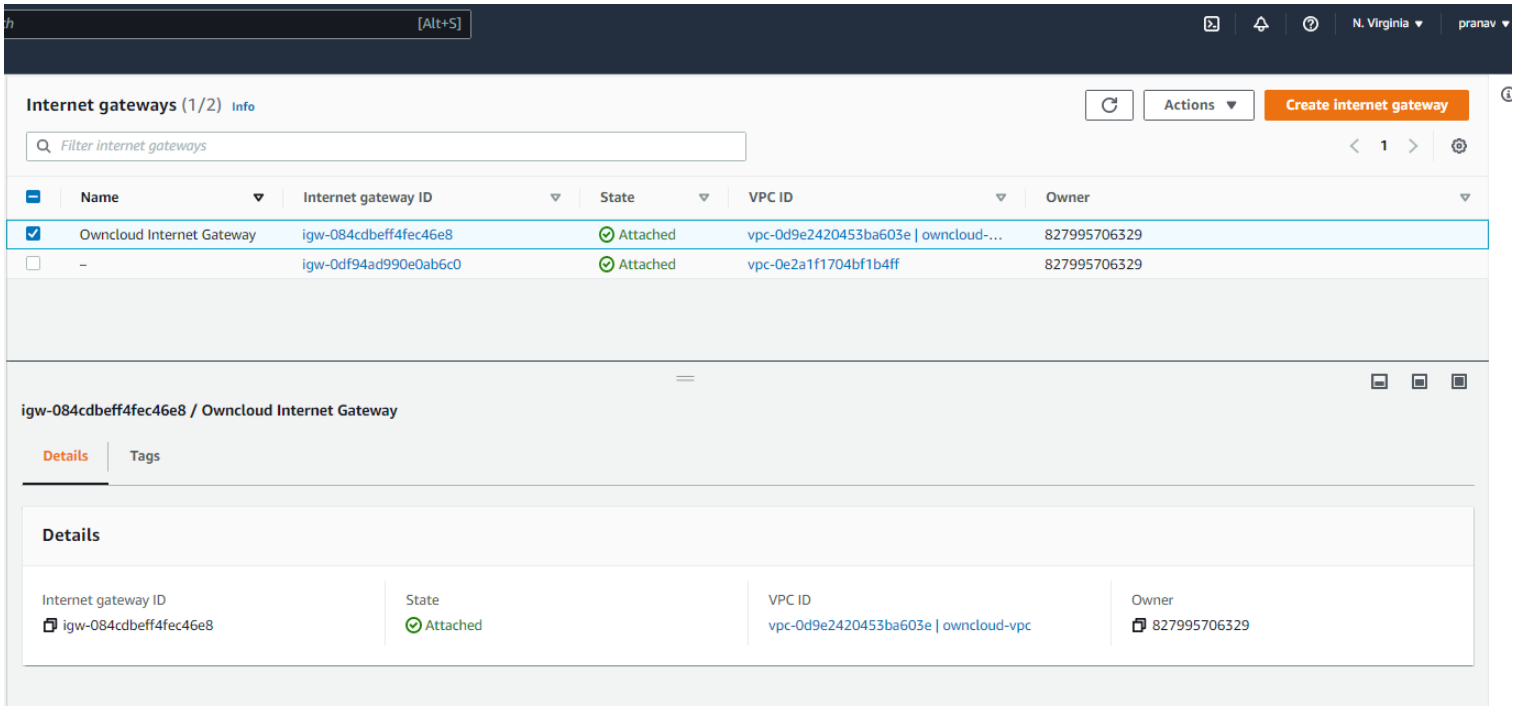
Private Subnet created

Step 2 : Internet Gateway and Route tables

Step number : a
Step name : Creation and Configuration of **Internet Gateway**
Instructions : 1) Navigate to VPCs->Internet Gateway
2) Click on "Create Internet Gateway"
3) Enter the name tag " Owncloud Internet Gateway" and click on "Create Internet Gateway"
4) After the gateway is created, select it and click on Actions ->Attach to VPC
5) Select the owncloud-vpc and click on "Attach Internet Gateway"



Attaching VPC to Internet gateway



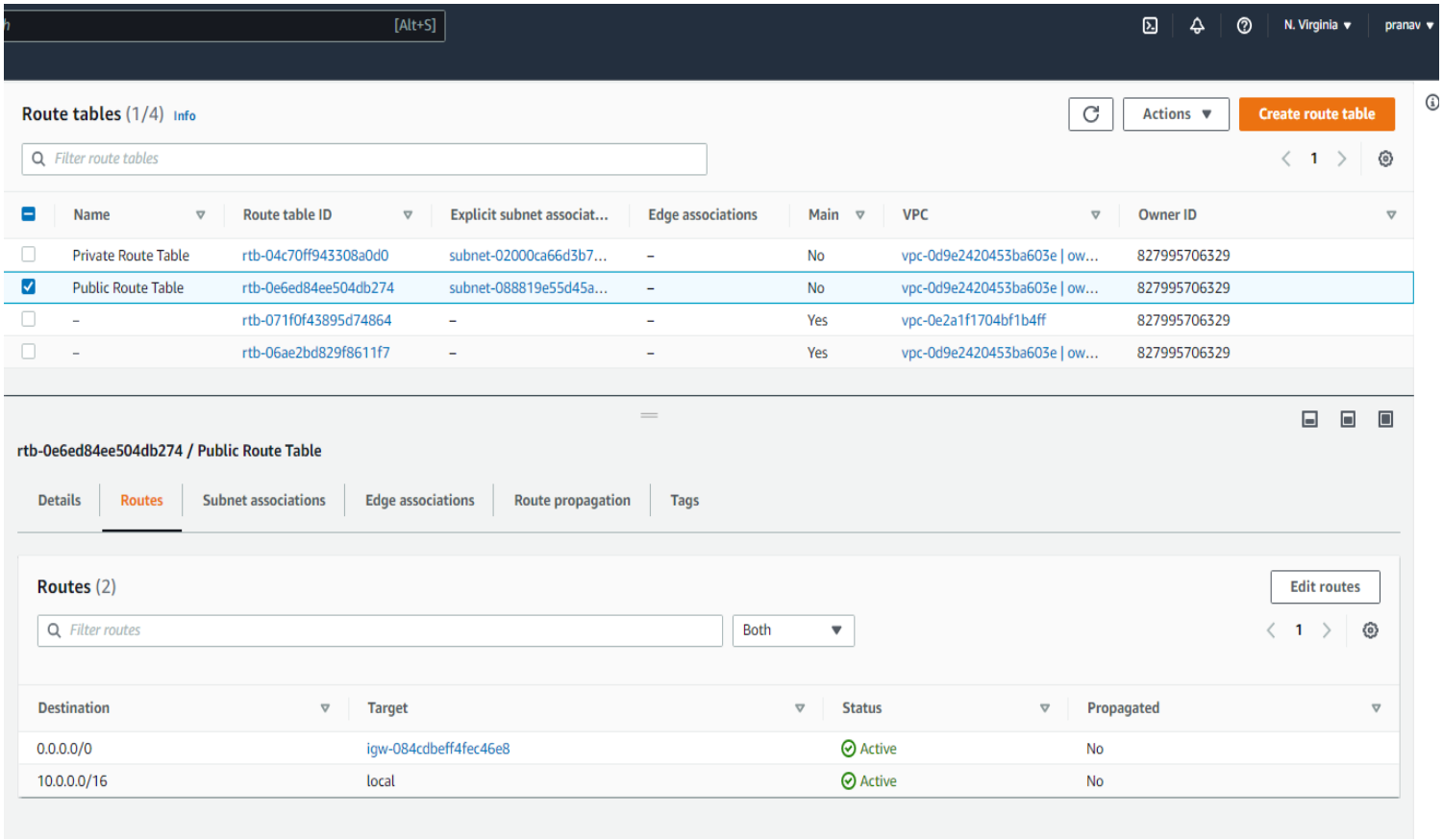
Internet gateway created with attaching VPC

Step number : b
Step name : Creation of **public route table**
Instructions : 1) Navigate to VPC -> Route Tables and click on Create Route table
2) Enter the name tag "Public Route Table", select the owncloud-vpc from the dropdown and click on Create
3) Once the route table is created, select it and select the Routes tab below the list of route tables

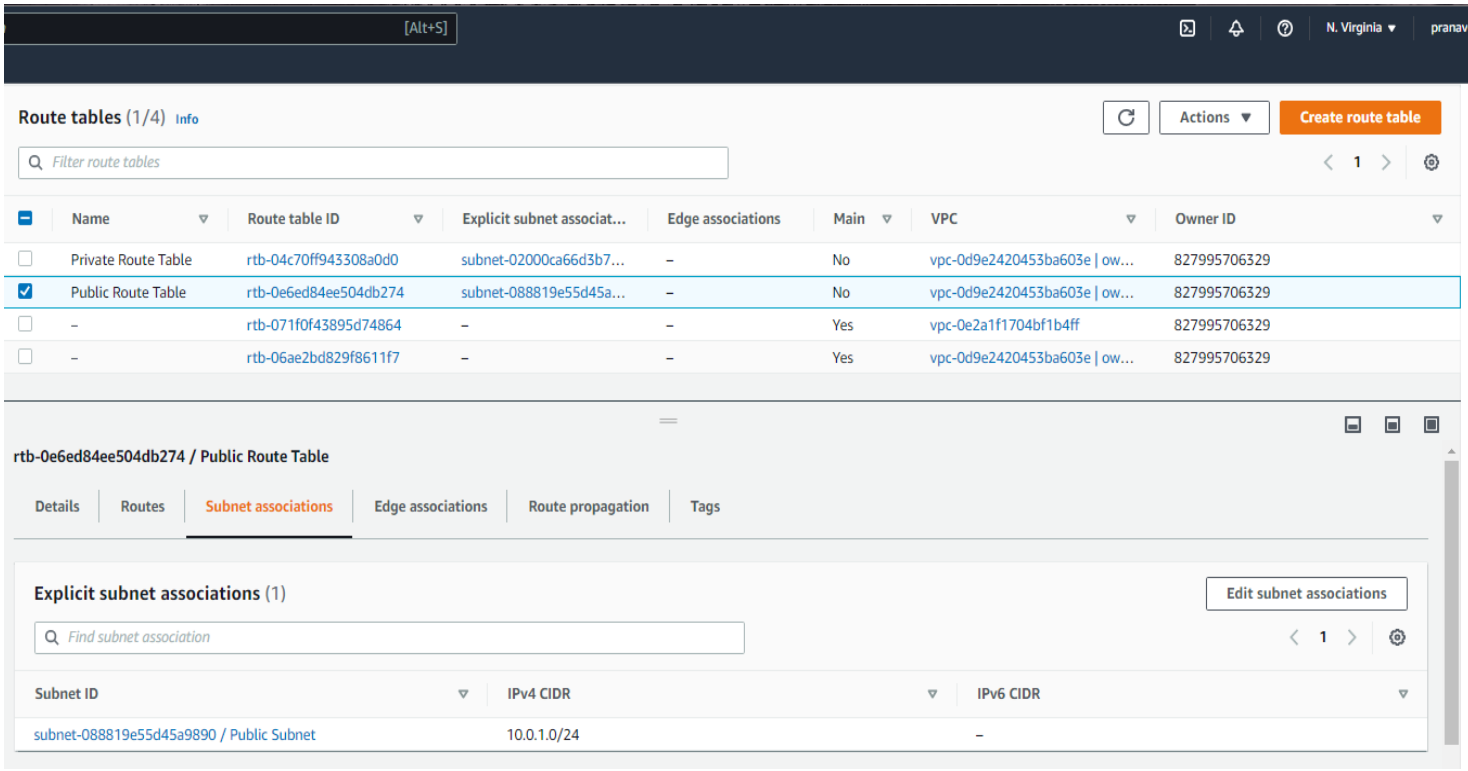
4) Click in Edit Routes and add the following route
(Don't edit the existing one)
- Destination : 0.0.0.0/0
- Target : Select Internet Gateway and the select the Owncloud Internet Gateway
Click on Save Routes

5) Select the Subnet Associations tab and click on Edit Subnet Associations

6) Select the Public Subnet from the list and click on Save

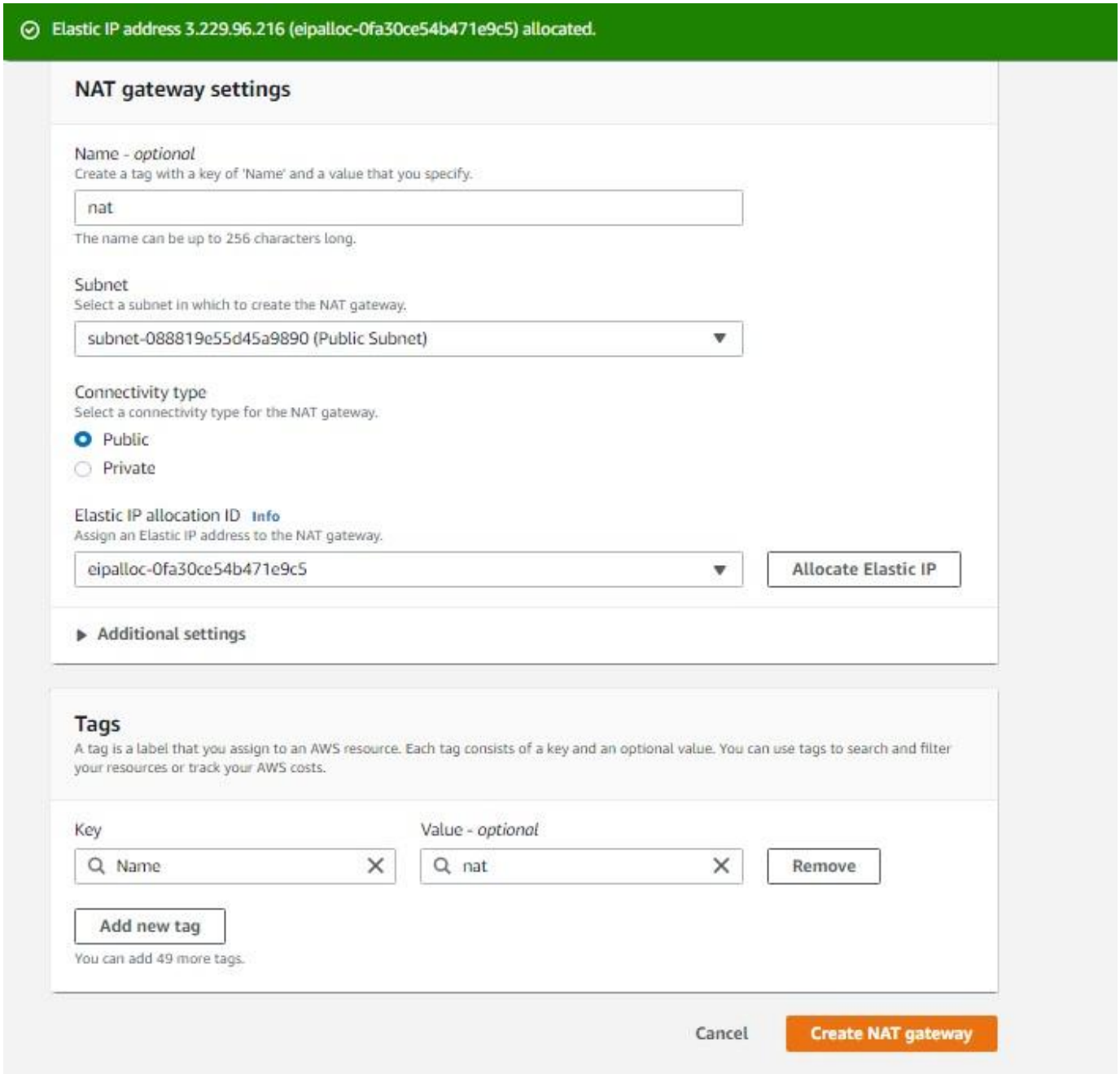


Adding a route with destination 0.0.0.0/0 and Target igw-084cdbeff4fec46e8(Owncloud Internet Gateway)

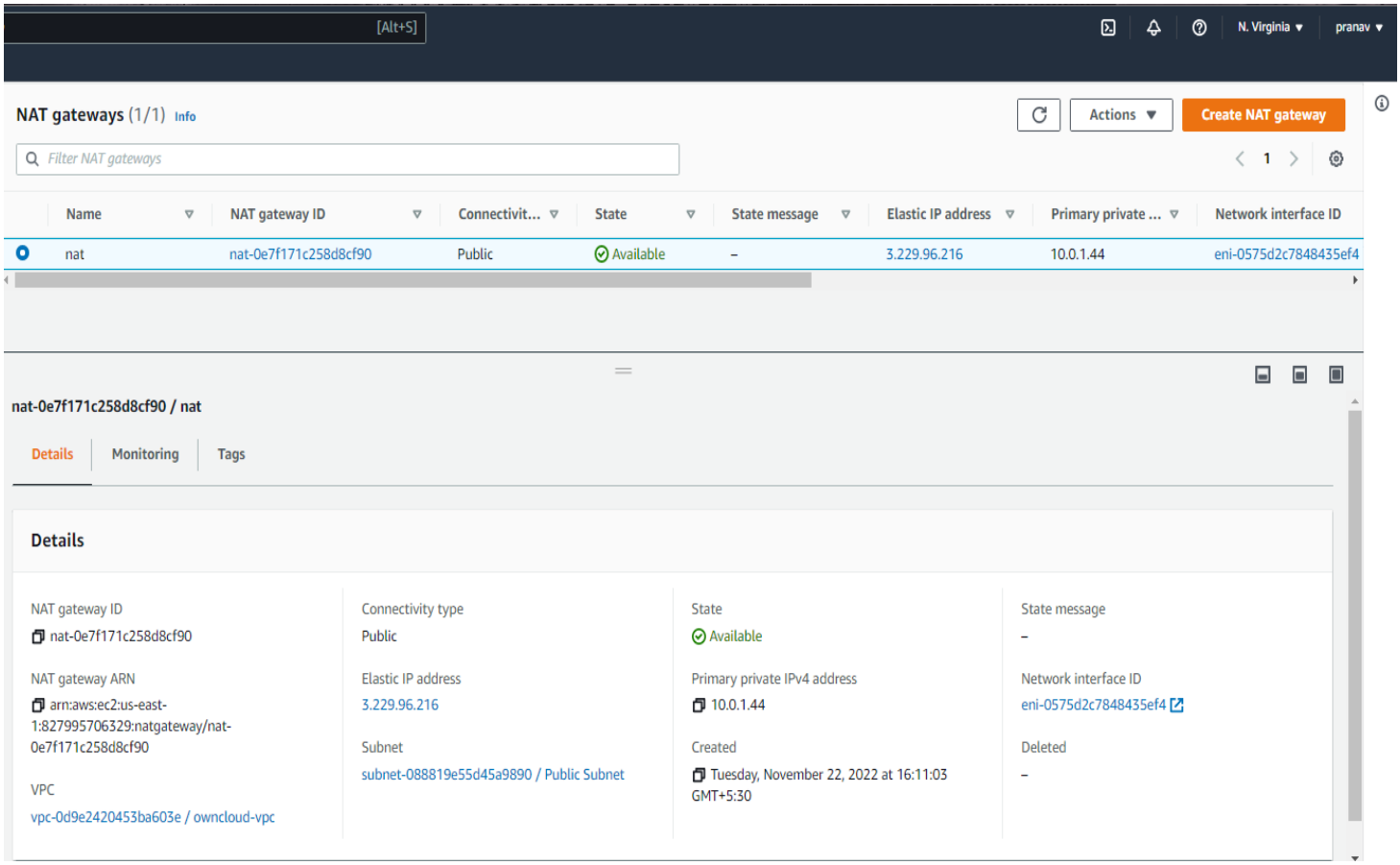


Adding public subnet to the subnet association

Step number : c
Step name : Creation of **NAT** gateway
Instructions : 1) Navigate to VPC using the Services button at the top of the screen
2) Select NAT Gateway at the left side of the screen
3) Click on Create NAT Gateway
- Deploy it in the public subnet
- Connectivity type : Public
- Allocate an elastic IP by clicking on “Allocate Elastic IP”
4) Click on “Create NAT Gateway” to create the gateway

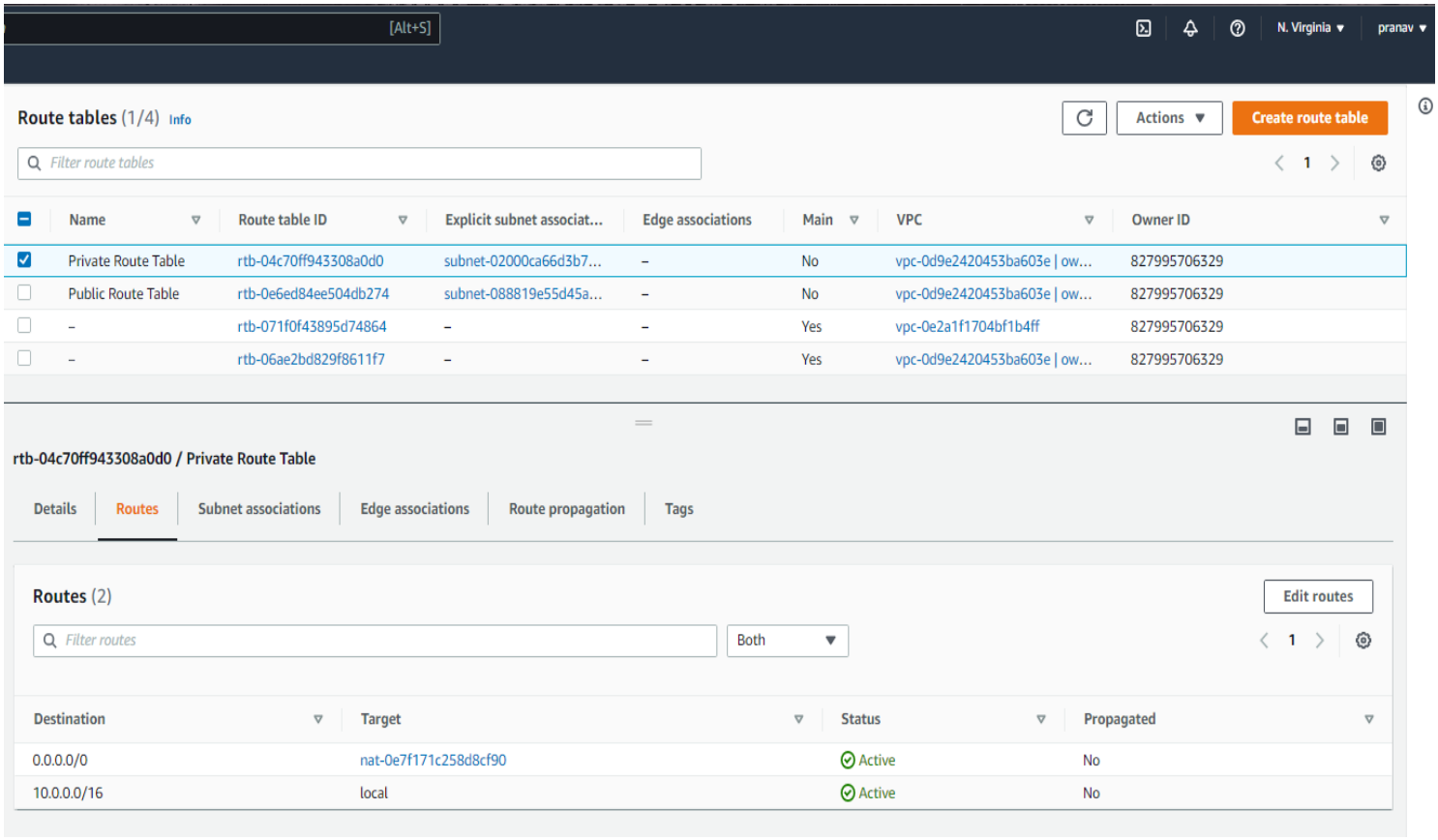


Nat gateway creation

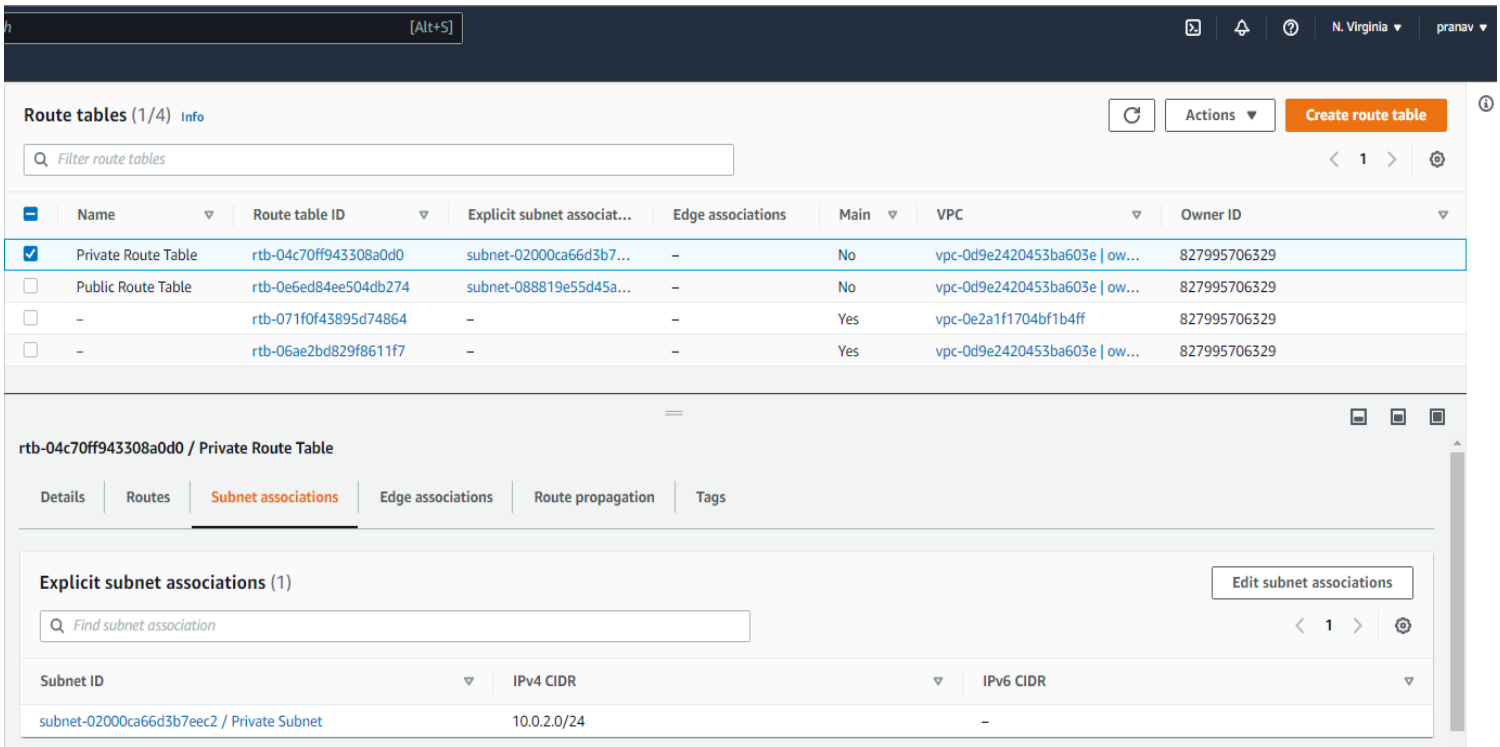


Nat gateway after creation

Step number : d
Step name : Creation of **private route table**
Instructions
1) Navigate to VPC -> Route Tables and click on Create Route table
2) Enter the name tag "Private Route Table", select the owncloud-vpc from the dropdown and click on Create
3) Once the route table is created, select it and select the Routes tab below the list of route tables
4) Click in Edit Routes and add the following route (Don't edit the existing one)
- Destination : 0.0.0.0/0
- Target: Select NAT Gateway and select the NAT Gateway created in the previous step
Click on Save Routes
5) Select the Subnet Associations tab and click on Edit Subnet Associations
6) Select the private Subnet from the list and click on Save



Adding a route with destination 0.0.0.0/0 and Target nat-0e7f171c258d8cf90(Nat gateway)



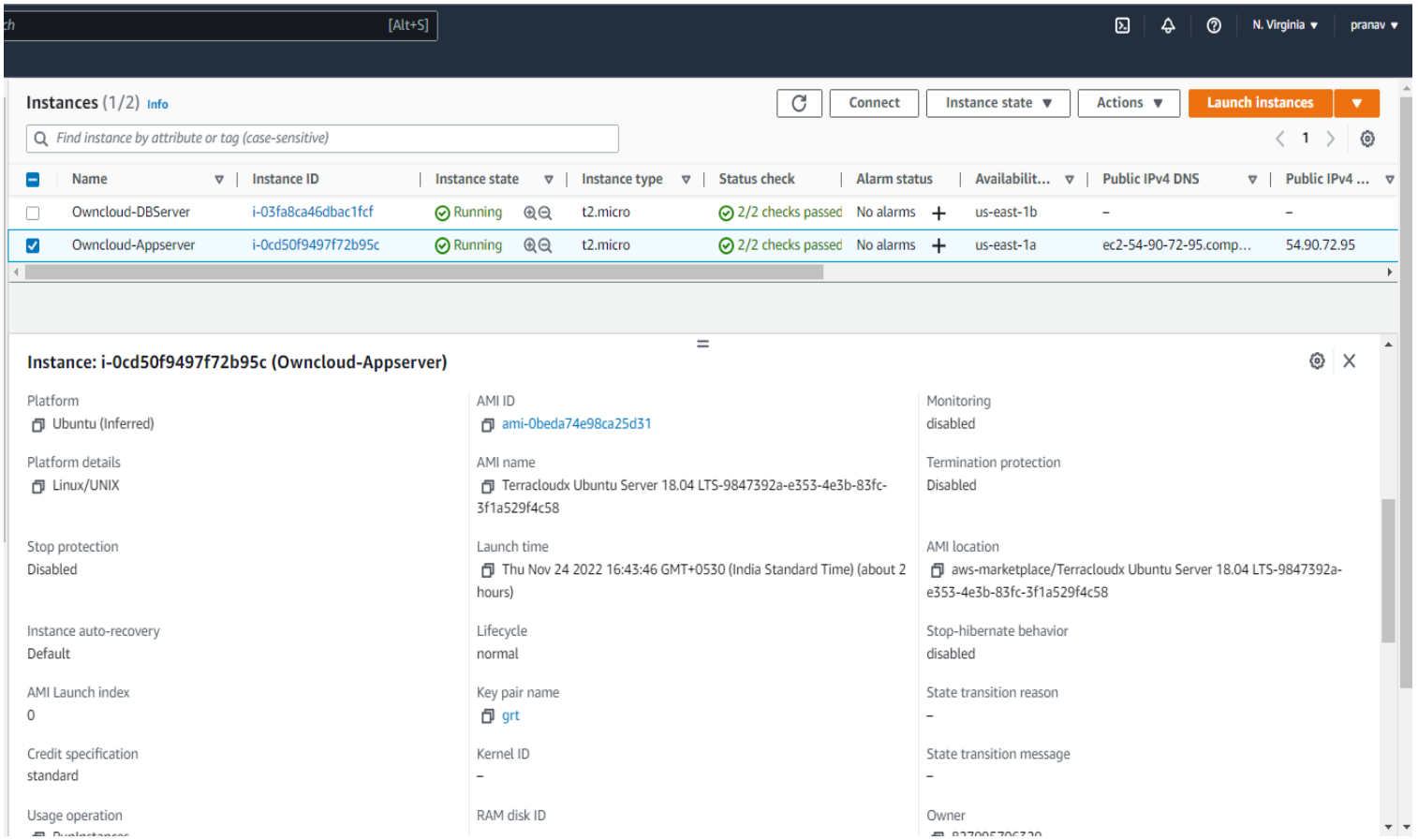
Adding private subnet to the subnet association

Step 3 : Creation of Application servers and database

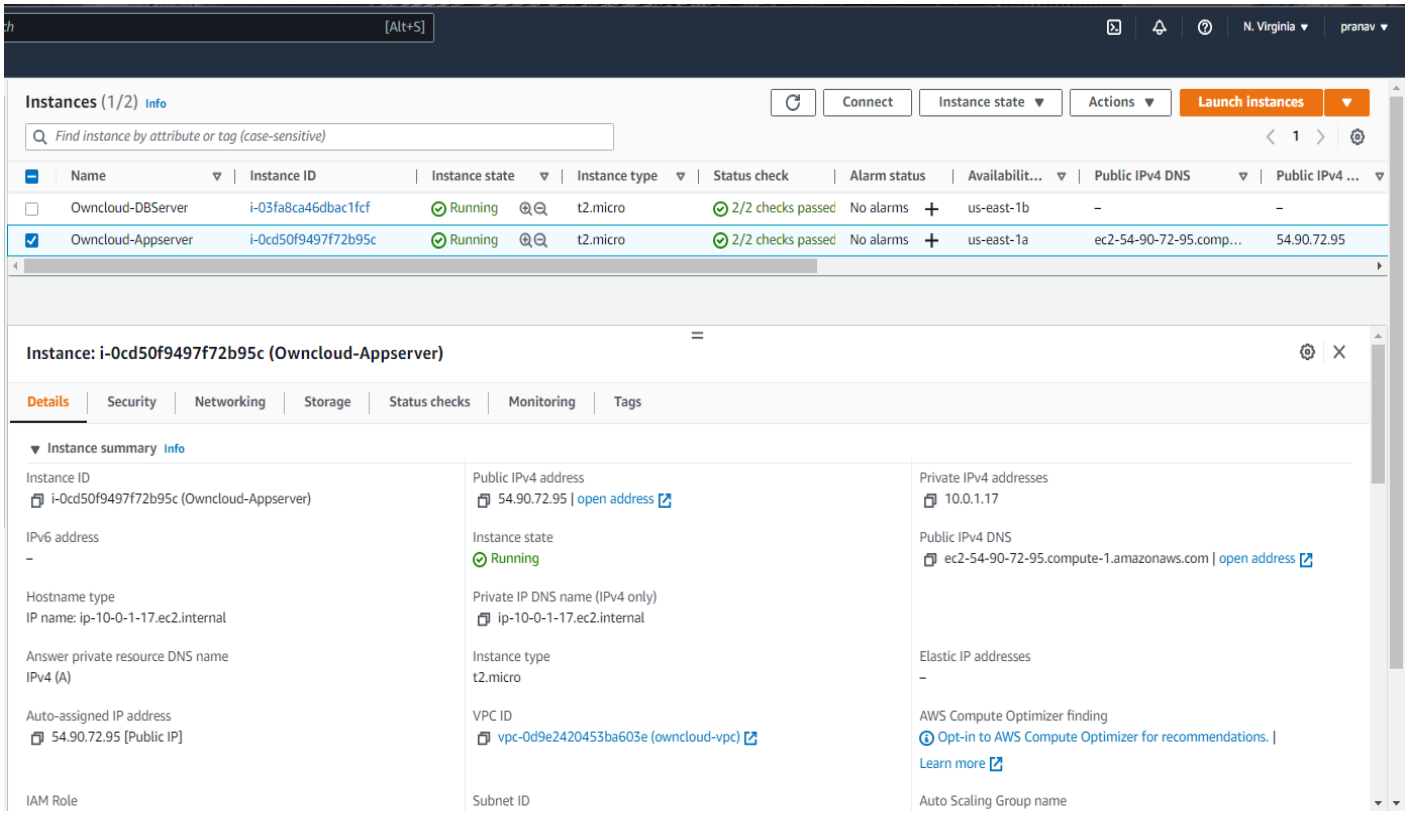
Step number : a
Step name : Creation of **Owncloud application server**
Instructions : 1) Navigate to EC2 using the Services button at the top of the screen
2) Select Instances at the left side of the screen
3) Click on Launch Instance
- Select the **Ubuntu 18.04**
- Select the instance type **t2.micro**
- Select Network as "**owncloud-vpc** " and subnet as "**Public Subnet**"
- For the security group, open the **ports 80 and 443** for source set to "**Anywhere**"

4) Launch the instance after creating a new pem file and downloading it.

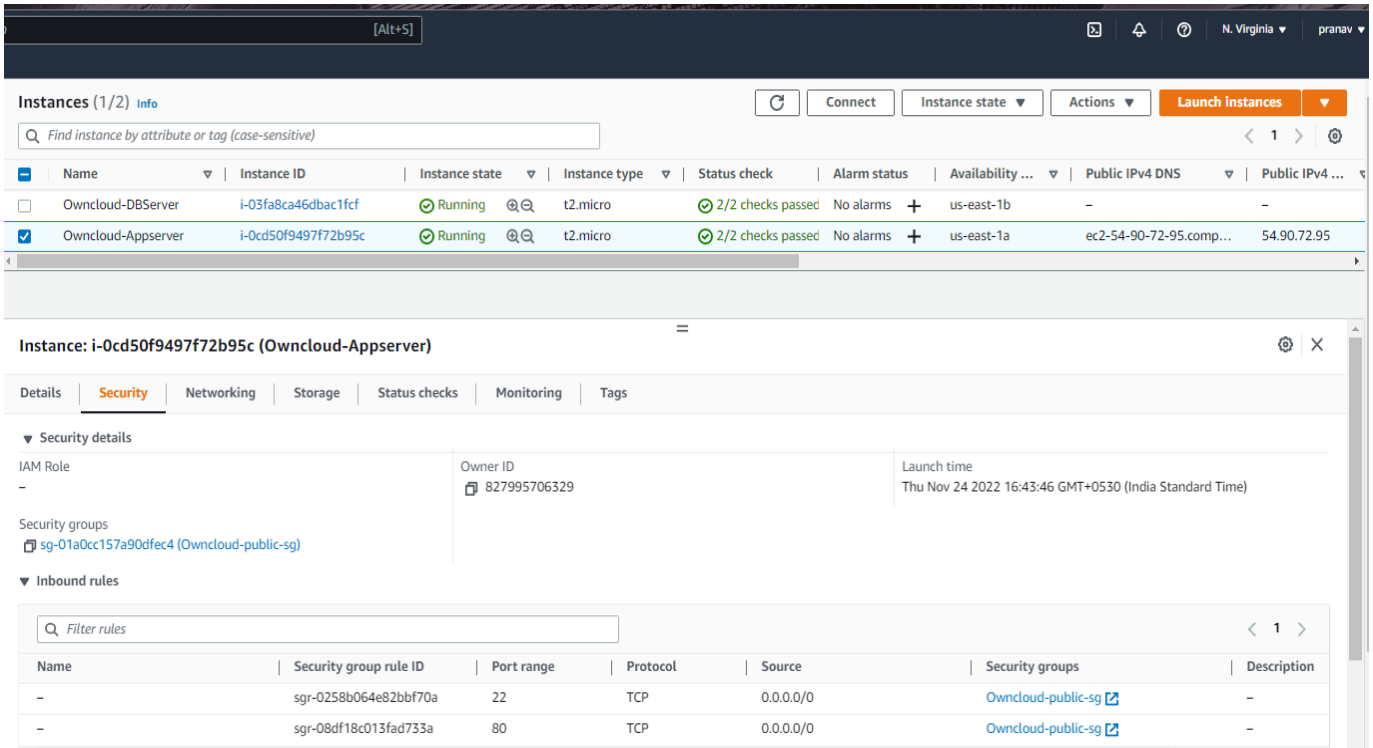
NOTE:pem file created was grt.pem



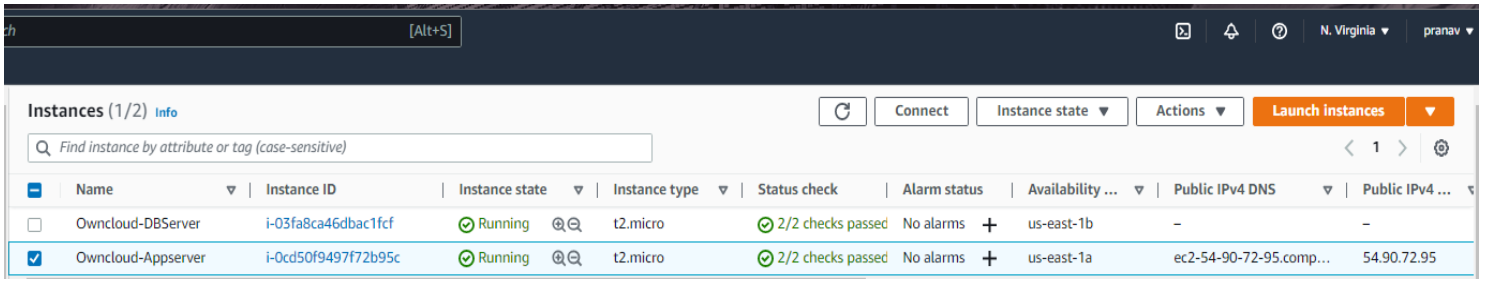
AMI used Ubuntu 18.04



Instance configuration screen with public IP 54.90.72.95



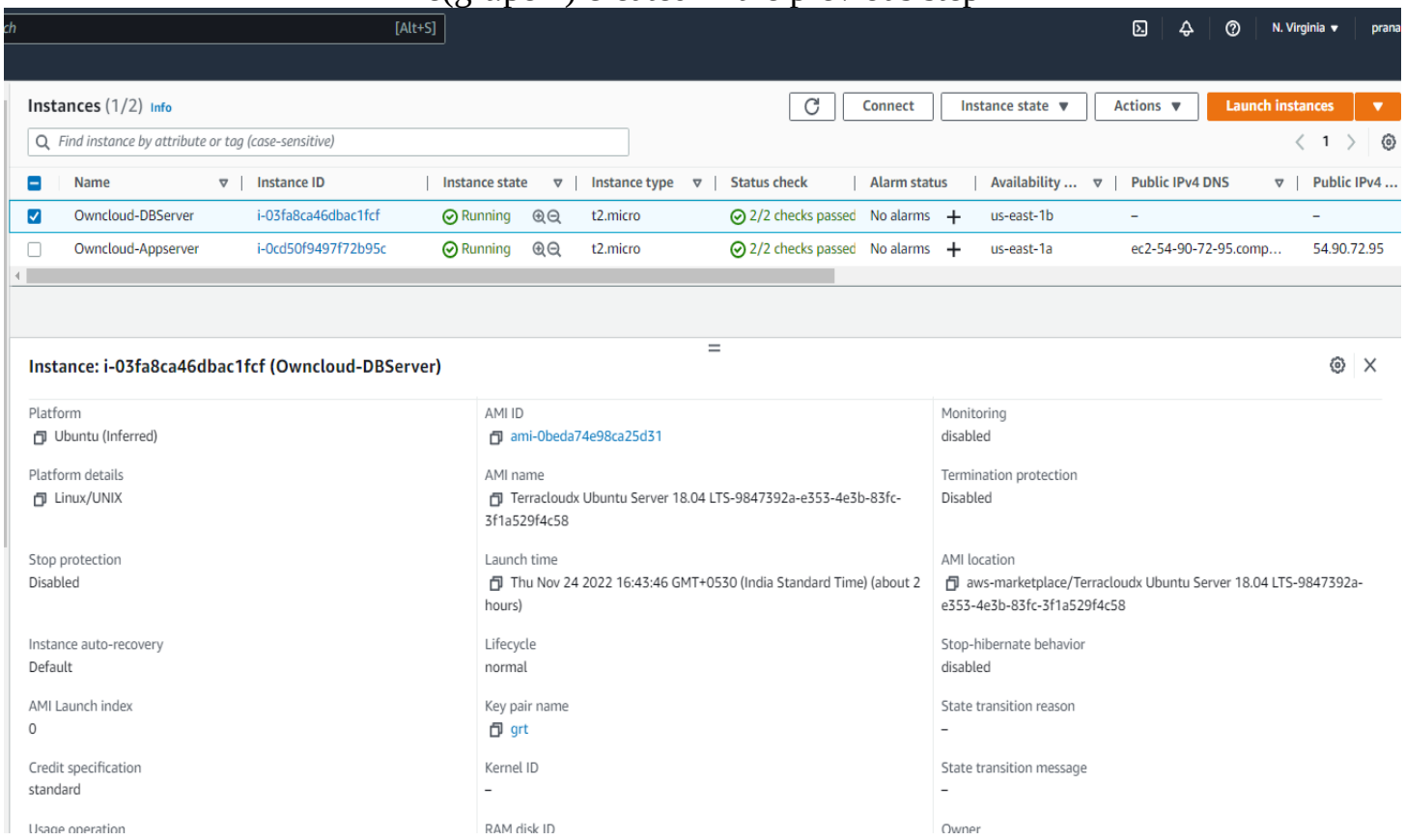
This will be assigned to Owncloud App server EC2 instance in public subnet.
It opens **SSH port 22** for remote access and **HTTP port 80** for web access.
This opens unrestricted access to above ports for the world source set to **"Anywhere"**



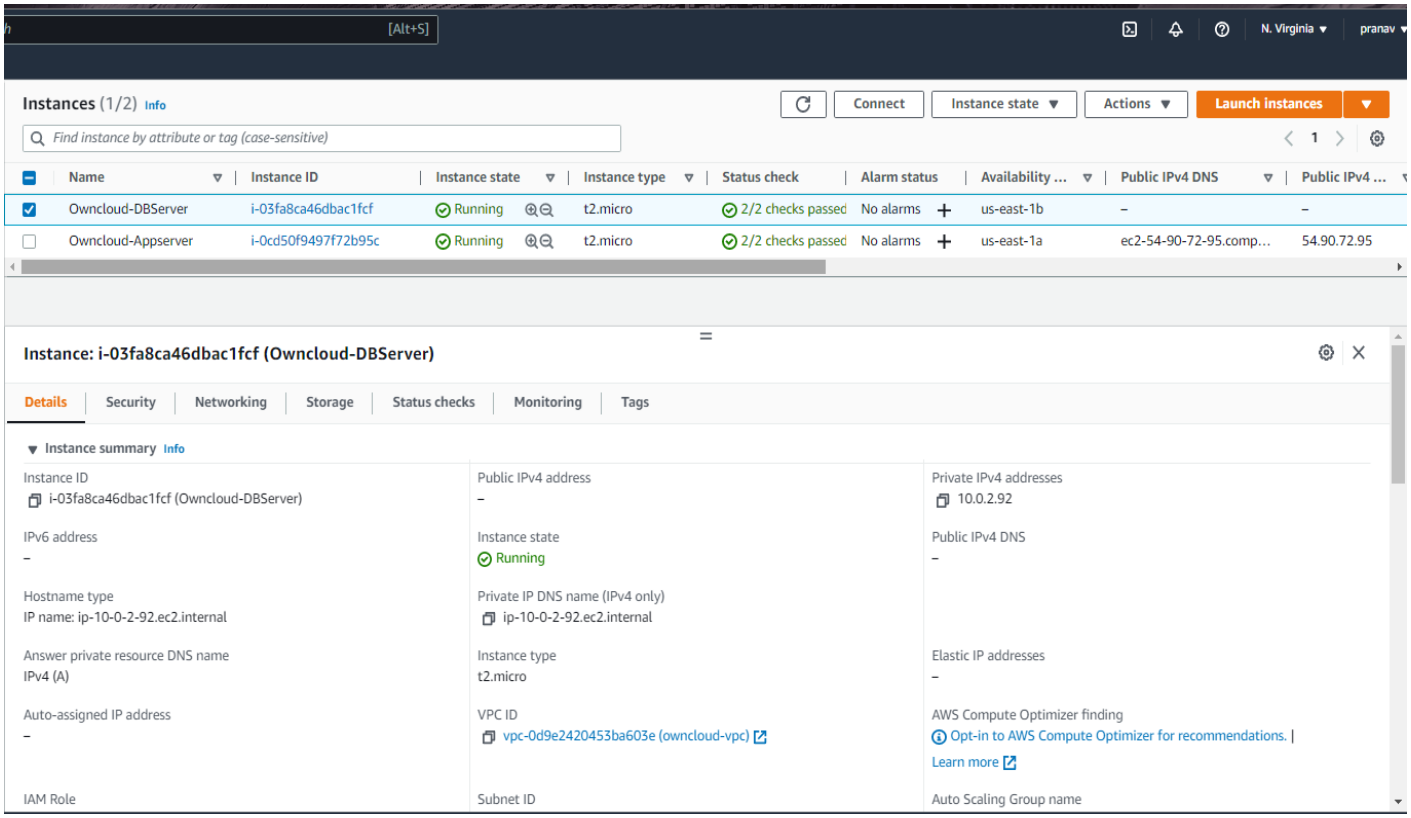
Owncloud-Appserver Instance after creation

Step number : b
Step name : Creation of **database server**
Instructions : 1) Navigate to EC2 using the Services button at the top of the screen
2) Select Instances at the left side of the screen
3) Click on Launch Instance
- Select the **Ubuntu 18.04**
- Select the instance type **t2.micro**
- Select Network as **"owncloud-vpc"** and subnet as **"Private Subnet"**
- For the security group, open the **ports 80,22 and 3306** for source set to **"Public Subnet"** which is already created

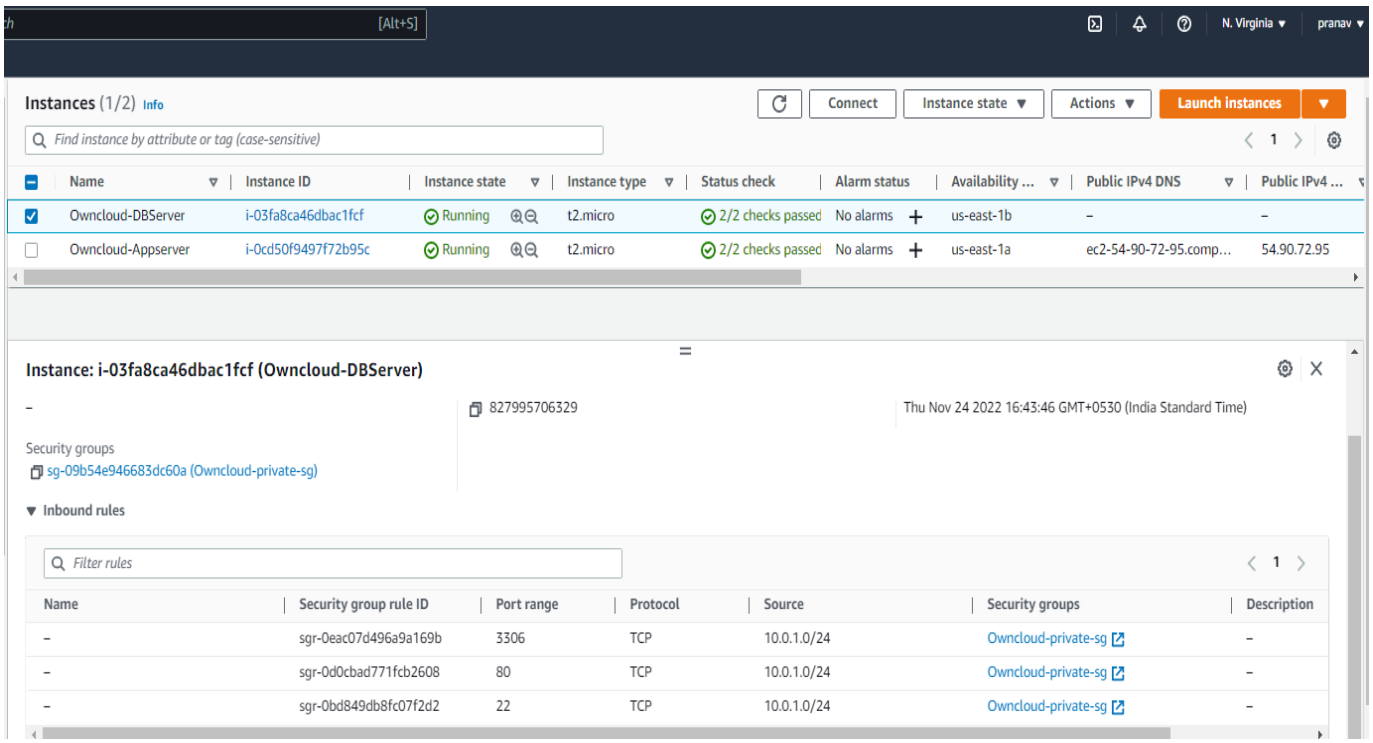
4) Launch the instance by selecting the same pem file(grt.pem) created in the previous step



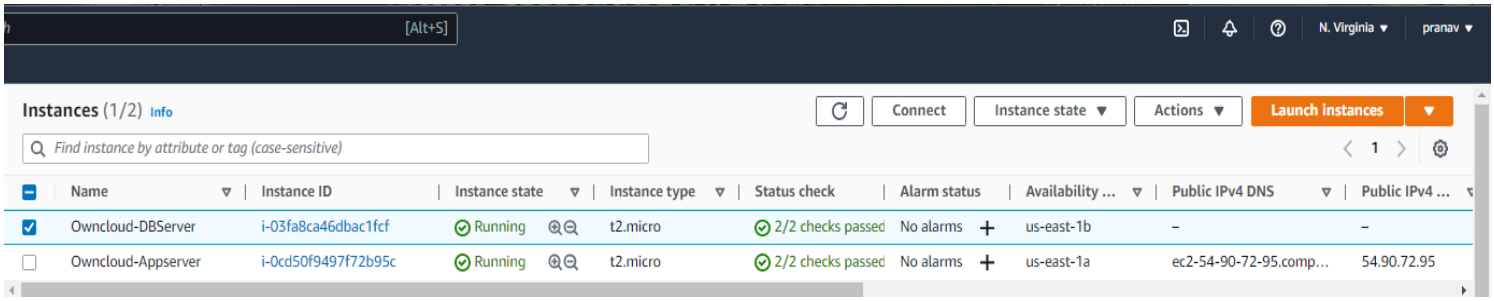
AMI used Ubuntu 18.04



Instance configuration screen with private IP 10.0.2.92



This will be assigned to database server EC2 instance in private subnet.
It opens **SSH port 22** for remote access, **HTTP port 80** for web access and **MYSQL DB port 3306** for remote database connection.
This will enable restricted access to server from **Public subnet(10.0.1.0/24)** only.



Owncloud-DBserver Instance after creation

Step4: Install Apache and PHP on Ubuntu 18.04

Step name : Installing **Apache and PHP** on **Ubuntu 18.04**
Instructions : After creating **Ubuntu 18.04** instance
(**Owncloud application server)** using 7 steps workflow.
Which Open ports 80 and 22 using security group.

1) ssh to created instance

NOTE: I used MobaXterm ssh client

a)Remote host:public IP(54.90.72.95)
b)Username:Ubuntu
c)Use private key:Select the key from local machine(grt.pem)

2) Install apache web server using following commands

- sudo apt-get update
- sudo apt-get install apache2

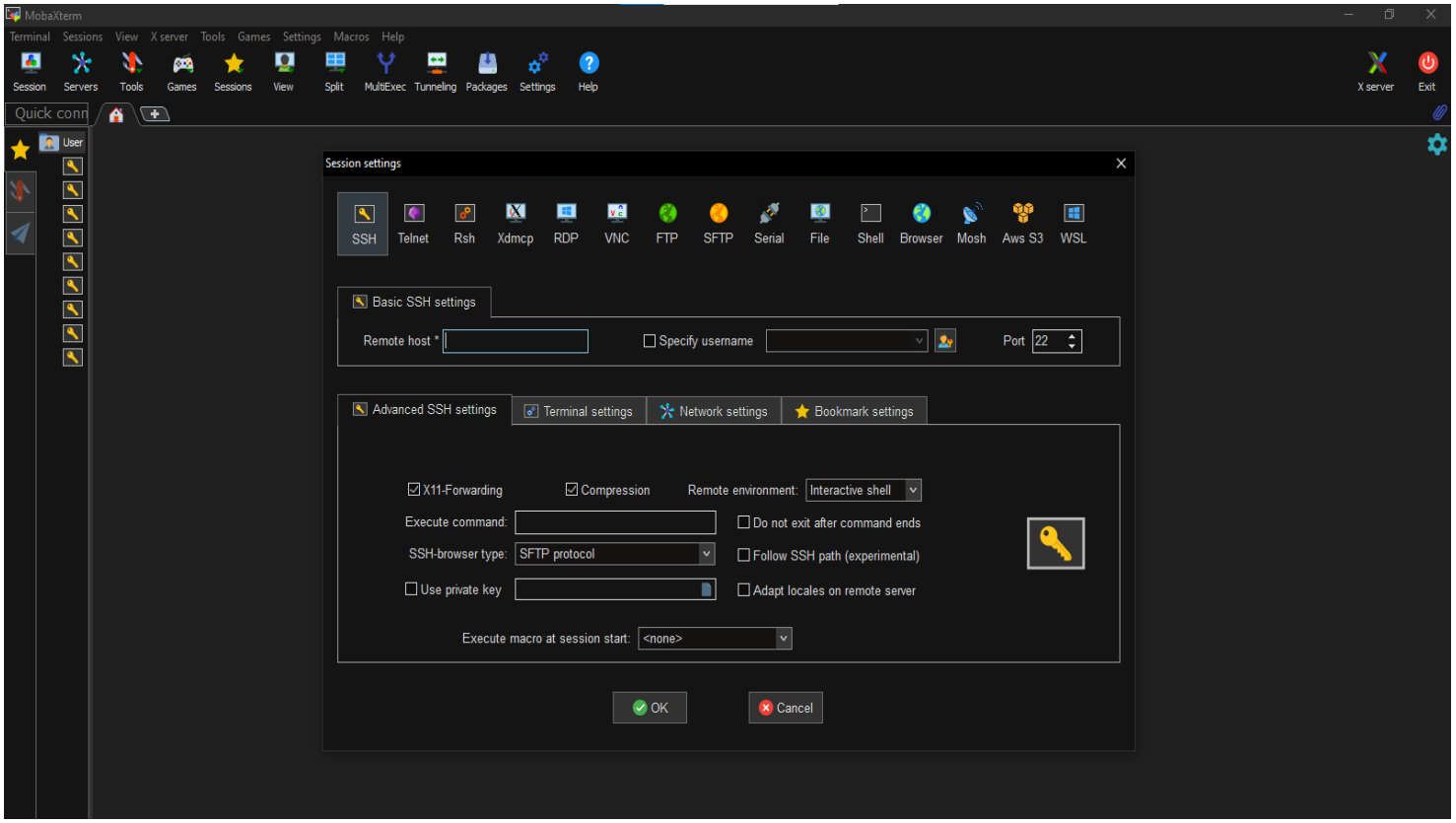
3) Validate installation by accessing public ip (54.90.72.95) of EC2 instance in browser

4) Use the following commands to install **php**
- sudo apt install php libapache2-mod-php php-mysql

5) Make index.php as the default first load page

A. Edit **/etc/apache2/mods-enabled/dir.conf** file and make **index.php** as first access page
DirectoryIndex index.php index.html index.cgi index.pl index.xhtml index.htm

B. Restart the web server - sudo systemctl restart apache2



1) **MobaXterm to ssh configuration screen**

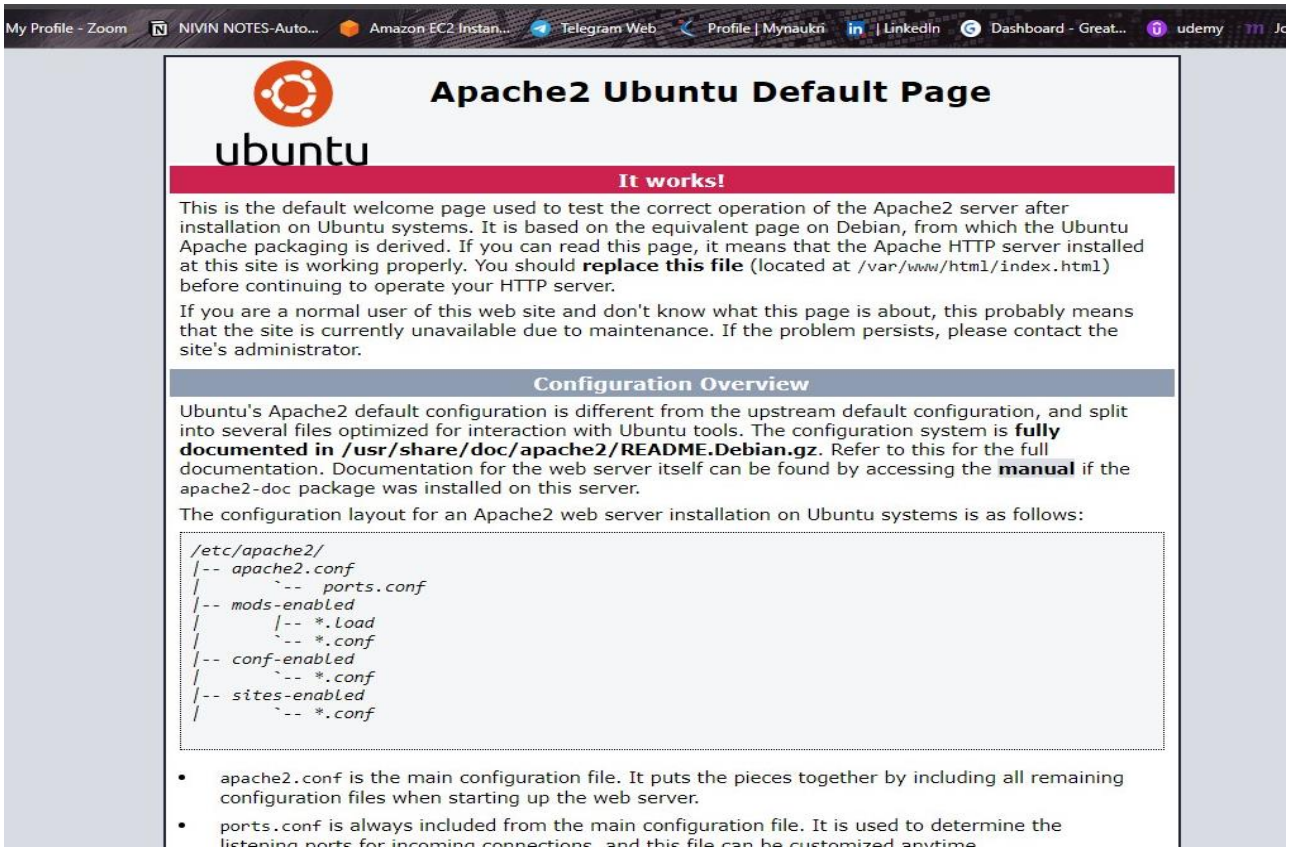
```
ubuntu@ip-10-0-1-17:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu bionic-security InRelease
Reading package lists... Done
```

2) To run **sudo apt-get update** before installing any package, and necessary to run it to install the latest updates.

As it gives Information about what updated versions of packages are available

```
ubuntu@ip-10-0-1-17:~$ sudo apt-get install apache2
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0 ssl-cert
Suggested packages:
  www-browser apache2-doc apache2-suexec-pristine | apache2-suexec-custom openssl-blacklist
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1 libaprutil1 libaprutil1-dbd-sqlite3 libaprutil1-ldap liblua5.2-0 ssl-cert
0 upgraded, 10 newly installed, 0 to remove and 52 not upgraded.
Need to get 1730 kB of archives.
After this operation, 6997 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libapr1 amd64 1.6.3-2 [90.9 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libaprutil1 amd64 1.6.1-2 [84.4 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libaprutil1-dbd-sqlite3 amd64 1.6.1-2 [10.6 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libaprutil1-ldap amd64 1.6.1-2 [8764 B]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 liblua5.2-0 amd64 5.2.4-1.1build1 [108 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 apache2-bin amd64 2.4.29-1ubuntu4.25 [1072 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 apache2-utils amd64 2.4.29-1ubuntu4.25 [83.8 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 apache2-data all 2.4.29-1ubuntu4.25 [160 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 apache2 amd64 2.4.29-1ubuntu4.25 [95.1 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 ssl-cert all 1.0.39 [17.0 kB]
Fetched 1730 kB in 0s (28.6 MB/s)
Preconfiguring packages ...
Selecting previously unselected package libapr1:amd64.
(Reading database ... 85434 files and directories currently installed.)
Preparing to unpack .../0-libapr1_1.6.3-2_amd64.deb ...
Unpacking libapr1:amd64 (1.6.3-2) ...
```

Installing apache2 using command **sudo apt-get install apache2**



3) **Default Ubuntu 18.04 Apache web page appears by Accessing public ip (54.90.72.95) of EC2 instance in browser**

```
ubuntu@ip-10-0-1-17:~$ sudo apt install php libapache2-mod-php php-mysql
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libapache2-mod-php7.2 libsodium23 php-common php7.2 php7.2-cli php7.2-common php7.2-json php7.2-mysql php7.2-opcache php7.2-readline
Suggested packages:
  php-pear
The following NEW packages will be installed:
  libapache2-mod-php libapache2-mod-php7.2 libsodium23 php php-common php-mysql php7.2 php7.2-cli php7.2-common php7.2-json php7.2-mysql php7.2-opcache
  php7.2-readline
0 upgraded, 13 newly installed, 0 to remove and 52 not upgraded.
Need to get 4142 kB of archives.
After this operation, 18.0 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 php-common all 1:60ubuntu1 [12.1 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 php7.2-common amd64 7.2.24-0ubuntu0.18.04.15 [891 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 php7.2-json amd64 7.2.24-0ubuntu0.18.04.15 [18.9 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 php7.2-opcache amd64 7.2.24-0ubuntu0.18.04.15 [165 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 php7.2-readline amd64 7.2.24-0ubuntu0.18.04.15 [12.2 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libsodium23 amd64 1.0.16-2 [143 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 php7.2-cli amd64 7.2.24-0ubuntu0.18.04.15 [1412 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libapache2-mod-php7.2 amd64 7.2.24-0ubuntu0.18.04.15 [1353 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libapache2-mod-php all 1:7.2+60ubuntu1 [3212 B]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 php7.2 all 7.2.24-0ubuntu0.18.04.15 [9236 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 php all 1:7.2+60ubuntu1 [3084 B]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 php7.2-mysql amd64 7.2.24-0ubuntu0.18.04.15 [117 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 php-mysql all 1:7.2+60ubuntu1 [2004 B]
Fetched 4142 kB in 0s (41.5 MB/s)
Selecting previously unselected package php-common.
(Reading database ... 86149 files and directories currently installed.)
Preparing to unpack .../00-php-common_1%3a60ubuntu1_all.deb ...
Unpacking php-common (1:60ubuntu1) ...
Selecting previously unselected package php7.2-common.
Preparing to unpack .../01-php7.2-common_7.2.24-0ubuntu0.18.04.15_amd64.deb ...
Unpacking php7.2-common (7.2.24-0ubuntu0.18.04.15) ...
Selecting previously unselected package php7.2-json.
Preparing to unpack .../02-php7.2-json_7.2.24-0ubuntu0.18.04.15_amd64.deb ...
Unpacking php7.2-json (7.2.24-0ubuntu0.18.04.15) ...
Selecting previously unselected package php7.2-opcache.
Preparing to unpack .../03-php7.2-opcache_7.2.24-0ubuntu0.18.04.15_amd64.deb ...
Unpacking php7.2-opcache (7.2.24-0ubuntu0.18.04.15) ...
Selecting previously unselected package php7.2-readline.
Preparing to unpack .../04-php7.2-readline_7.2.24-0ubuntu0.18.04.15_amd64.deb ...
Unpacking php7.2-readline (7.2.24-0ubuntu0.18.04.15) ...
Selecting previously unselected package libsodium23:amd64.
Preparing to unpack .../05-libsodium23_1.0.16-2_amd64.deb ...
```

4) To install PHP. In addition to the php package, you'll also need **libapache2-mod-php** to integrate PHP into Apache, and the **php-mysql** package to allow PHP to connect to MySQL databases. Run the command **sudo apt install php libapache2-mod-php php-mysql** to install all three packages and their dependencies.

```
ubuntu@ip-10-0-1-17:~$ sudo nano /etc/apache2/mods-enabled/dir.conf file
```

5) open the **dir.conf** configuration file in a text editor **nano**

```
<IfModule mod_dir.c>
        DirectoryIndex index.php index.html index.cgi index.pl index.php index.xhtml index.htm
</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

making index.php as first access page

```
ubuntu@ip-10-0-1-17:~$ sudo systemctl restart apache2
```

To restart the **Apache web server** in order for your changes to be recognized

Step 5: Owncloud Application Installation on Ubuntu 18.04

Step name : Installing **Owncloud Application** on Ubuntu 18.04

Instructions : **A.** Run following below commands

1) curl https://attic.owncloud.org/download/repositories/10.2/Ubuntu_18.04/Release.key | sudo apt-key add -
2) echo 'deb http://attic.owncloud.org/download/repositories/10.2/Ubuntu_18.04/ /' |sudo tee /etc/apt/sources.list.d/owncloud.list
3) sudo apt update
4) sudo apt install php-bz2 php-curl php-gd php-imagick php-intl php-mbstring php-xml php-zip owncloud-files

```
root@ip-10-0-1-17:/home/ubuntu# curl https://attic.owncloud.org/download/repositories/10.2/Ubuntu_18.04/Release.key |sudo apt-key add -
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  4485  100  4485    0     0  11441      0 --:--:-- --:--:-- --:--:-- 11412
OK
root@ip-10-0-1-17:/home/ubuntu# echo 'deb http://attic.owncloud.org/download/repositories/10.2/Ubuntu_18.04/ /' |sudo tee /etc/apt/sources.list.d/owncloud.list
deb http://attic.owncloud.org/download/repositories/10.2/Ubuntu_18.04/ /
root@ip-10-0-1-17:/home/ubuntu# sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:6 http://security.ubuntu.com/ubuntu bionic-security/main amd64 Packages [2470 kB]
Get:7 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [1245 kB]
Ign:5 https://attic.owncloud.org/download/repositories/10.2/Ubuntu_18.04  InRelease
Get:8 https://attic.owncloud.org/download/repositories/10.2/Ubuntu_18.04  Release [608 B]
Get:9 https://attic.owncloud.org/download/repositories/10.2/Ubuntu_18.04  Release.gpg [481 B]
Get:10 https://attic.owncloud.org/download/repositories/10.2/Ubuntu_18.04  Packages [739 B]
Fetched 3805 kB in 2s (2519 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
52 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-10-0-1-17:/home/ubuntu# sudo apt install php-bz2 php-curl php-gd php-imagick php-intl php-mbstring php-xml php-zip owncloud-files
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  fontconfig-config fonts-dejavu-core fonts-droid-fallback fonts-noto-mono ghostscript gsfonts imagemagick-6-common libavahi-client3 libavahi-common-data
  libavahi-common3 libcups2 libcupsfilters1 libcupsimage2 libfftw3-double3 libfontconfig1 libgd3 libgomp1 libgs9 libgs9-common libijs-0.35 libjbig0 libjbig2dec0
  libjpeg-turbo8 libjpeg8 liblcms2-2 liblqr-1-0 libltdl7 libmagickcore-6.q16-3 libmagickwand-6.q16-3 libpaper-utils libpaper1 libtiff5 libwebp6 libxpm4 libzip4
  php7.2-bz2 php7.2-curl php7.2-gd php7.2-intl php7.2-mbstring php7.2-xml php7.2-zip poppler-data ttf-dejavu-core
Suggested packages:
  fonts-noto ghostscript-x cups-common libfftw3-bin libfftw3-dev libgd-tools liblcms2-utils libmagickcore-6.q16-3-extra poppler-utils fonts-japanese-mincho
  | fonts-ipafont-mincho fonts-japanese-gothic | fonts-ipafont-gothic fonts-arphic-ukai fonts-arphic-uming fonts-nanum
The following NEW packages will be installed:
  fontconfig-config fonts-dejavu-core fonts-droid-fallback fonts-noto-mono ghostscript gsfonts imagemagick-6-common libavahi-client3 libavahi-common-data
  libavahi-common3 libcups2 libcupsfilters1 libcupsimage2 libfftw3-double3 libfontconfig1 libgd3 libgomp1 libgs9 libgs9-common libijs-0.35 libjbig0 libjbig2dec0
  libjpeg-turbo8 libjpeg8 liblcms2-2 liblqr-1-0 libltdl7 libmagickcore-6.q16-3 libmagickwand-6.q16-3 libpaper-utils libpaper1 libtiff5 libwebp6 libxpm4 libzip4
  owncloud-files php-bz2 php-curl php-gd php-imagick php-intl php-mbstring php-xml php-zip php7.2-bz2 php7.2-curl php7.2-gd php7.2-intl php7.2-mbstring php7.2-xml
  php7.2-zip poppler-data ttf-dejavu-core
0 upgraded, 53 newly installed, 0 to remove and 52 not upgraded.
Need to get 36.7 MB of archives.
After this operation, 176 MB of additional disk space will be used.
```

Installing Owncloud

B. Change default site directory to owncloud files directory using sudo user

- edit /etc/apache2/sites-enabled/000-default.conf
- update directory root path to /var/www/owncloud
- restart the server - sudo systemctl reload apache2
- Access the owncloud application using public ip of EC2 instance in browser

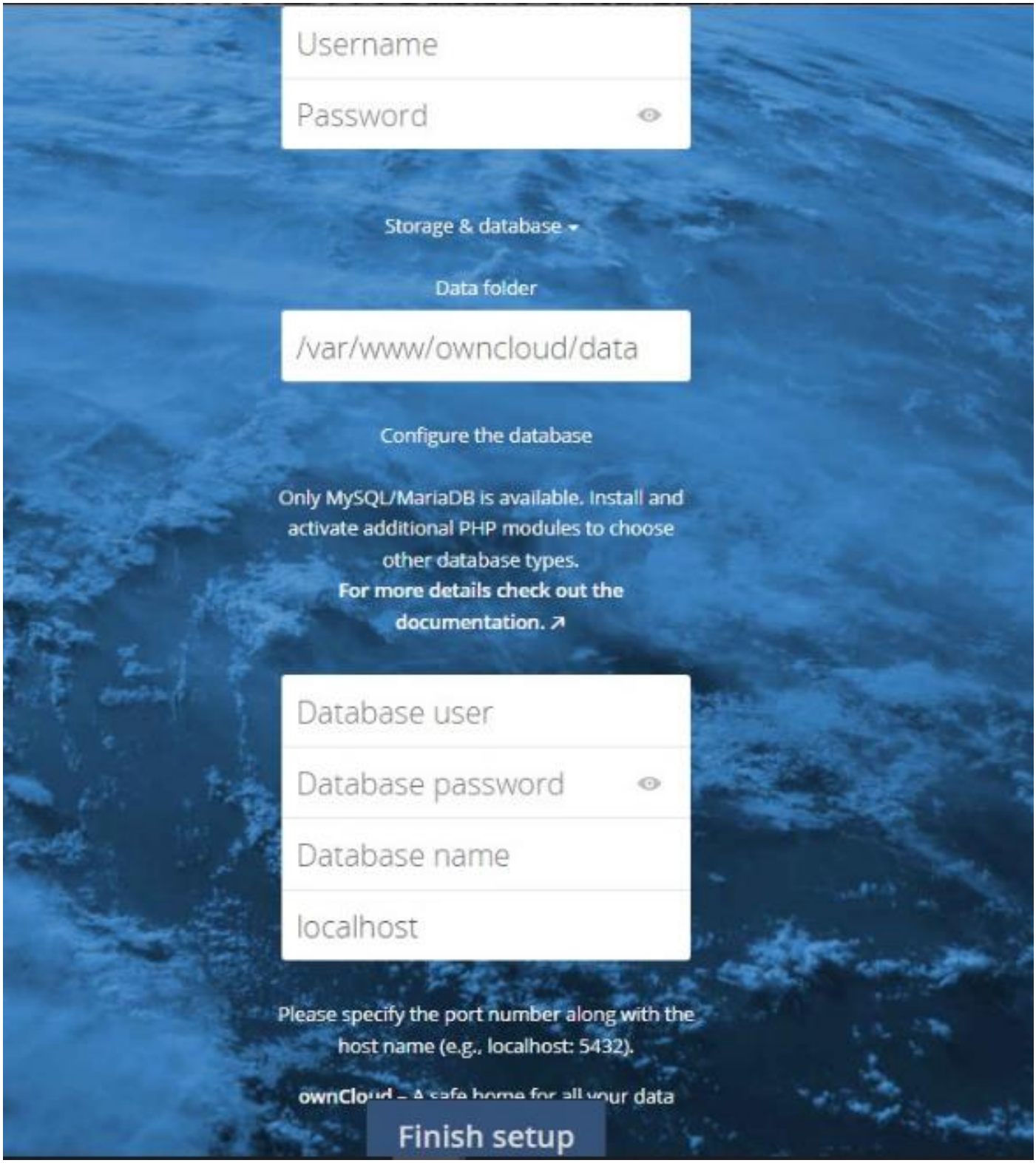
```
ubuntu@ip-10-0-1-17:~$ sudo nano edit /etc/apache2/sites-enabled/000-default.conf
```

```
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/owncloud

        # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
        # error, crit, alert, emerg.
        # It is also possible to configure the loglevel for particular
        # modules, e.g.
        #LogLevel info ssl:warn

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

        # For most configuration files from conf-available/, which are
        # enabled or disabled at a global level, it is possible to
        # include a line for only one particular virtual host. For example the
        # following line enables the CGI configuration for this host only
        # after it has been globally disabled with "a2disconf".
        #Include conf-available/serve-cgi-bin.conf
</VirtualHost>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

Updating directory root path to /var/www/owncloud

```
ubuntu@ip-10-0-1-17:~$ sudo systemctl reload apache2
```

To reload the Apache service to activate the new changes



Accessing the owncloud application using public ip(54.90.72.95) of EC2 instance in browser

Step6: Install MYSQL Server on Ubuntu 18.04

Step name : MYSQL Server on Ubuntu 18.04

Instructions : SSH into private instance.

Install Mysql-server in your Machine by running below command from your terminal.

1)Sudo apt-get install mysql-server –y

2) sudo mysql_secure_installation

3) sudo mysql -u root –p

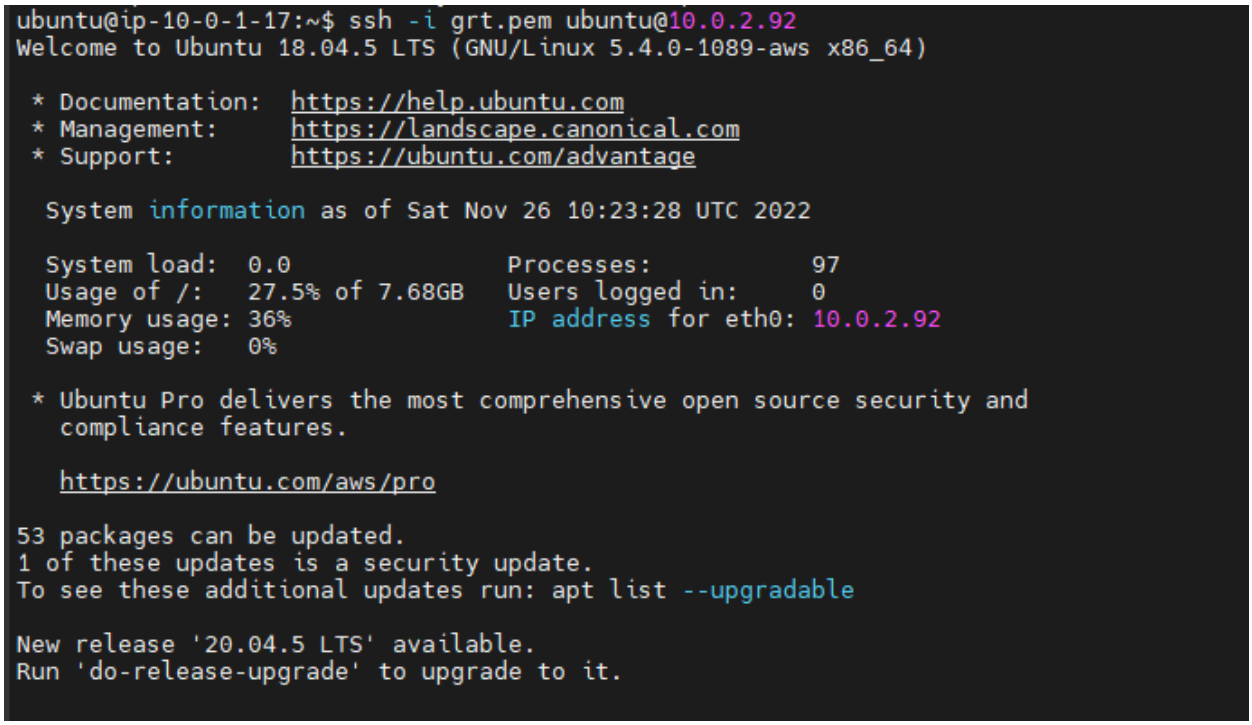
4) create user 'admin'@'%' identified by 'admin';

5) grant all privileges on *.* to 'admin'@'%' with grant option;

6)FLUSH PRIVILEGES;

7) exit

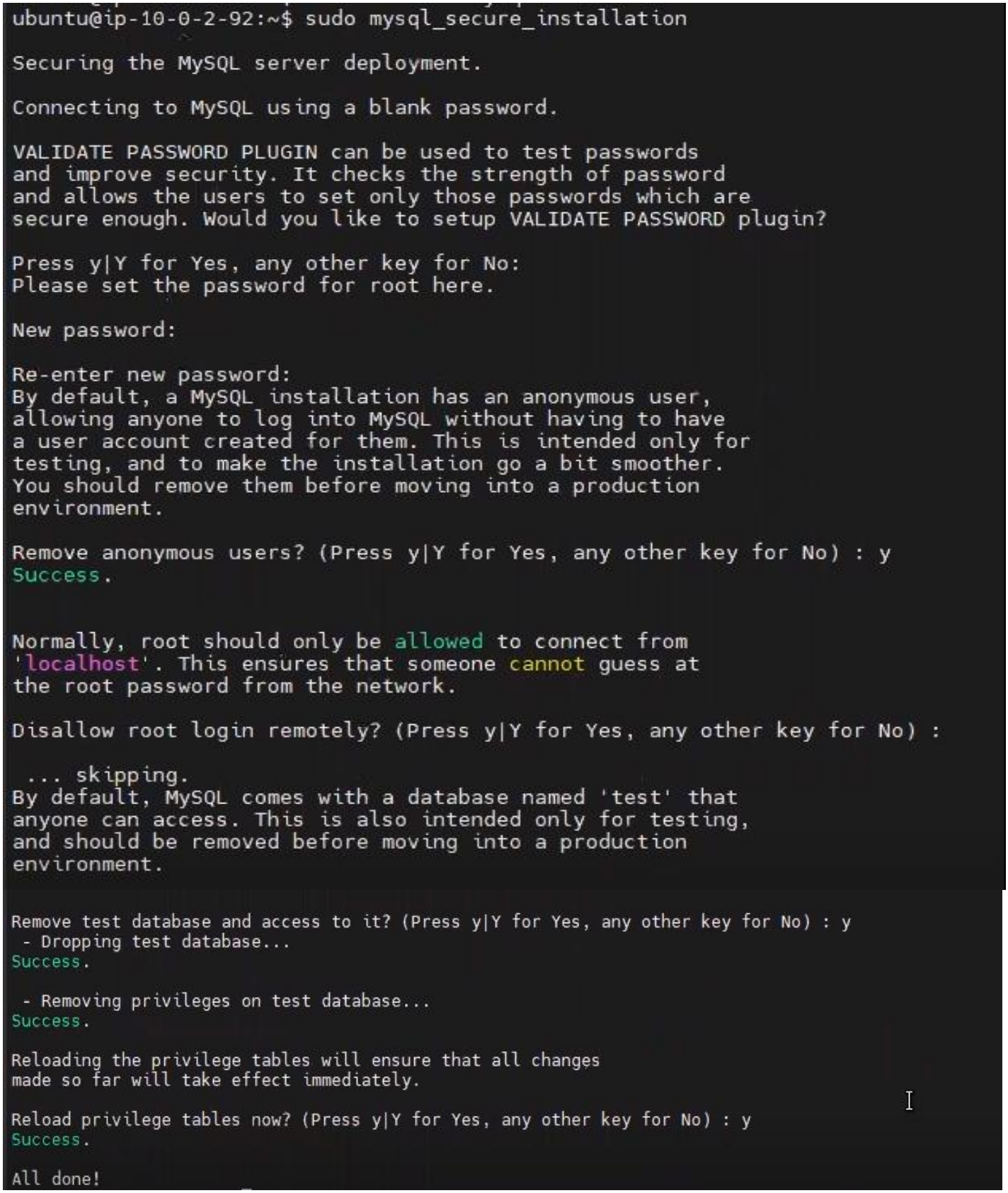
8) sudo service mysql restart



SSH into the instance which is private subnet through owncloud-Appserver instance.

```
ubuntu@ip-10-0-2-92:~$ sudo  apt-get install mysql-server -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libaio1 libcgi-fast-perl libcgi-pm-perl libencode-locale-perl libevent-core-2.1-6 libfcgi-perl libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl
  libhttp-date-perl libhttp-message-perl libio-html-perl liblwp-mediatypes-perl libtimedate-perl liburi-perl mysql-client-5.7 mysql-client-core-5.7 mysql-common
  mysql-server-5.7 mysql-server-core-5.7
Suggested packages:
  libdata-dump-perl libipc-sharedcache-perl libwww-perl mailx tinyca
The following NEW packages will be installed:
  libaio1 libcgi-fast-perl libcgi-pm-perl libencode-locale-perl libevent-core-2.1-6 libfcgi-perl libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl
  libhttp-date-perl libhttp-message-perl libio-html-perl liblwp-mediatypes-perl libtimedate-perl liburi-perl mysql-client-5.7 mysql-client-core-5.7 mysql-common
  mysql-server mysql-server-5.7 mysql-server-core-5.7
0 upgraded, 21 newly installed, 0 to remove and 52 not upgraded.
Need to get 20.1 MB of archives.
After this operation, 157 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 mysql-common all 5.8+1.0.4 [7308 B]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 libaio1 amd64 0.3.110-5ubuntu0.1 [6476 B]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 mysql-client-core-5.7 amd64 5.7.40-0ubuntu0.18.04.1 [6755 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 mysql-client-5.7 amd64 5.7.40-0ubuntu0.18.04.1 [2028 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 mysql-server-core-5.7 amd64 5.7.40-0ubuntu0.18.04.1 [7542 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libevent-core-2.1-6 amd64 2.1.8-stable-4build1 [85.9 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 mysql-server-5.7 amd64 5.7.40-0ubuntu0.18.04.1 [3006 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libhtml-tagset-perl all 3.20-3 [12.1 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 liburi-perl all 1.73-1 [77.2 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libhtml-parser-perl amd64 3.72-3build1 [85.9 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libcgi-pm-perl all 4.38-1 [185 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libfcgi-perl amd64 0.78-2build1 [32.8 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libcgi-fast-perl all 1:2.13-1 [9940 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libencode-locale-perl all 1.05-1 [12.3 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libhtml-template-perl all 2.97-1 [59.0 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libtimedate-perl all 2.3000-2 [37.5 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libhttp-date-perl all 6.02-1 [10.4 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libio-html-perl all 1.001-1 [14.9 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 liblwp-mediatypes-perl all 6.02-1 [21.7 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/main amd64 libhttp-message-perl all 6.14-1 [72.1 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates/main amd64 mysql-server all 5.7.40-0ubuntu0.18.04.1 [9944 B]
Fetched 20.1 MB in 1s (36.9 MB/s)
Preconfiguring packages ...
Selecting previously unselected package mysql-common.
(Reading database ... 85434 files and directories currently installed.)
Preparing to unpack .../0-mysql-common_5.8+1.0.4_all.deb ...
Unpacking mysql-common (5.8+1.0.4) ...
```

1) **<u>Sudo apt-get install mysql-server –y</u> :** This will show you a list of the packages that will be installed, along with the amount of disk space they'll take up.



2) When the installation is complete, run a simple security script (**sudo mysql_secure_installation**)that comes pre-installed with MySQL which will remove some dangerous defaults and lock down access to your database system

```
ubuntu@ip-10-0-2-92:~$ sudo service mysql start
ubuntu@ip-10-0-2-92:~$ sudo service mysql status
● mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
   Active: active (running) since Sat 2022-11-26 10:07:56 UTC; 3min 38s ago
  Process: 927 ExecStart=/usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/mysqld.pid (code=exited, status=0/SUCCESS)
  Process: 838 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exited, status=0/SUCCESS)
 Main PID: 930 (mysqld)
    Tasks: 27 (limit: 1134)
   CGroup: /system.slice/mysql.service
           └─930 /usr/sbin/mysqld --daemonize --pid-file=/run/mysqld/mysqld.pid

Nov 26 10:07:54 ip-10-0-2-92 systemd[1]: Starting MySQL Community Server...
Nov 26 10:07:56 ip-10-0-2-92 systemd[1]: Started MySQL Community Server.
```

Service mysql start:To start mysql service
Service mysql status:To check status of server

```
ubuntu@ip-10-0-2-92:~$ sudo mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 5.7.40-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

3) If you set up password authentication for MySQL root account, you may have to use this syntax

```
ubuntu@ip-10-0-2-92:~$ sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf
ubuntu@ip-10-0-2-92:~$ sudo service mysql restart
```

```
  GNU nano 2.9.3                                              /etc/mysql/mysql.conf.c

[mysqld_safe]
socket          = /var/run/mysqld/mysqld.sock
nice            = 0

[mysqld]
#
# * Basic Settings
#
user            = mysql
pid-file        = /var/run/mysqld/mysqld.pid
socket          = /var/run/mysqld/mysqld.sock
port            = 3306
basedir         = /usr
datadir         = /var/lib/mysql
tmpdir          = /tmp
lc-messages-dir = /usr/share/mysql
skip-external-locking
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
#bind-address           = 127.0.0.1
#
# * Fine Tuning
#
key_buffer_size         = 16M
max_allowed_packet      = 16M
thread_stack            = 192K
thread_cache_size       = 8
# This replaces the startup script and checks MyISAM tables if needed
# the first time they are touched
myisam-recover-options  = BACKUP
#max_connections        = 100
#table_open_cache       = 64
#thread_concurrency     = 10
#
# * Query Cache Configuration
#
query_cache_limit       = 1M
query_cache_size        = 16M
#
# * Logging and Replication
```

By default it doesn't allow to connect remote.Explicitly we are making it allow by commenting out bind-address by #

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| owncloud           |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

SHOW DATABASES: lists the databases on the MySQL server host

```
mysql> use mysql;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

The USE statement tells MySQL to use the named database as the default (current) database for subsequent statements.

```
mysql> create user 'admin'@'%' identified by 'admin';
Query OK, 0 rows affected (0.00 sec)
```

4) Creating admin user with password admin
% means remote hosts can login to MySQL server from any other server.

```
mysql> grant all privileges on *.* to 'admin'@'%' with grant option;
Query OK, 0 rows affected (0.00 sec)
```

5)The **asterisks** in this command refer to the database and table that they can access. This command allows to the user to read, edit,execute and perform all tasks across all the databases and tables. With **grant option** means admin user has power to give permission to other users.

```
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)
```

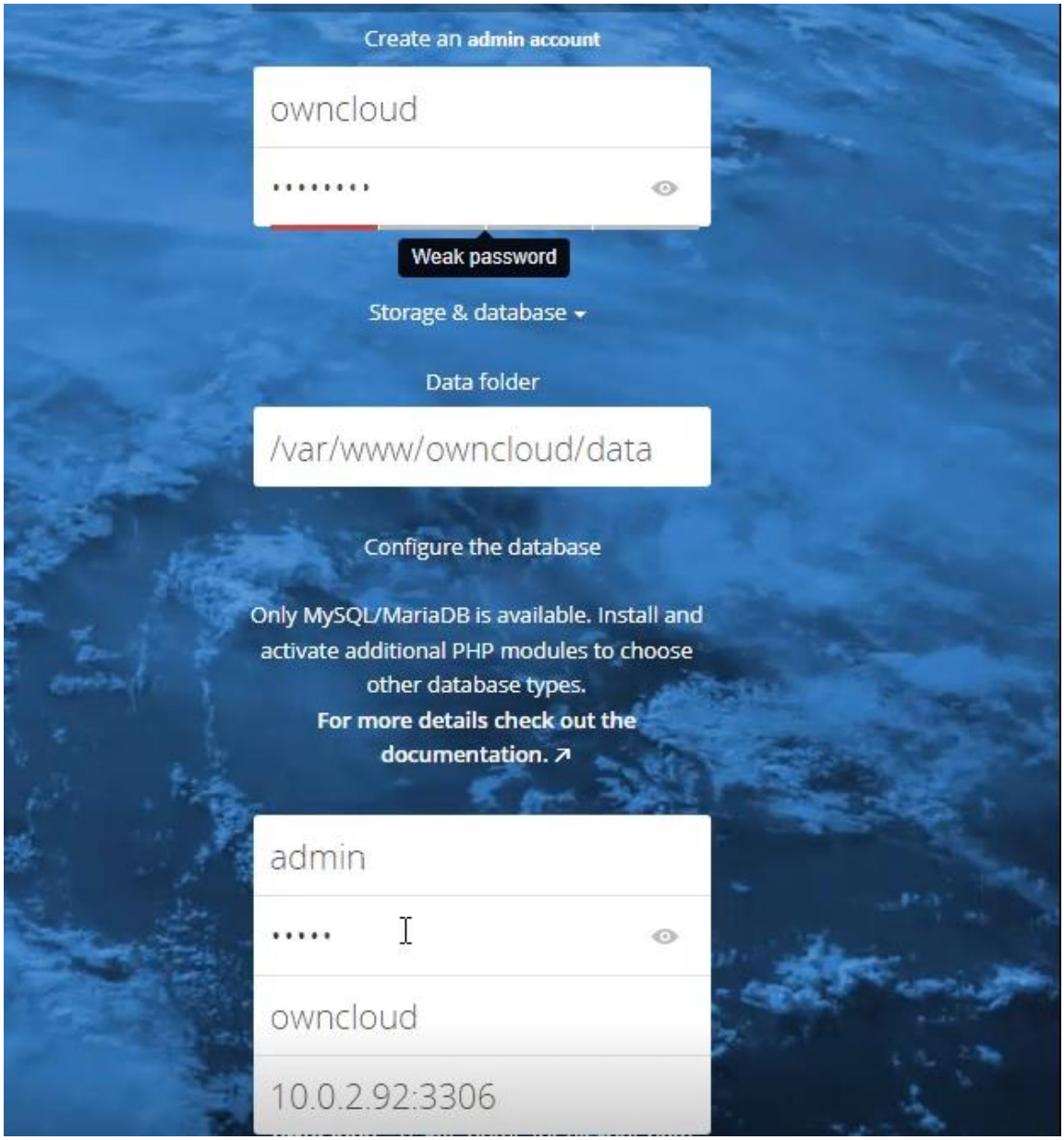
6) **flush privileges** operation to ensure that the running instance of MySQL knows about the recent privilege assignment.

```
mysql> exit;
Bye
```

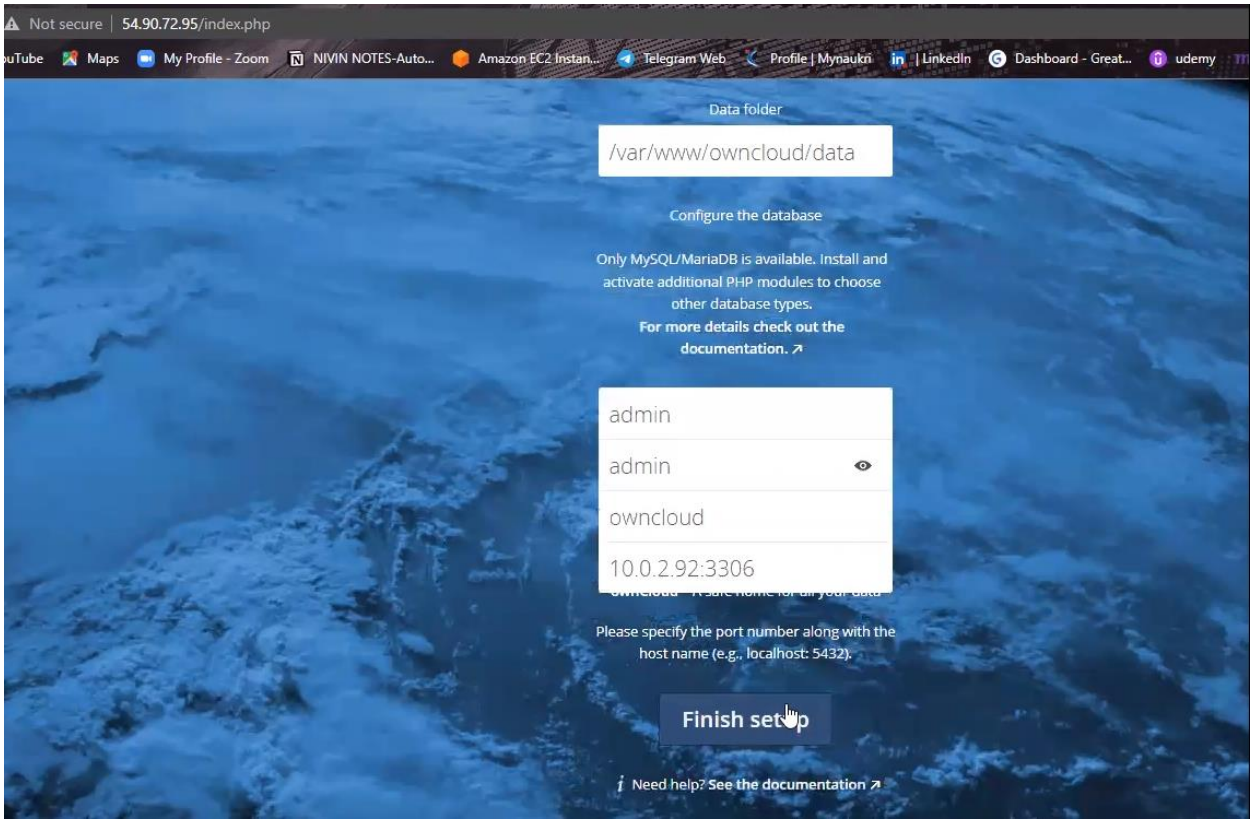
7) **exit** the MySQL session

```
ubuntu@ip-10-0-2-92:~$ sudo service mysql restart
```

8)To **restart** mysql service



Accessing the owncloud application using public ip(54.90.72.95) of EC2 instance in browser



Enter admin user and password. Enter database values.

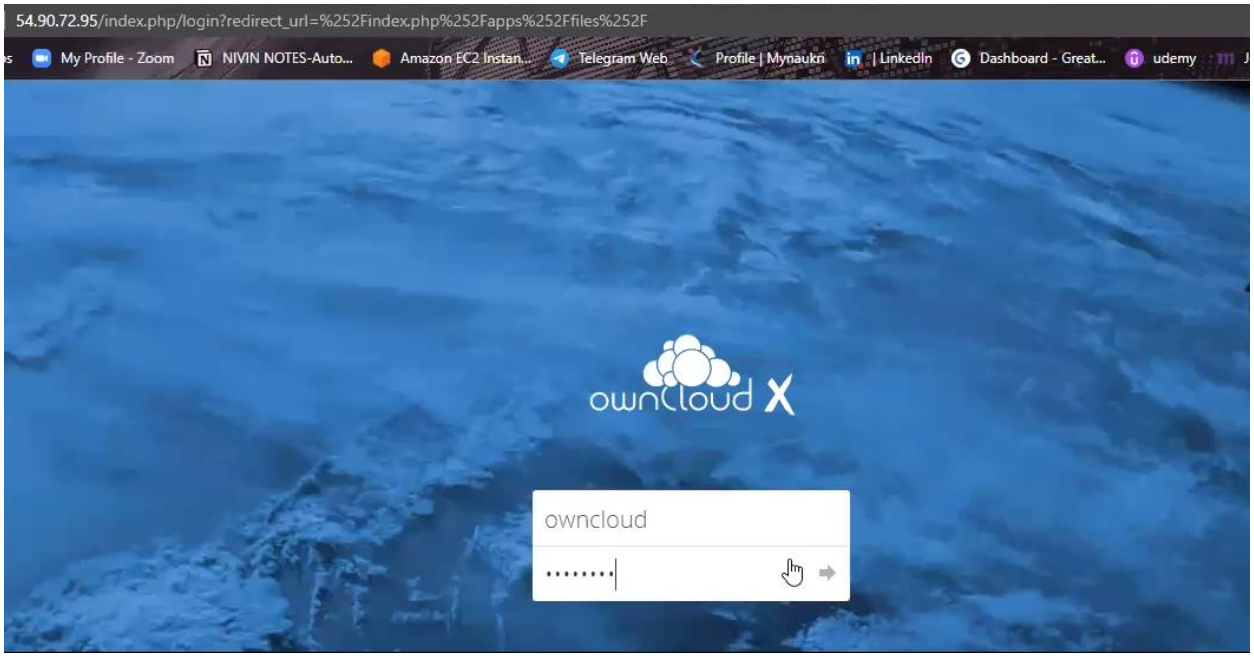
Usename : owncloud
Password : owncloud

Data folder : /var/www/owncloud/data(default)

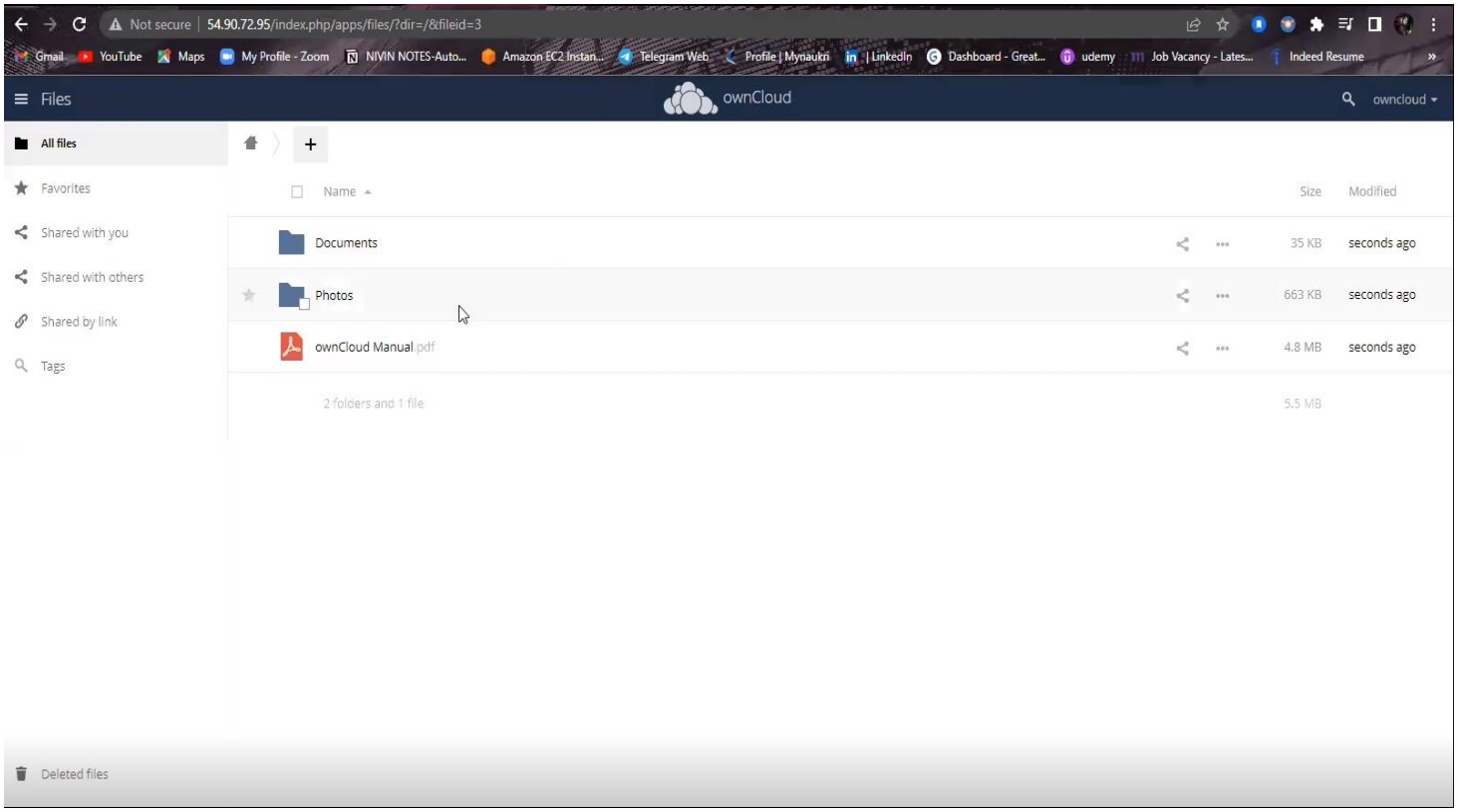
Database user : admin
Database password : admin

Database name : owncloud

Database server IP : <IP OF database server in private subnet:port>
10.0.2.92:3306



Login to owncloud using Usename : owncloud
Password : owncloud



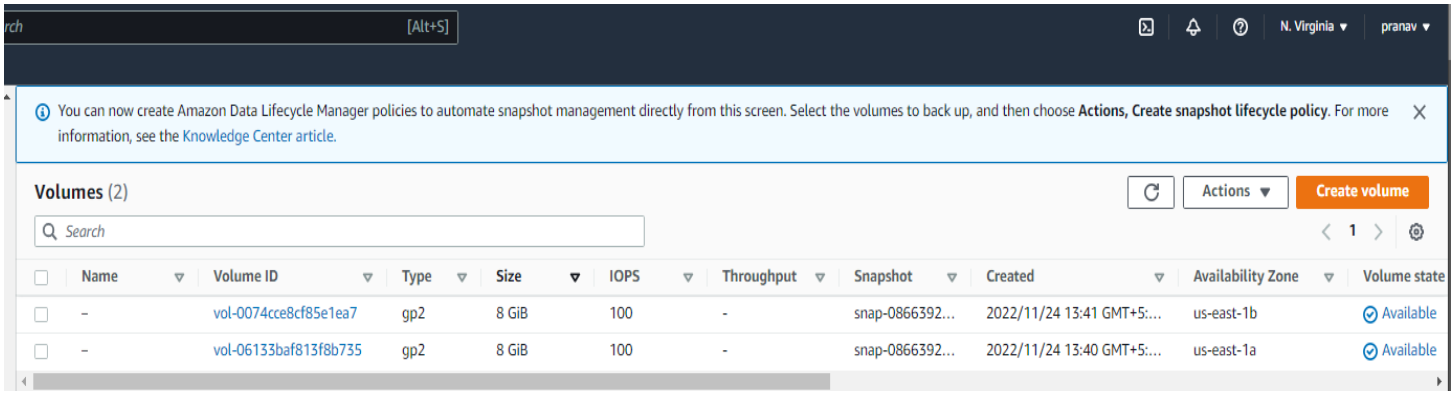
Owncloud web page

Lessons and Observations:

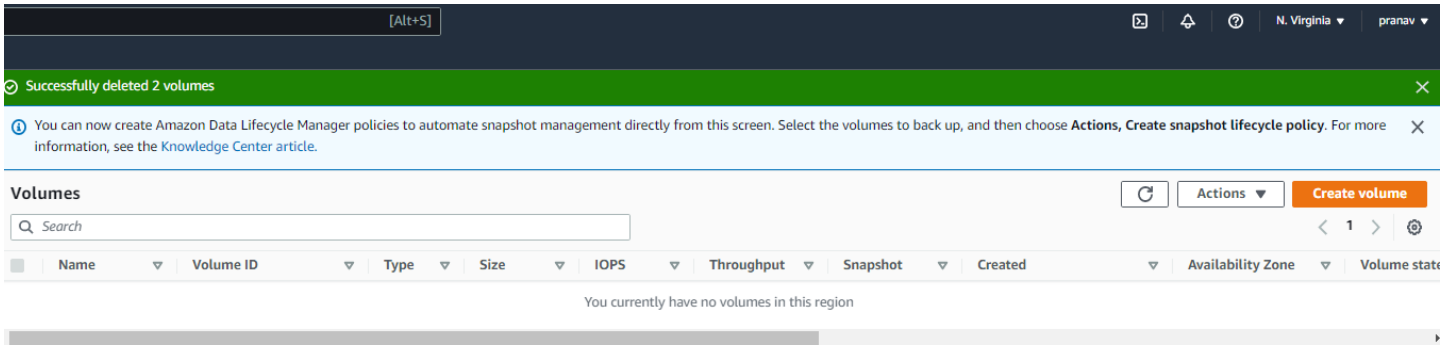
- Implemented 2 different subnets (one public and the other private) in a custom VPC(owncloud-vpc)
- Install and configure owncloud on an Ubuntu 18.04 on the public subnet.
- Created Internet gateway and Route tables.
- Configured the security groups to allow the ports 80 and 443 for source set to "Anywhere" in Owncloud Application server.
- Learnt to Install and configure MySQL on an Ubuntu 18.04 on the private subnet.
- Configured the security groups to allow the ports 80,22 and 3306 for source set to "Public Subnet" of owncloud Appserver.
- SSH 'ed into the instance which is private subnet through owncloud-Appserver instance.
- when tried to give command sudo apt-get update server was not reachable to internet so used NAT gateway on DBserver of private subnet to access internet.
- Admin is the super user for entire mysql databases .Owncloud user is a super user for only the owncloud databases
- Accessing the owncloud application using public ip(54.90.72.95) of EC2 instance in browser
- **ownCloud** is an open-source file sharing server and collaboration platform that can store your personal content, like documents and pictures, in a centralized location.
  This allows you to take control of your content and security by not relying on third-party content hosting services like Dropbox.

Resource Cleaning up!:

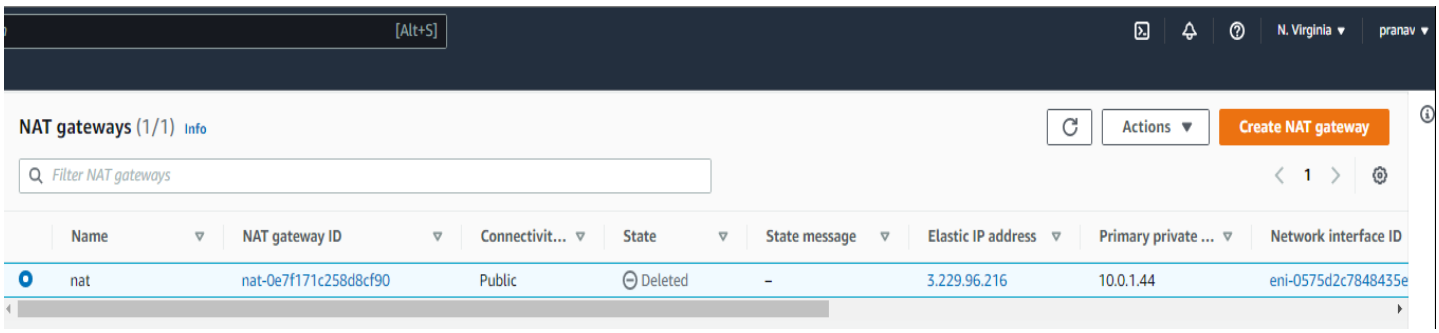
1)volume detached:



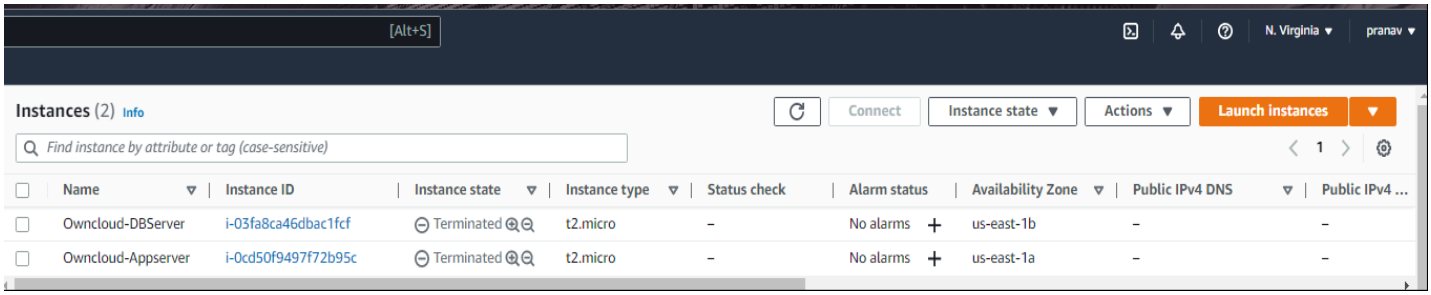
Volume deleted:



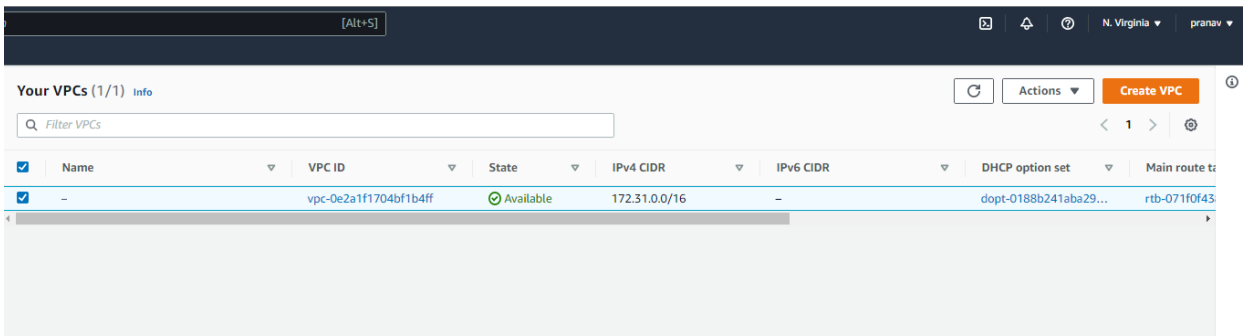
2)Nat gateway deleted



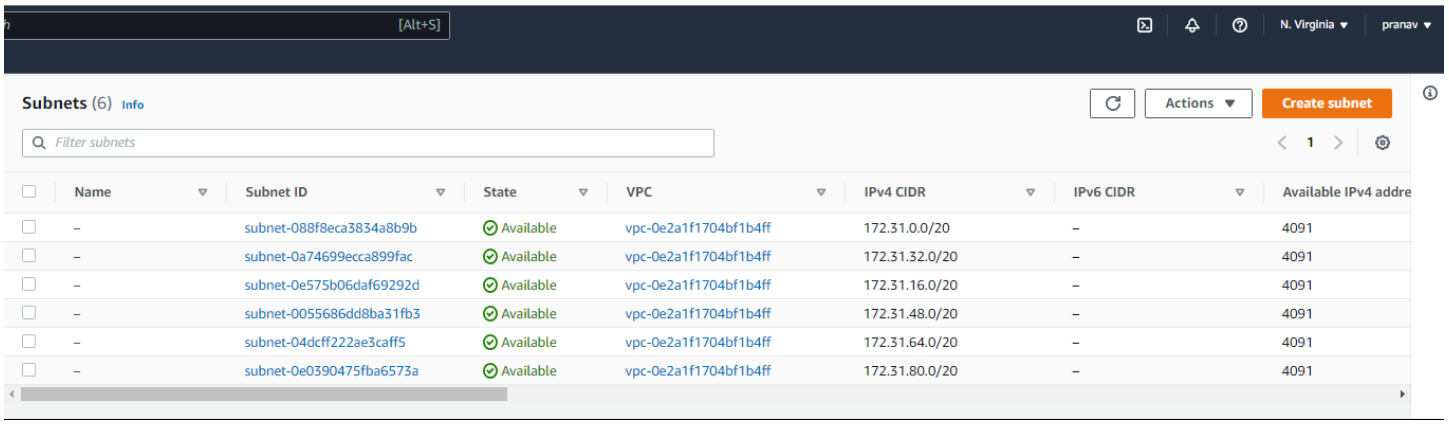
3)Terminated **owncloud-Appserver** and **owncloud-Dbserver** instances



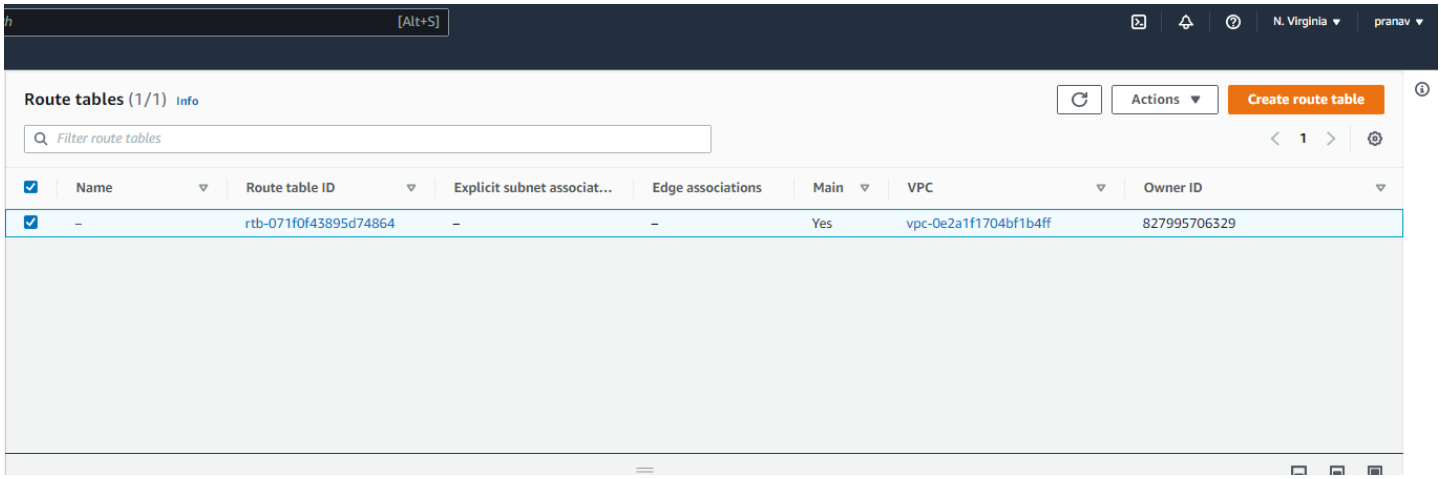
4) Deleted custom created vpc(**owncloud-vpc**) default vpc is present



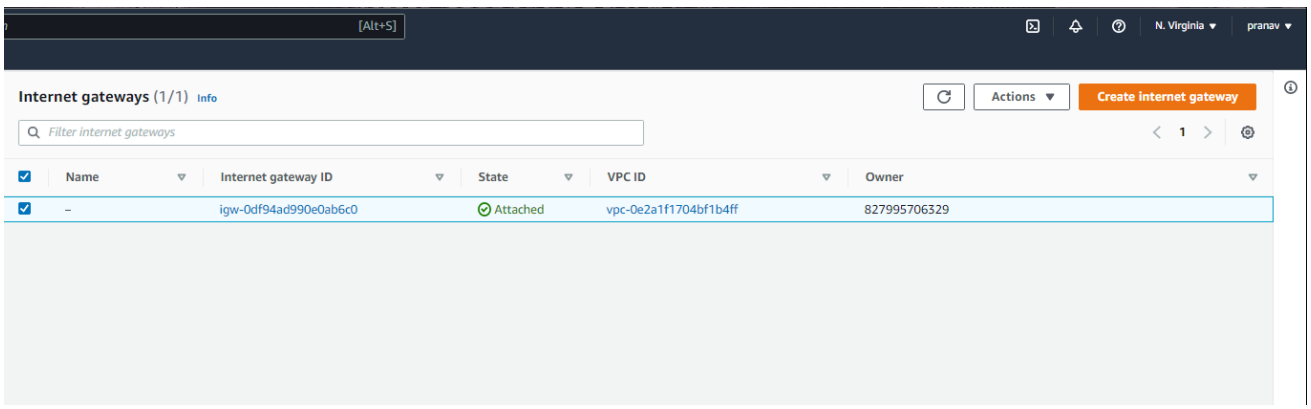
5) Subnets have been deleted(**Public Subnet and Private Subnet**)



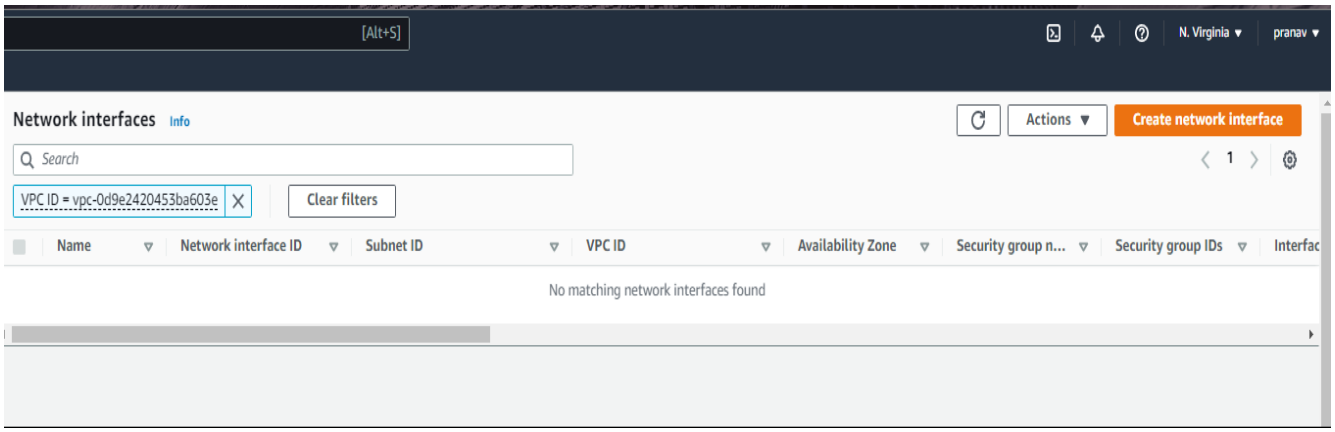
6) Route tables (Public Route Table and Private Route Table) have been deleted



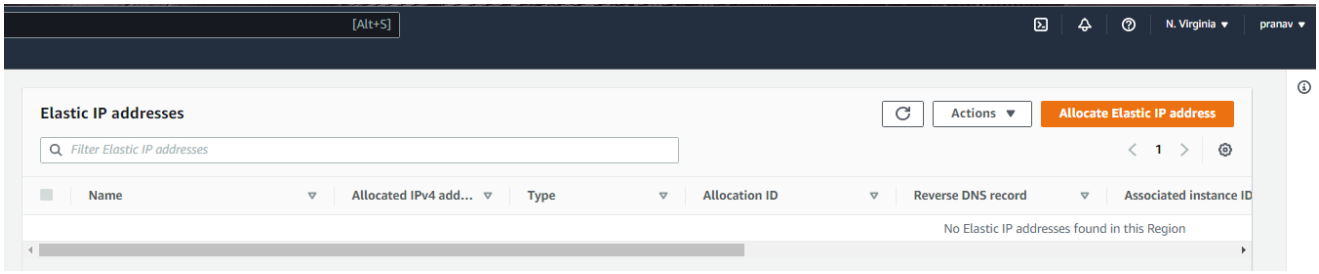
7)Deleted owncloud internet gatway. Default internet gateway is present



8)Network interface have been deleted



9)Elastic IP have been released



Note: Even when there are no instances there will be default Vpc's, Route tables,subnets,internet gate way are present.