Scenario :-

- Application Running on Physical/Virtual Machines
- Work load in your Datacentre.

Problem:-

- Complex Management
- Scale up & scale down
- Cost
- Manual Process
- Time Consuming

Solution:-

- **Cloud Setup –**

  Pay-As-You-go, IAAS, Flexibility, Ease of Infra Management

- **AWS Service:-**

  EC2 Instances

  ELB

  Autoscaling

  S3

  Route 53

- **Objective:-**

  Flexible Infra

  No Upfront Cost

  Modernize Effectively

  IAAC

Architecture of AWS Setup for Web Application:-

EC2 Instances, ELB, Autoscaling, EFS/S3 for shared storage, IAM Role

Users -> ELB -> Autoscaling Group -> Route 53

Flow of Execution:-

1. Login to AWS Account
2. Create Key Pair
3. Create Security Groups
4. Launch Instances with user data (Bash Scripts)
5. Update IP to name mapping in route 53
6. Build Application from source code
7. Upload to S3 bucket
8. Download Artifact to Tomcat Ec2 instance
9. Setup ELB with HTTPS[Cert from Amazon Certificate Manager]
10. Build Autoscaling Group for Tomcat Instances.

Prerequisite to start the project:-

Clone this below Repository in your system:-

https://github.com/Hussain147/webapp-on-aws.git

Install Chocolatey in your local machine with powershell

https://chocolatey.org/install#individual

AWS Account

1. Login to AWS Account:-

- North Virginia Region

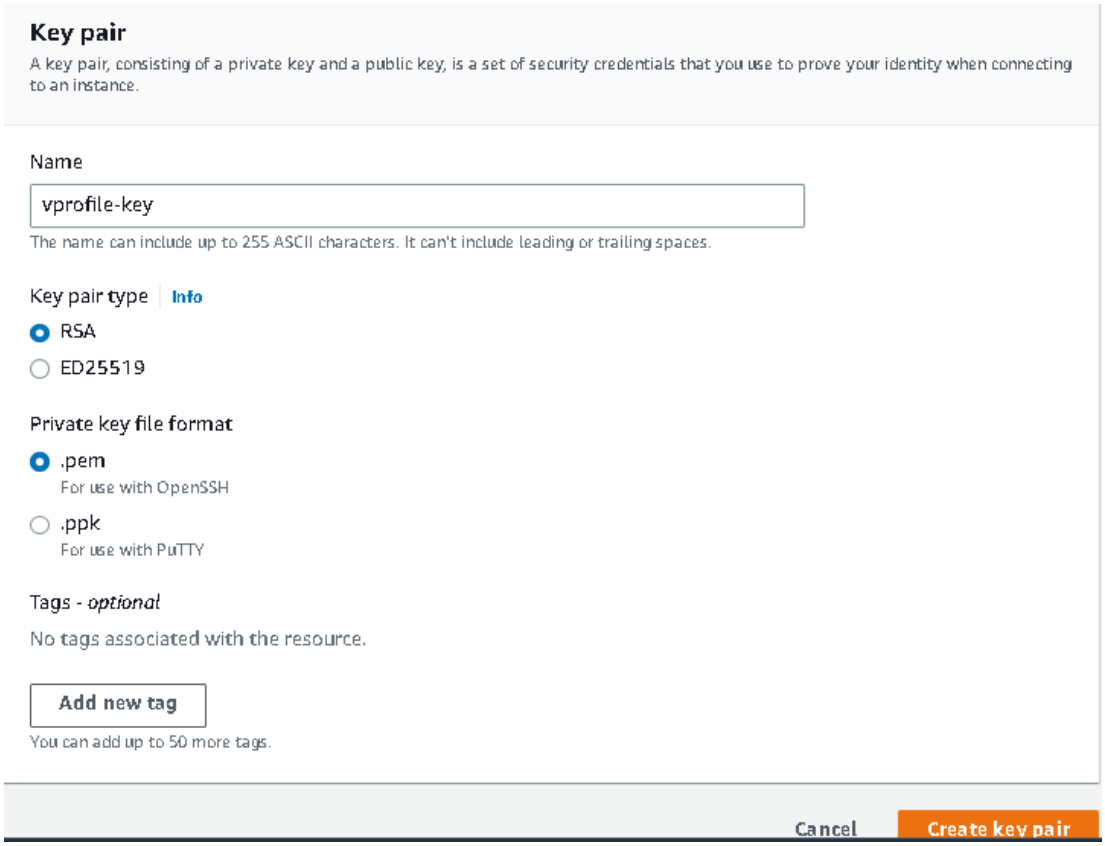
2. Create Key Pair:-

Goto Key pair > click create key pair >

Name : vprofile-key

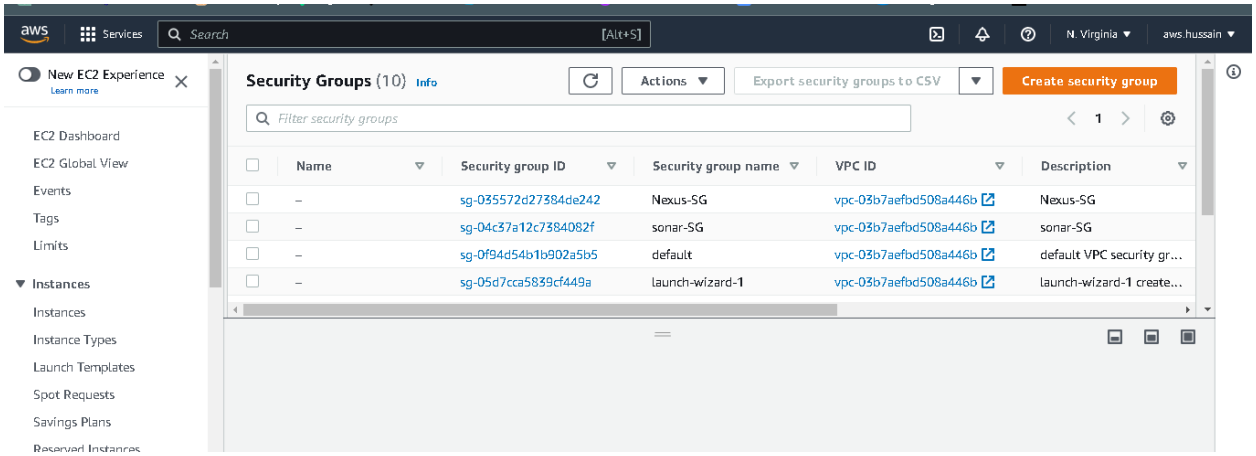
Key format : .pem

Click Create key Pair.



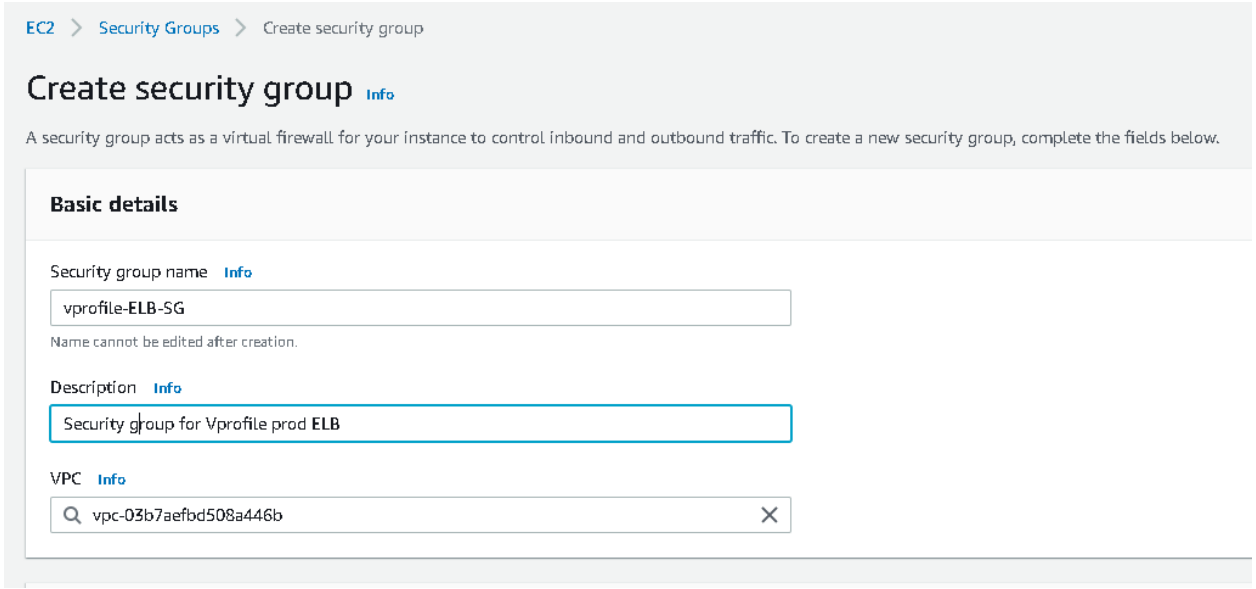
3. Create Security Groups:-

Go to Security group > create security group for ELB >



Click Create Security Group >

Name : vprofile-ELB-SG



Allow the Security group rules as follows:-

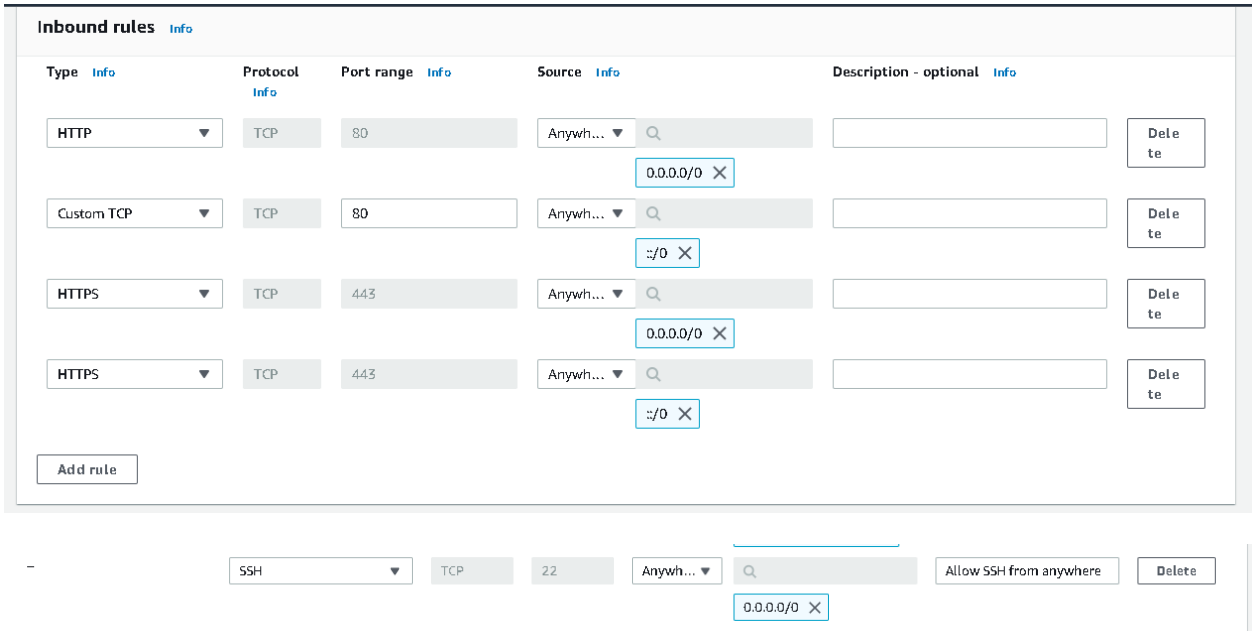
HTTP : 80 : Anywhere Ipv4

Custom TCp : 80 : anywhere Ipv6

HTTPS : 443 : Anywhere Ipv4

HTTPS : 443 : Anywhere Ipv6

Ssh : 80 : Anywhere from ipv4



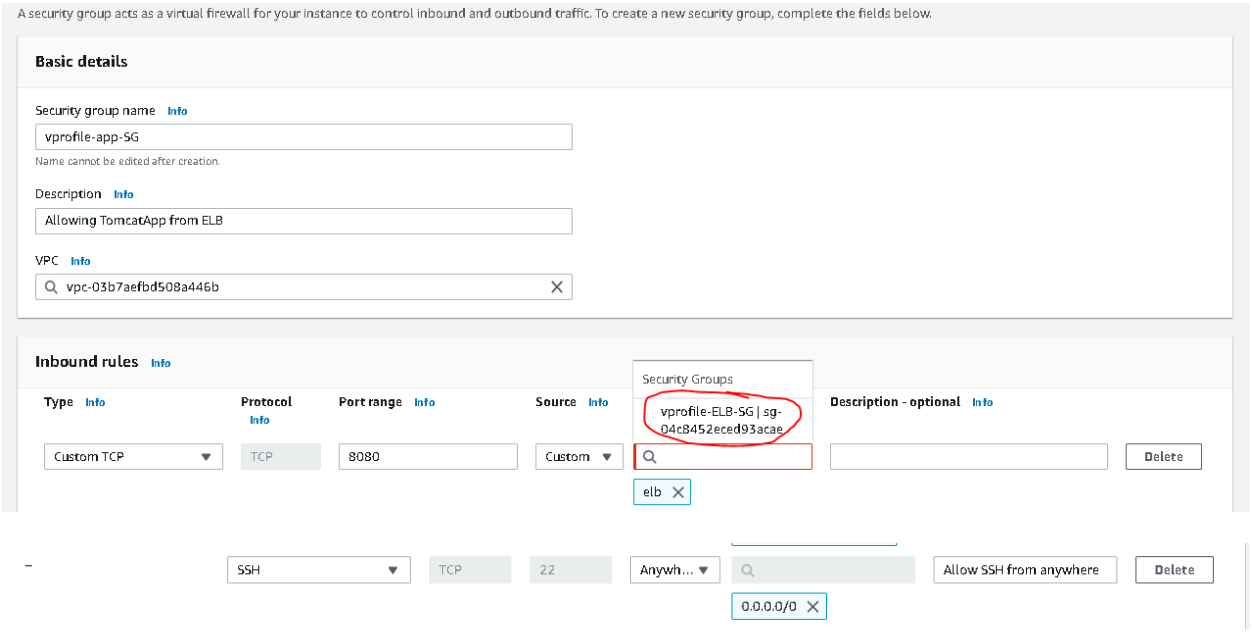
Next we are going to create security group for Tomcat App:-

Click Create Security group > Name : vprofile-app-sg

Allow these rules :-

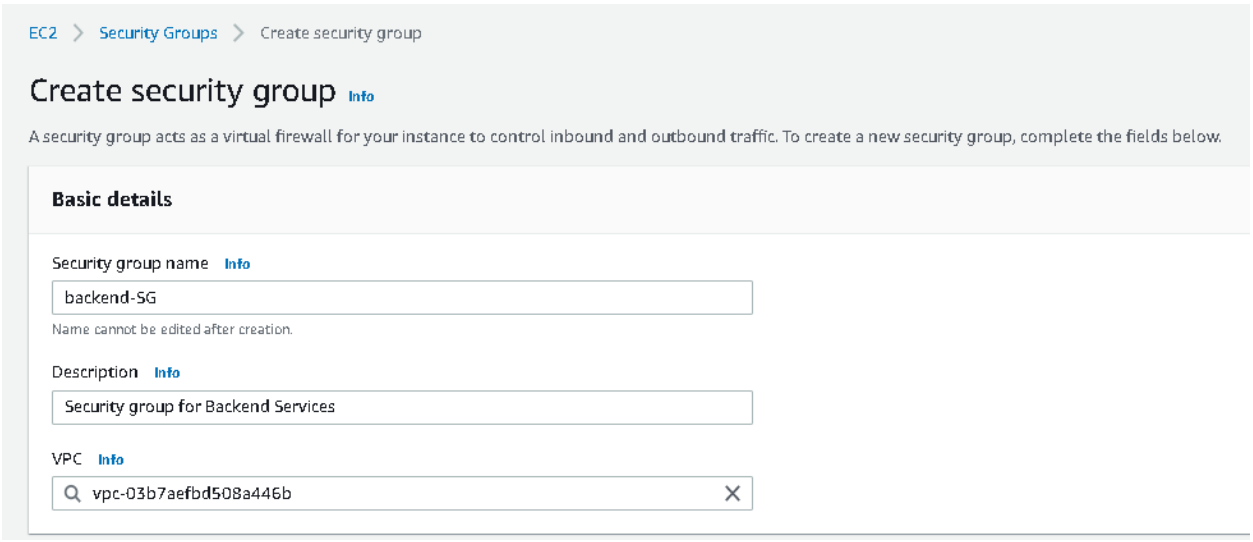
Cusom TCP : 8080 : Allow from ELB SG (type ELB in search bar, you will get the SG id)

Ssh : 80 : Anywhere from ipv4



Now Create a another security group for backend services like RabbitMQ, MemcacheD, 7 MySQL

Click New Security Group > Name : backend services



In inbound Rules > Give these below rules

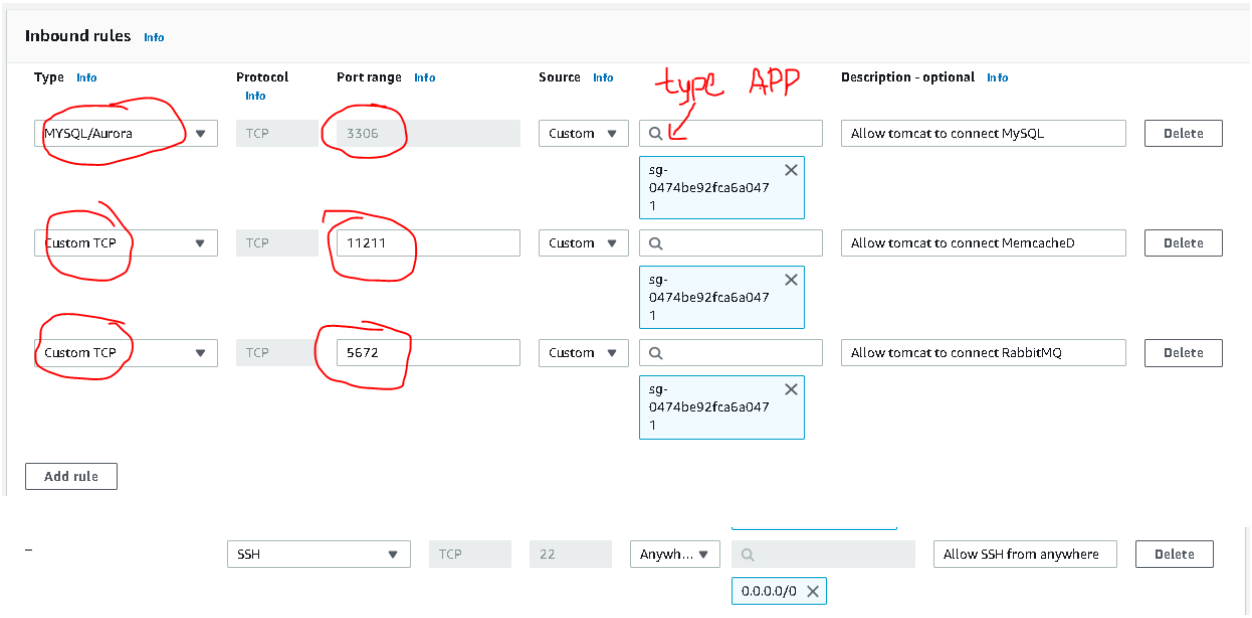
Note :- MySQL Works on port 3306. MemcacheD works on port 11211. RabbitMQ works on 5672

MySQL : 3306 : type app in search bar, you will get tomcat app SG, Then select it.

Custom TCP : 11211 : select tomcat app SG

Custom TCP : 5672 : selct tomcat app SG

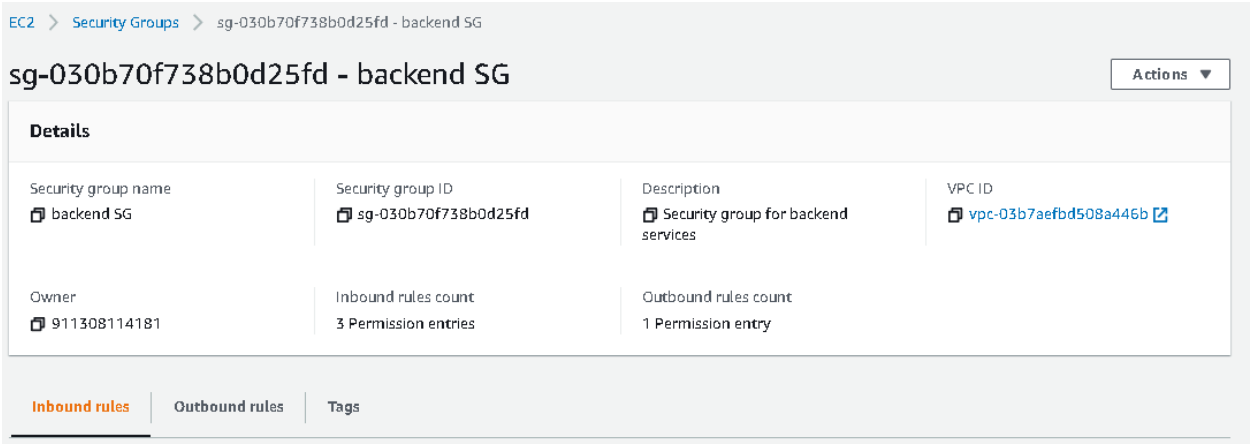
Ssh : 80 : Anywhere from ipv4



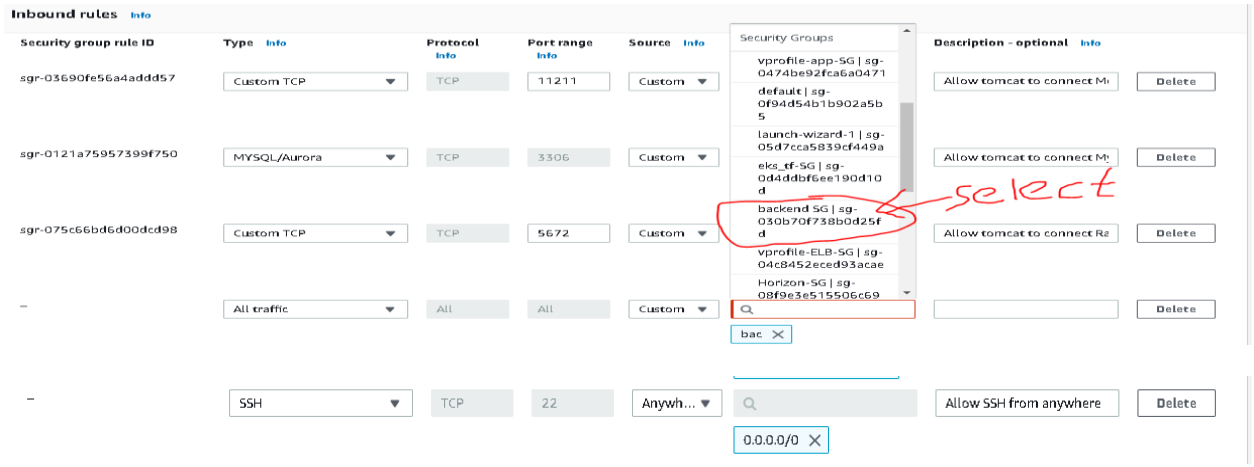
Click Create

Again select our backend security group >

Add another rule which is backend SG itself, so that our backend services will communicate internally with each other.



Edit Inbound Rules >



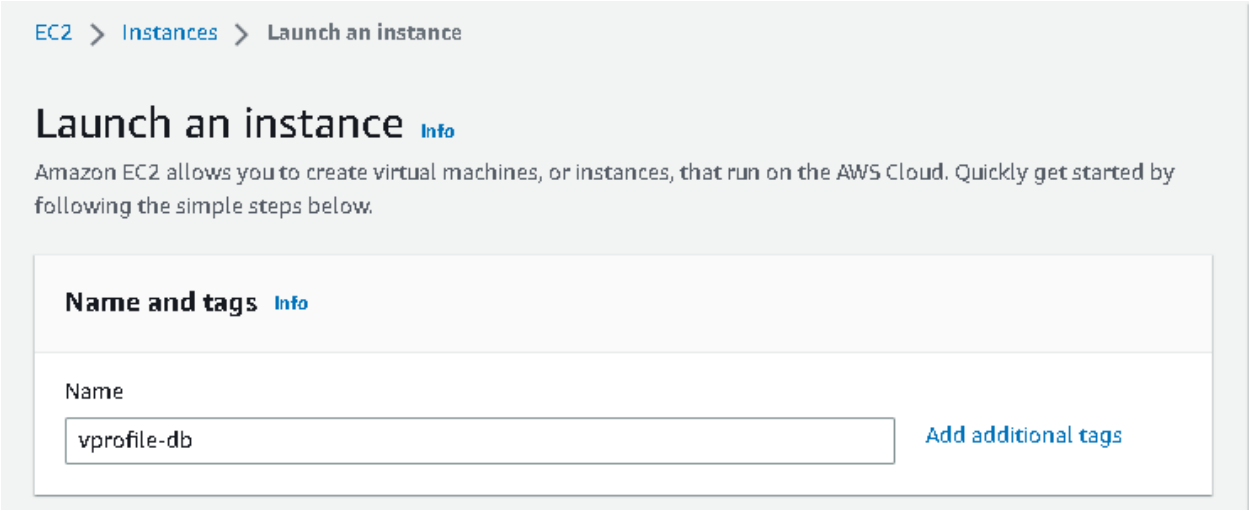
4. Launch Instances with user data (Bash Scripts):-

Now Launch Instance as mentioned below :-

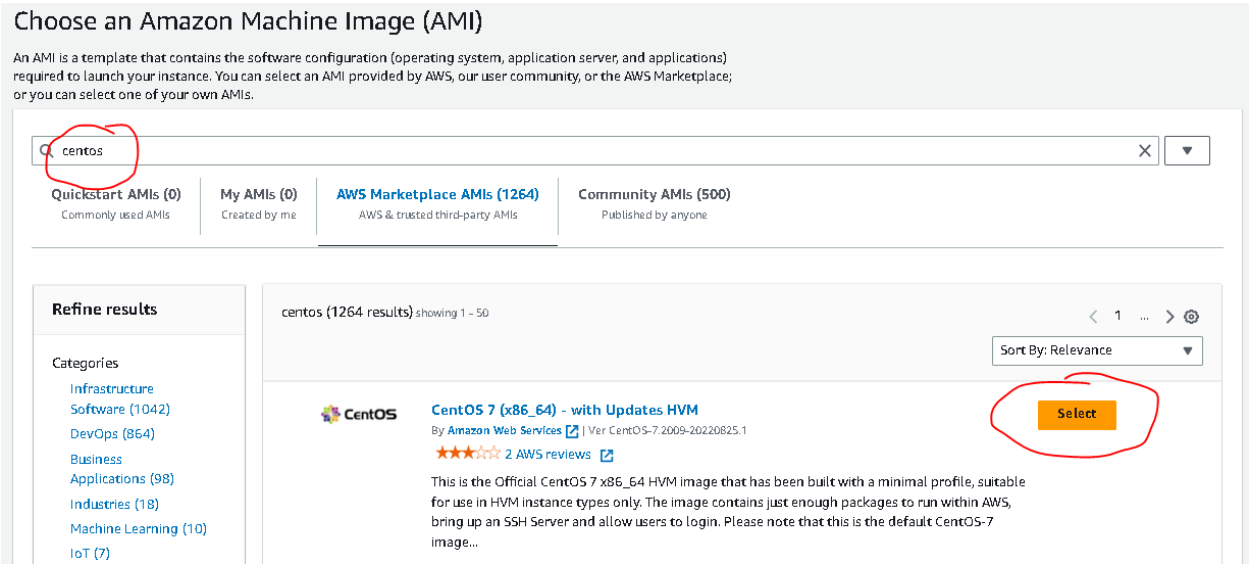
Launch CentOS Instance for MySQL

Launch CentOS Instance for MySQL

Launch an instance for Database-> Name : vprofile-db

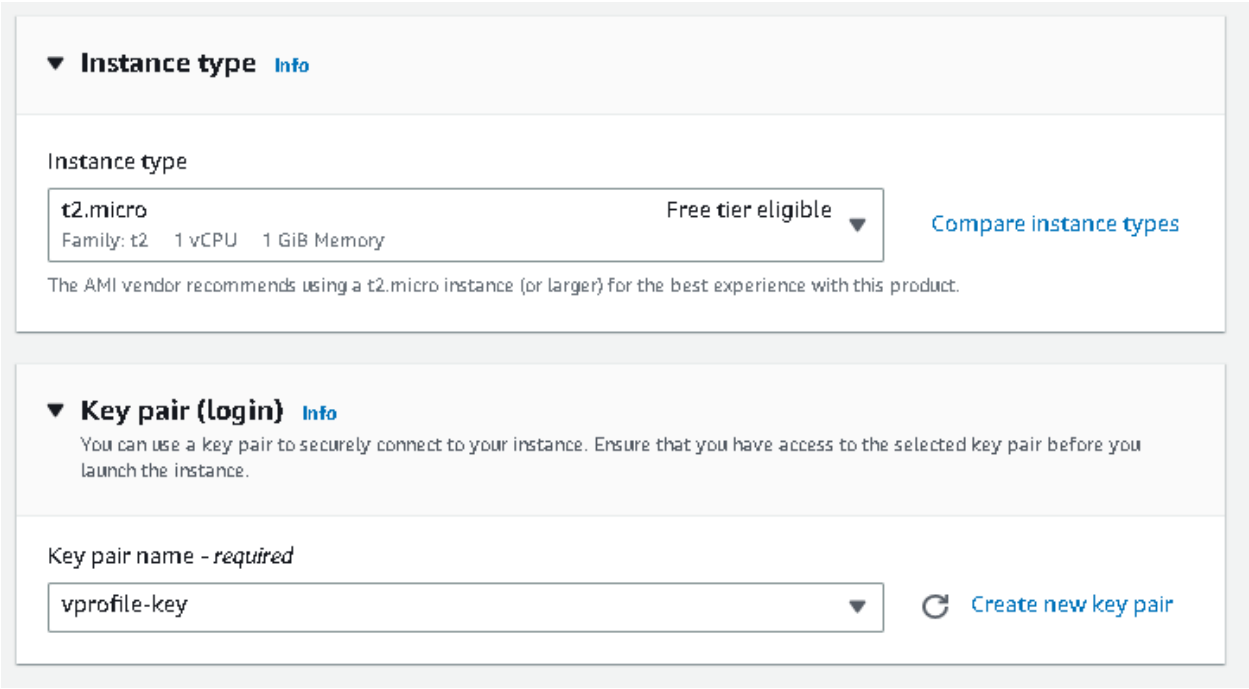


Choose AMI CentOS 7

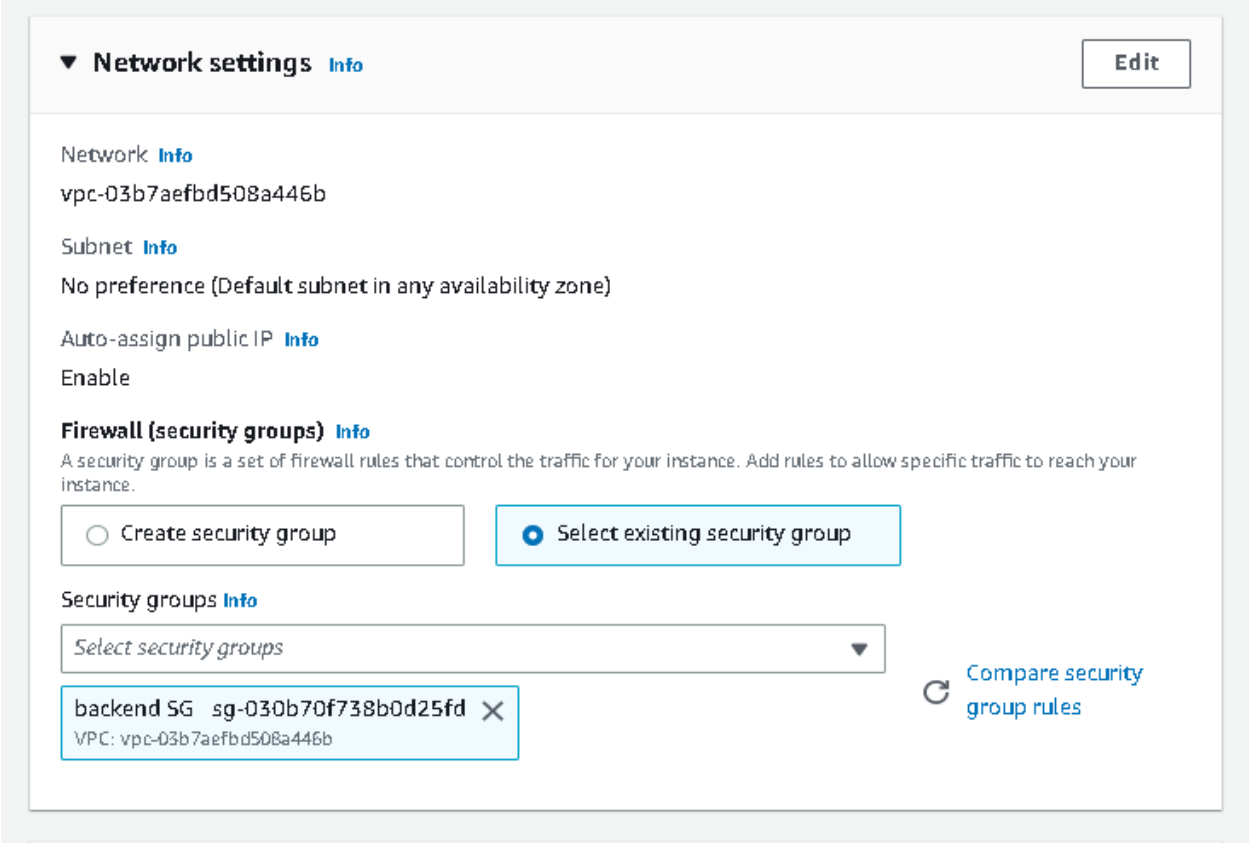


Select instance type : t2 micro

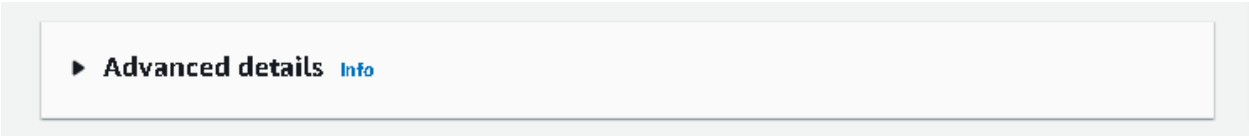
Select Key pair : vprofile-key



Select Existing Security Group : backend SG



Click on Advanced dateils > Go to User Data



Paste the MySQL user data from the given link :

https://github.com/Hussain147/webapp-on-aws/blob/main/userdata/mysql.sh

User data - *optional* **Info**

Enter user data in the field.

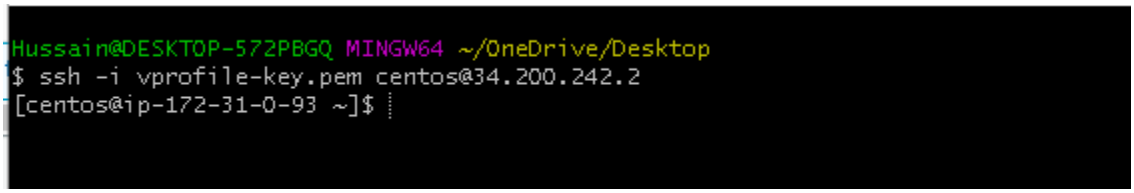
```
#!/bin/bash
DATABASE_PASS='admin123'
sudo yum update -y
sudo yum install epel-release -y
sudo yum install git zip unzip -y
sudo yum install mariadb-server -y


# starting & enabling mariadb-server
sudo systemctl start mariadb
sudo systemctl enable mariadb
cd /tmp/
git clone -b vp-rem https://github.com/devopshydclub/vprofile-repo.git
#restore the dump file for the application
sudo mysqladmin -u root password "$DATABASE_PASS"
```

☐ User data has already been base64 encoded

Then Click Launch Instance

Do SSH to the launched instance >

ssh –i vprofile-key.pem centos@ipv4 public address



Now, check the status of MySQL whether it is active or not:-

systemctl status mariadb

```
[centos@ip-172-31-0-93 ~]$ systemctl status mariadb
● mariadb.service - MariaDB database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; vendor pres
et: disabled)
   Active: active (running) since Tue 2023-02-28 11:21:54 UTC; 2h 5min ago
  Process: 16809 ExecStartPost=/usr/libexec/mariadb-wait-ready $MAINPID (code=ex
ited, status=0/SUCCESS)
  Process: 16774 ExecStartPre=/usr/libexec/mariadb-prepare-db-dir %n (code=exite
d, status=0/SUCCESS)
 Main PID: 16808 (mysqld_safe)
   CGroup: /system.slice/mariadb.service
           ├─16808 /bin/sh /usr/bin/mysqld_safe --basedir=/usr
           └─16973 /usr/libexec/mysqld --basedir=/usr --datadir=/var/lib/mysq...

Feb 28 11:21:52 ip-172-31-0-93.ec2.internal systemd[1]: Starting MariaDB data...
Feb 28 11:21:52 ip-172-31-0-93.ec2.internal mariadb-prepare-db-dir[16774]: Da...
Feb 28 11:21:53 ip-172-31-0-93.ec2.internal mysqld_safe[16808]: 230228 11:21:...
Feb 28 11:21:53 ip-172-31-0-93.ec2.internal mysqld_safe[16808]: 230228 11:21:...
Feb 28 11:21:54 ip-172-31-0-93.ec2.internal systemd[1]: Started MariaDB datab...
Hint: Some lines were ellipsized, use -l to show in full.
[centos@ip-172-31-0-93 ~]$
```

Validate the Database ->

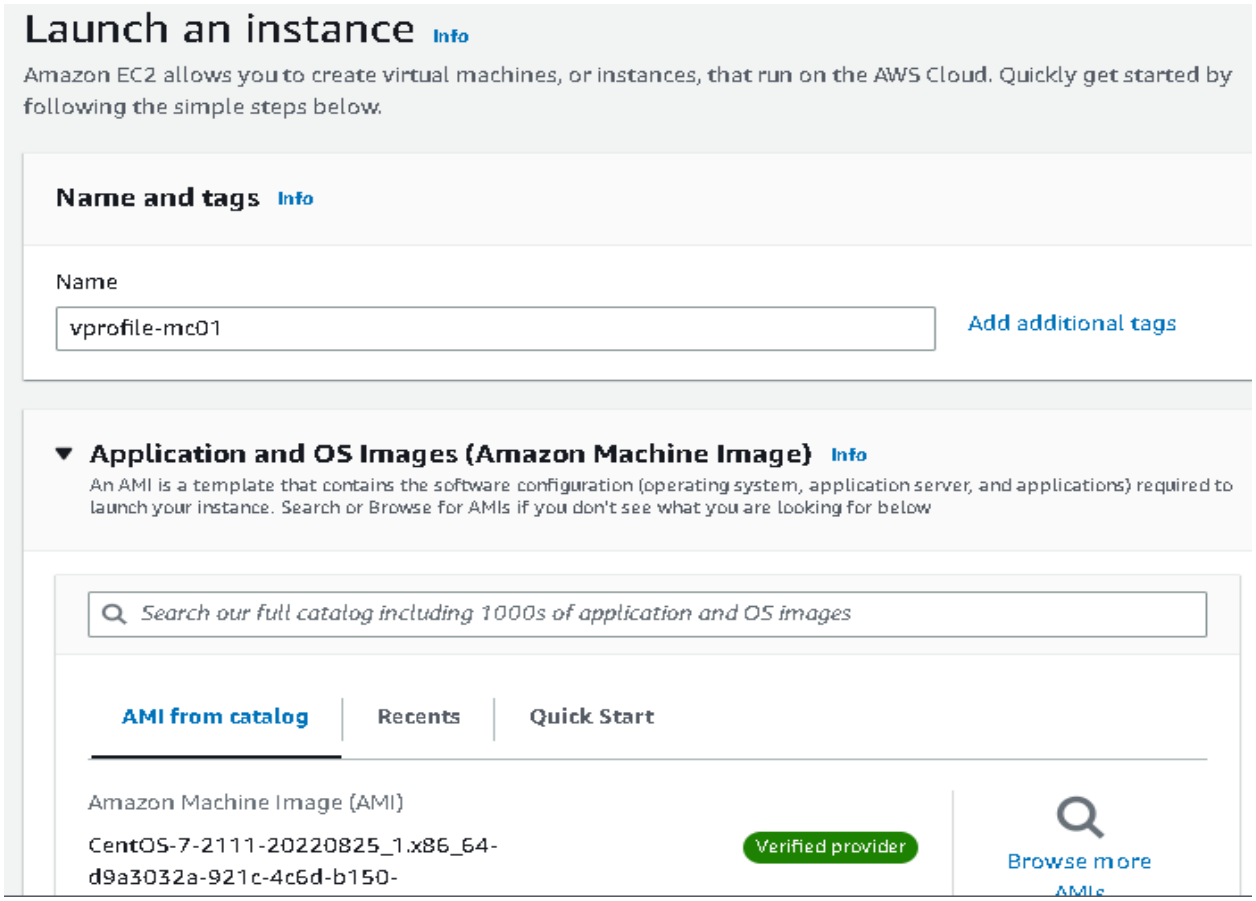
Mysql –u root –p

It will ask password -> Password : admin

show databases;

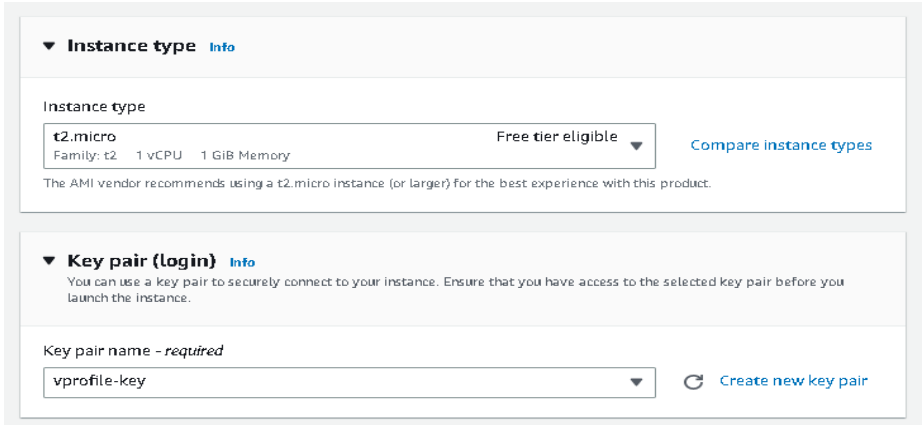
```
[centos@ip-172-31-0-93 ~]$ mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 6
Server version: 5.5.68-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| accounts           |
| mysql              |
| performance_schema |
| test               |
+--------------------+
5 rows in set (0.00 sec)
```

Now Go to AWS EC2 console > launch another instance for MemcacheD with CentOs

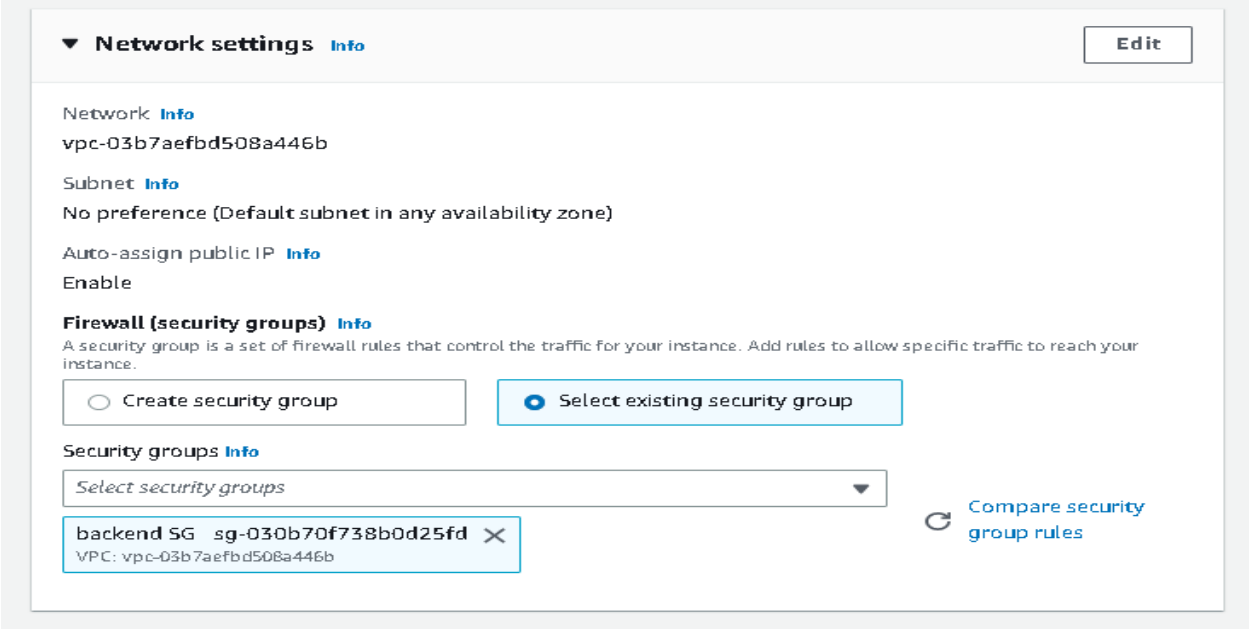


Instance Type : t2 micro

Key Pair : vprofile-key

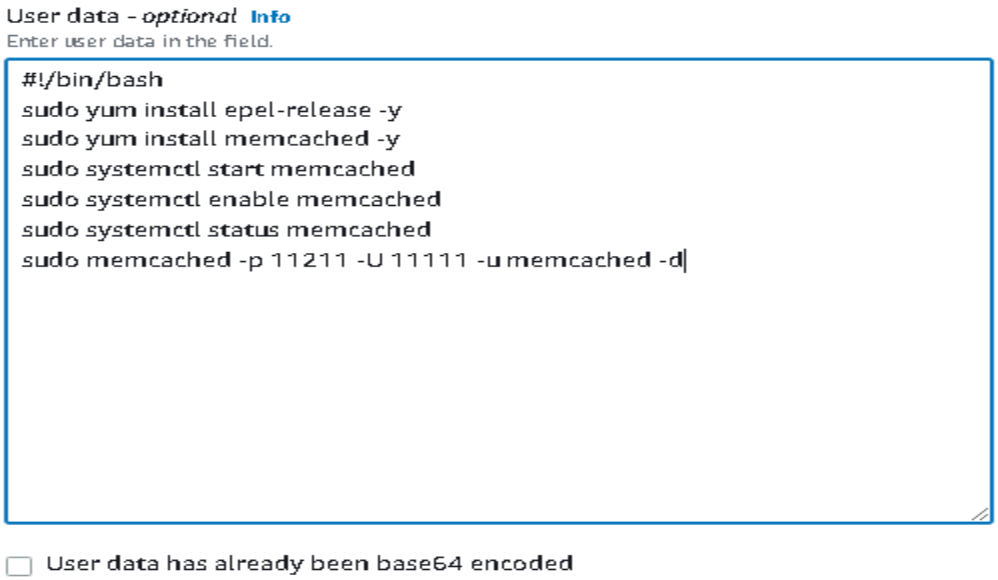


Security Group: backend SG



Paste the **MemcacheD** user data from the given link :

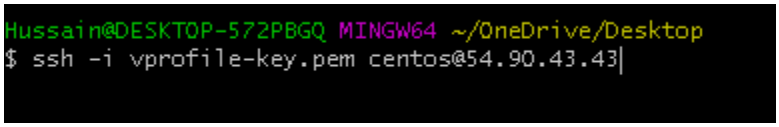
https://github.com/Hussain147/webapp-on-aws/blob/main/userdata/memcache.sh



Click Launch Instance

Now, do ssh :-

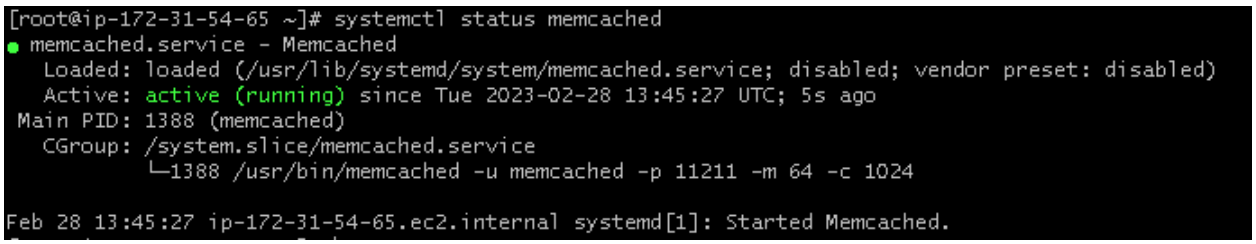
ssh –I vprofile-key.pem centos@ipv4 pub address



Wait for sometime to start our memcached service(it takes 2-5 minutes to start)

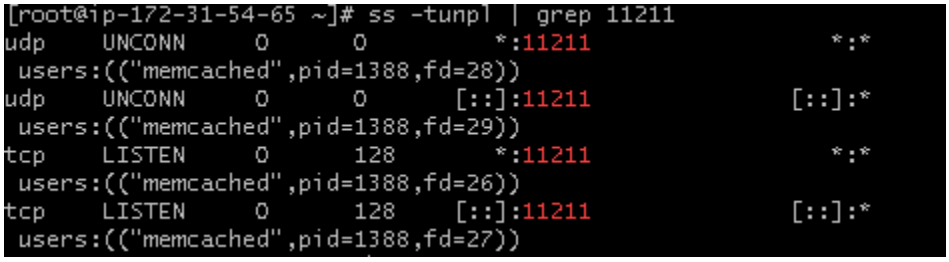
Now, Check the status of memcached:-

systemctl status memcached



Check whether it is running on right port:

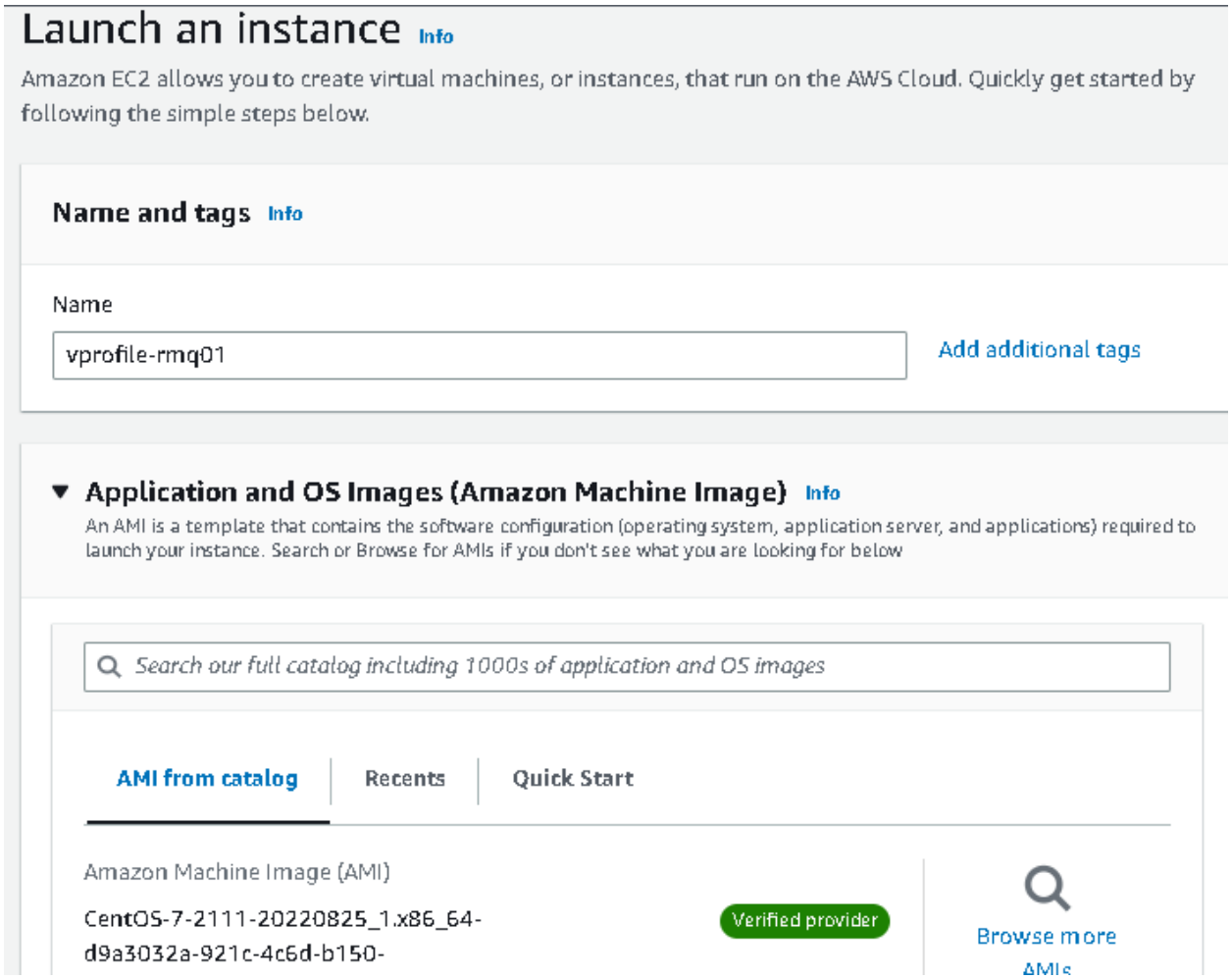
ss –tunpl | grep 11211



Now Go to AWS EC2 Console & launch another instance for RabbitMQ :

Name: vprofile-rmq01

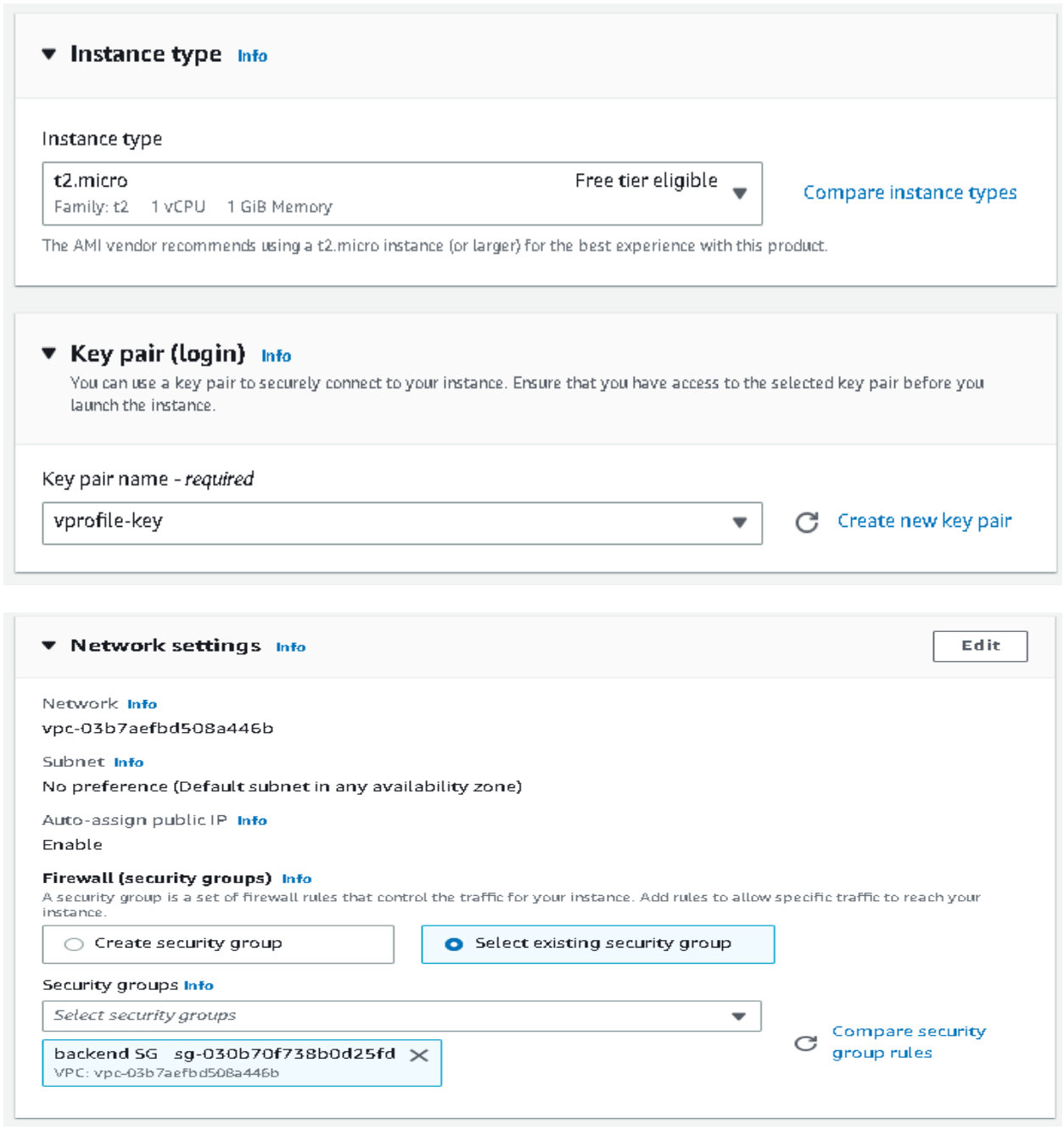
OS; CentOS



Instance Type : t2 micro

Key Pair: vprofile-key

Security Group : backend SG



Paste the **RabbitMQ** user data from the given link :

https://github.com/Hussain147/webapp-on-aws/blob/main/userdata/rabbitmq.sh

User data - *optional* Info

Enter user data in the field.

```
#!/bin/bash
sudo yum install epel-release -y
sudo yum update -y
sudo yum install wget -y
cd /tmp/
wget http://packages.erlang-solutions.com/erlang-solutions-2.0-1.noarch.rpm
sudo rpm -Uvh erlang-solutions-2.0-1.noarch.rpm
sudo yum -y install erlang socat
curl -s https://packagecloud.io/install/repositories/rabbitmq/rabbitmq-
server/script.rpm.sh | sudo bash
sudo yum install rabbitmq-server -y
sudo systemctl start rabbitmq-server
sudo systemctl enable rabbitmq-server
sudo systemctl status rabbitmq-server
sudo sh -c 'echo "[{rabbit, [{loopback_users, []}]}]." >
/etc/rabbitmq/rabbitmq.config'
sudo rabbitmqctl add_user test test
sudo rabbitmqctl set_user_tags test administrator
sudo systemctl restart rabbitmq-server
```

Click Launch Instance

Connect with ssh:

ssh –i vprofile-key.pem centos@ipv4 pub address

systemctl status rabbitmq-server

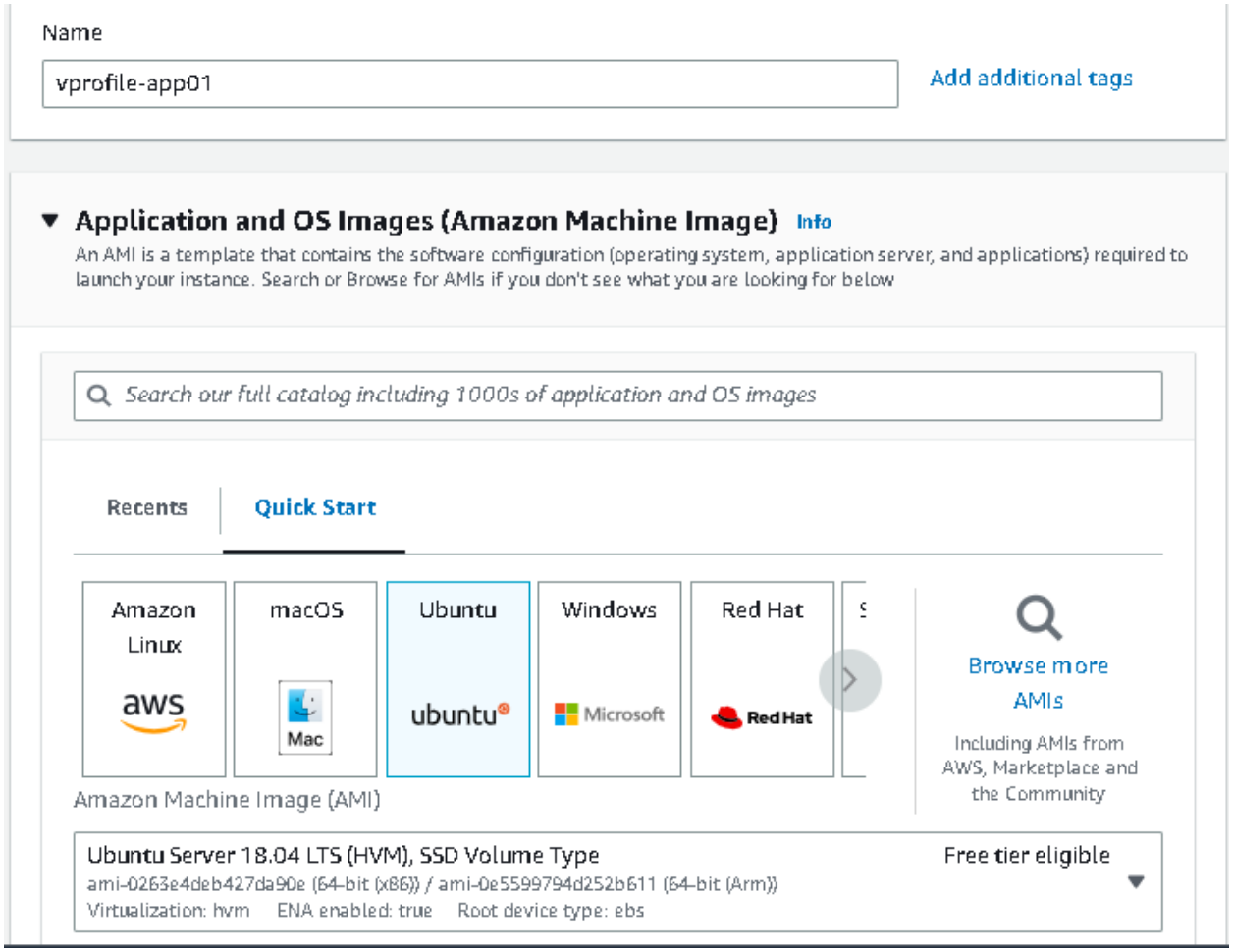
```
[root@ip-172-31-56-24 ~]# systemctl status rabbitmq-server
● rabbitmq-server.service - RabbitMQ broker
   Loaded: loaded (/usr/lib/systemd/system/rabbitmq-server.service; enabled; vendor preset: dis
abled)
   Active: active (running) since Tue 2023-02-28 14:05:33 UTC; 1s ago
  Process: 17640 ExecStop=/usr/sbin/rabbitmqctl shutdown (code=exited, status=0/SUCCESS)
 Main PID: 17676 (beam.smp)
   CGroup: /system.slice/rabbitmq-server.service
           ├─17676 /usr/lib64/erlang/erts-12.3.2.1/bin/beam.smp -W w -MBas ageffcbf -MHas ag...
           ├─17691 erl_child_setup 32768
           ├─17714 /usr/lib64/erlang/erts-12.3.2.1/bin/epmd -daemon
           ├─17731 inet_gethost 4
           └─17732 inet_gethost 4
```

Now, Launch another server for Tomcat Application:

Go to AWS EC2 Console > Launch instance:

Name: vprofile-app01

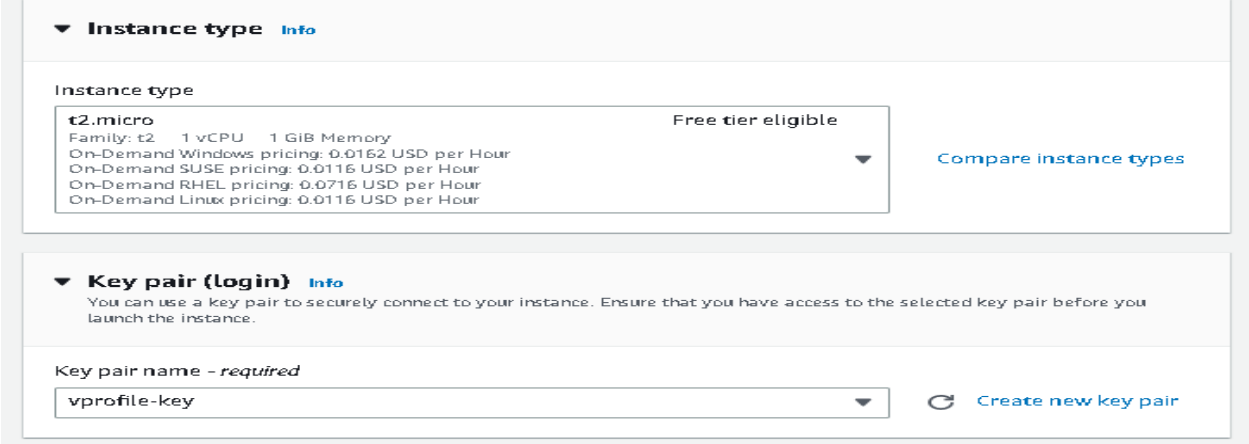
OS: Ubuntu Server 18.04 LTS

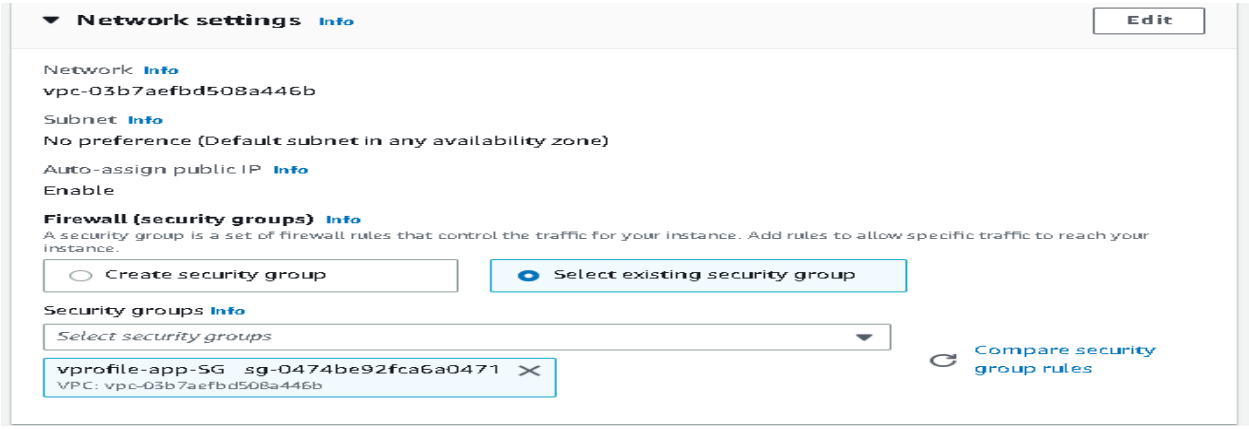


Instance Type: t2 micro

Key Pair: vprofile-key

Security Group: vprofile-app-SG





Paste the **Tomcat** user data from the given link :

https://github.com/Hussain147/webapp-on-aws/blob/main/userdata/tomcat_ubuntu.sh

User data - *optional* Info

Enter user data in the field.

```
#!/bin/bash
sudo apt update
sudo apt upgrade -y
sudo apt install openjdk-8-jdk -y
sudo apt install tomcat8 tomcat8-admin tomcat8-docs tomcat8-common git
-y
```

Click Launch Instance

5. Update IP to name mapping in route 53:-

Copy all the instances private ip's in a notepad like below

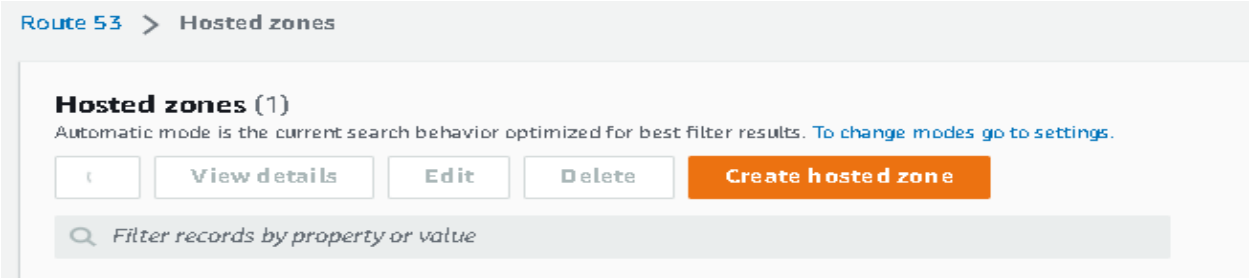
vprofile-db01:- Private Ip addr

vprofile-mc01 :- Private Ip addr

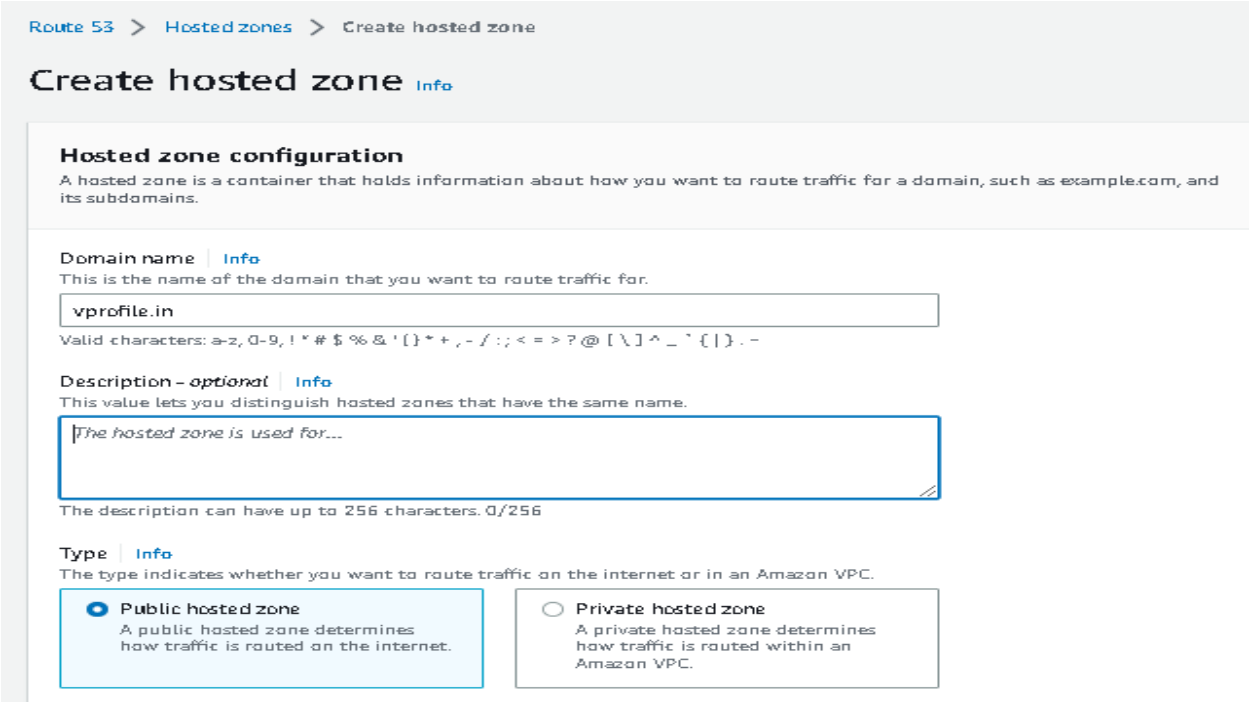
vprofile-rmq01 :- Private Ip addr

Now Goto Route 53 >

Click on Create Hosted Zone

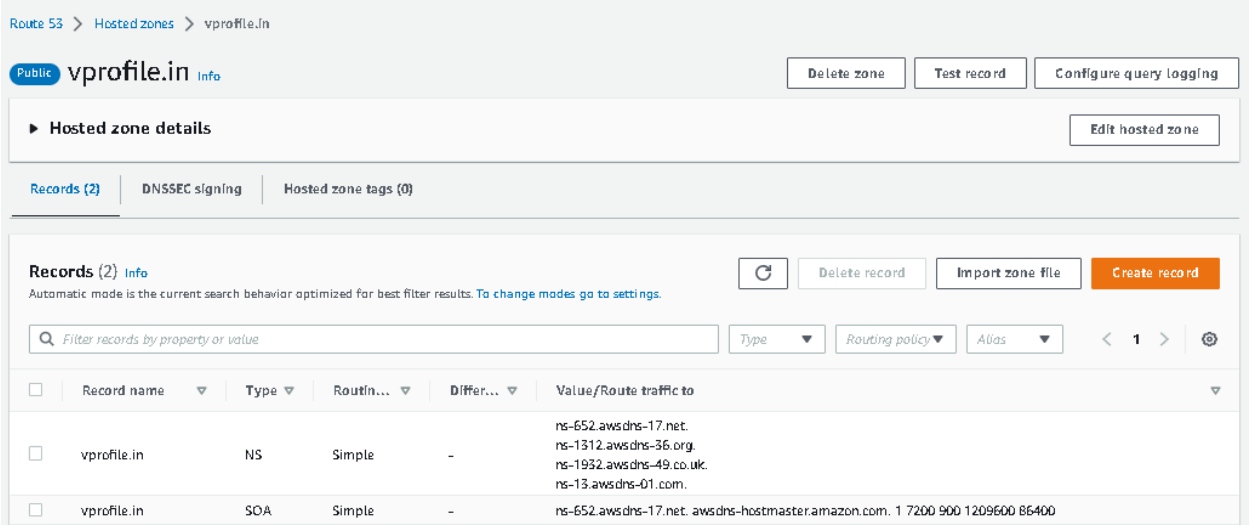


Domain Name: vprofile.in



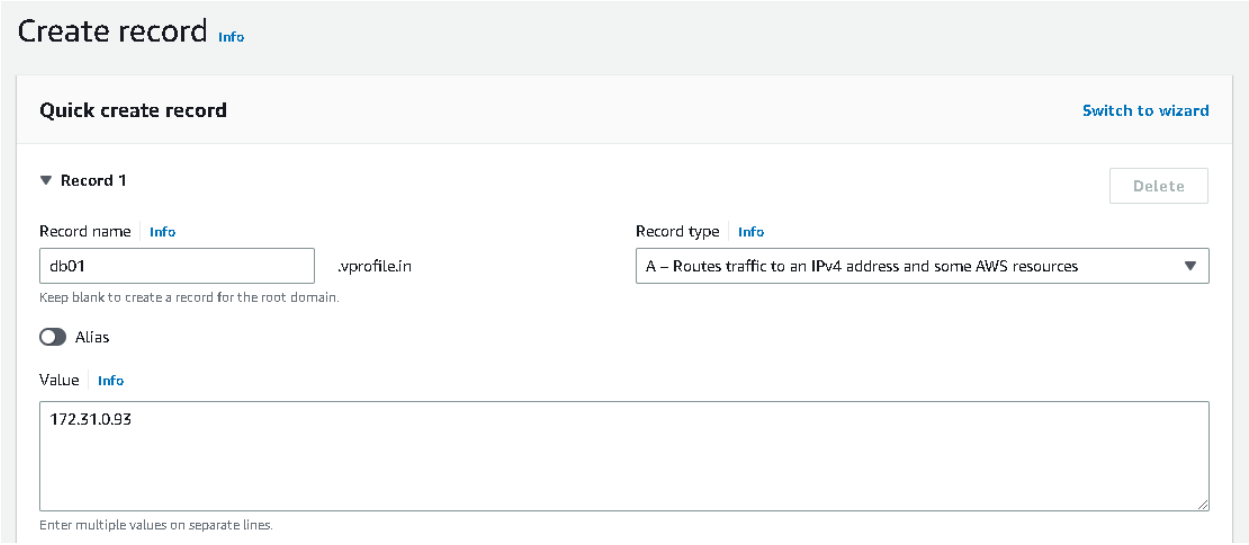
Click **Create Hosted Zone**

Now, Click create Record



Record Name: - db01

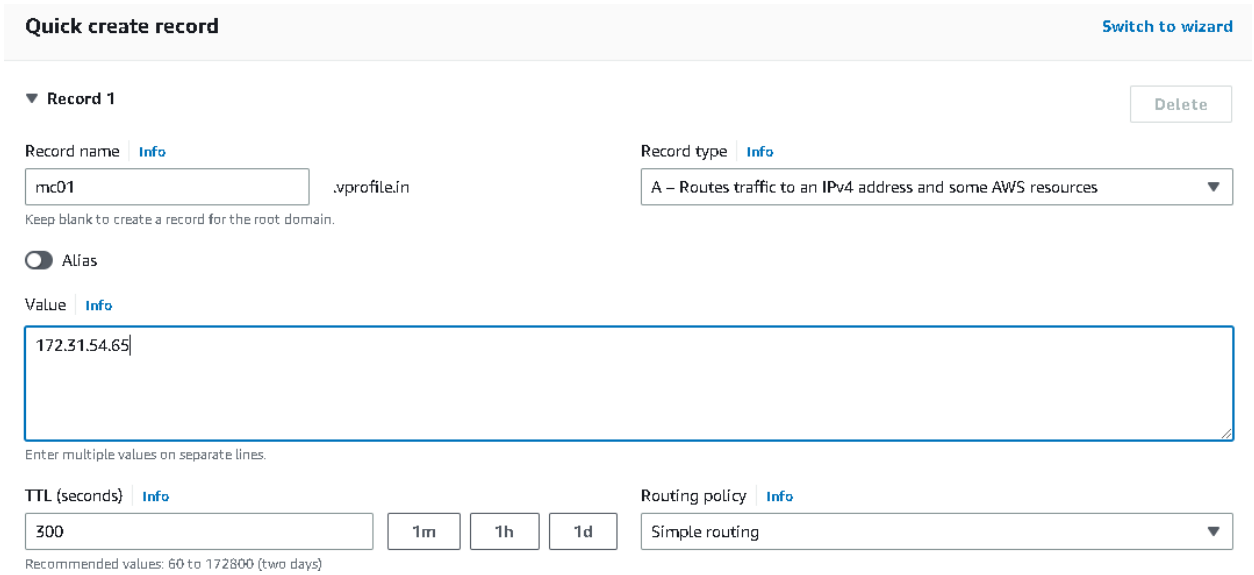
Value :- <db01 private ip>



Click Create Record

Record Name: - mc01

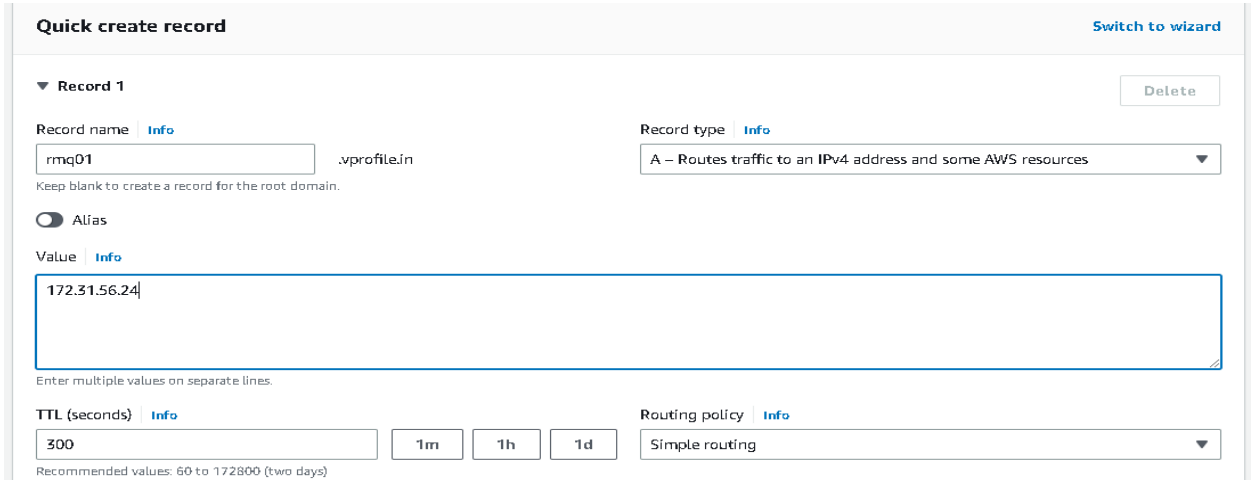
Value :- <mc01 private ip>



Click Create Record

Record Name: - rmq01

Value :- <rmq01 private ip>



Click **Create Record**

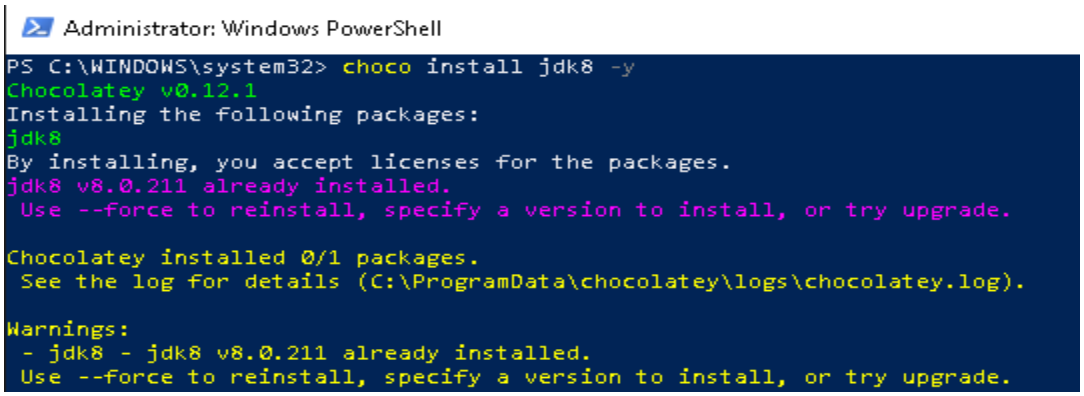
6. Build Application from source code:-

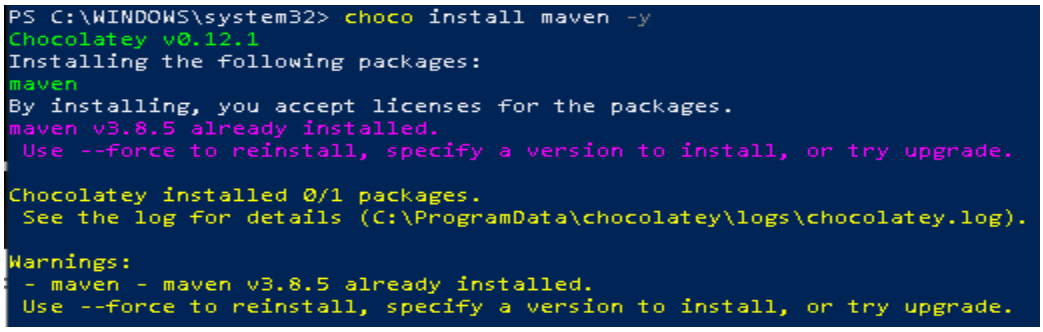
Now come to local machine > open powershell with administration > install jdk 8 & maven

Note :- you need to install chocolatey in powershell before installing the jdk8 & maven

choco install jdk8 –y

choco install mvn -y



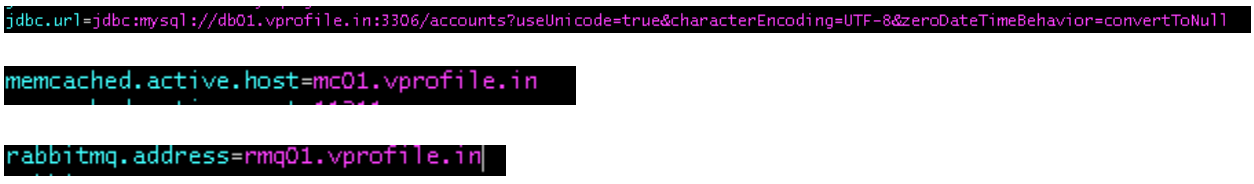


Now go to src > main > resources

You will have application.properties file

Edit the properties file as given in screenshots below

vi application.properties



```
#JDBC Configutation for Database Connection
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://db01.vprofile.in:3306/accounts?useUnicode=true&characterEncoding=UTF-8&zeroDateTimeBehavior=convertToNull
jdbc.username=admin
jdbc.password=admin123

#Memcached Configuration For Active and StandBy Host
#For Active Host
memcached.active.host=mc01.vprofile.in
memcached.active.port=11211
#For StandBy Host
memcached.standBy.host=127.0.0.2
memcached.standBy.port=11211

#RabbitMq Configuration
rabbitmq.address=rmq01.vprofile.in
rabbitmq.port=5672
irabbitmq.username=test
rabbitmq.password=test

#Elasticesearch Configuration
elasticsearch.host =192.168.1.85
elasticsearch.port =9300
elasticsearch.cluster=vprofile
elasticsearch.node=vprofilenode
~
```

Save and exit

Now come to our project directory where we see src directory

Type **mvn install** to build our artifact

mvn install (this will take couple of minutes to build an artifact)

```
Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Se
$ cd ../../../

Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Se
tup With AWS Project/webapp-on-aws (main)
$ ls
 Jenkinsfile   'Web App Setup With AWS Project.docx'   pom.xml   userdata/
 README.md      ansible/                               src/

Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Se
tup With AWS Project/webapp-on-aws (main)
$ mvn install
```

Type ls > you will get Target directory

```
Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Se
tup With AWS Project/webapp-on-aws (main)
$ ls
 Jenkinsfile   'Web App Setup With AWS Project.docx'   pom.xml   target/
 README.md      ansible/                               src/      userdata/
```

Goto Target Directory >type ls > You will get vprofile-v2.war(It is built by us using maven)

```
Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Setup With AWS Project/webapp-on-aws (main)
$ cd target

Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Setup With AWS Project/webapp-on-aws/target (main)
$ ls
classes/  generated-sources/  generated-test-sources/  jacoco.exec  maven-archiver/  maven-status/  site/  surefire-reports/  test-classes/  vprofile-v2/  vprofile-v2.war
```

7. Upload to S3 bucket:-

Now, we need to send this artifact(vprofile-v2.war) to the S3 bucket by using AWSCLI

Note:- You need to install awscli in local machine & configured

```
Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Setup With AWS Project/webapp-on-aws/target (main)
$ aws configure
AWS Access Key ID [****************ONHT]:
AWS Secret Access Key [****************wLNI]:
Default region name [us-east-1]:
Default output format [None]:
```

Create a S3 bucket:-

aws s3 mb s3://my-vprofile-artifact

Note:- You need to take unique bucket name(S3 bucket name should be different)

```
Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Se
tup With AWS Project/webapp-on-aws/target (main)
$ aws s3 mb s3://my-vprofile-artifact
make_bucket: my-vprofile-artifact
```

Now Copy our artifact to our S3 bucket & validate :-

aws s3 cp vprofile-v2.war s3://my-vprofile-artifact/vprofile-v2.war

aws s3 ls s3://my-vprofile-artifact/

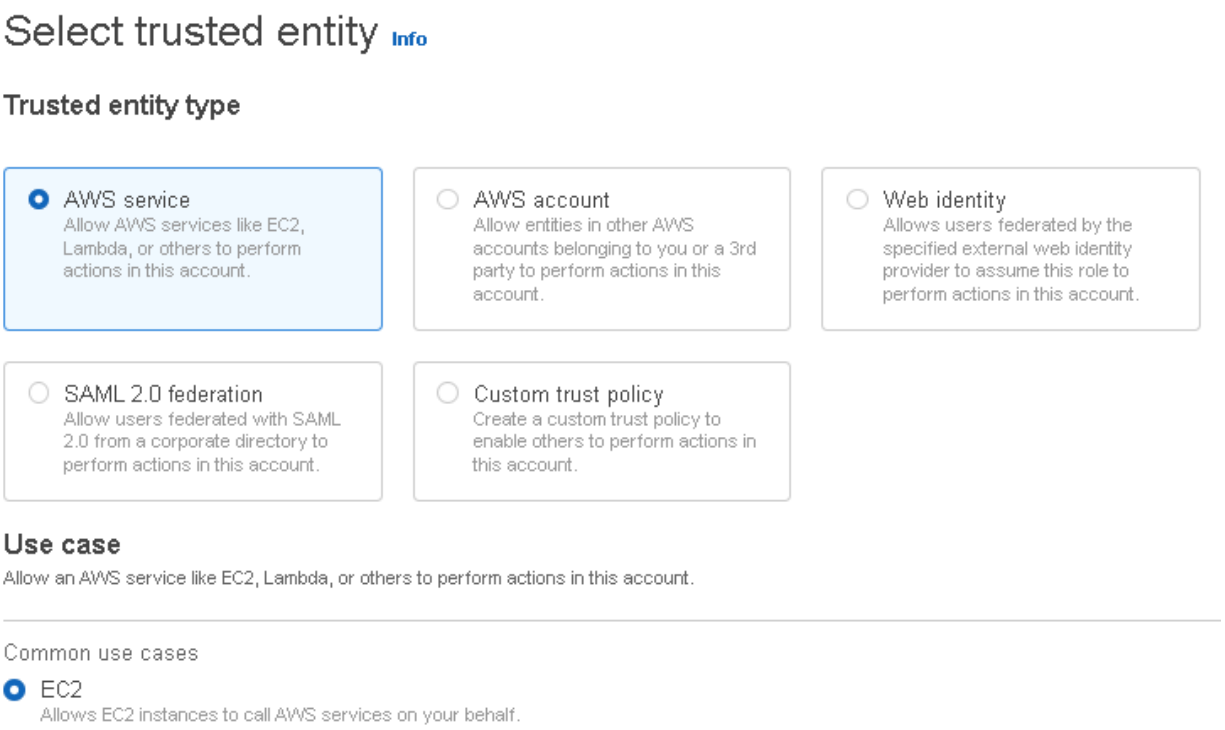
```
Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Se
tup With AWS Project/webapp-on-aws/target (main)
$ aws s3 cp vprofile-v2.war s3://my-vprofile-artifact/vprofile-v2.war
upload: .\vprofile-v2.war to s3://my-vprofile-artifact/vprofile-v2.war

Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop/Institute/Projects/Web App Se
tup With AWS Project/webapp-on-aws/target (main)
$ aws s3 ls s3://my-vprofile-artifact/
2023-02-28 20:23:27   48450938 vprofile-v2.war
```

Now we need to create a Role to store our artifact to the tomcat server

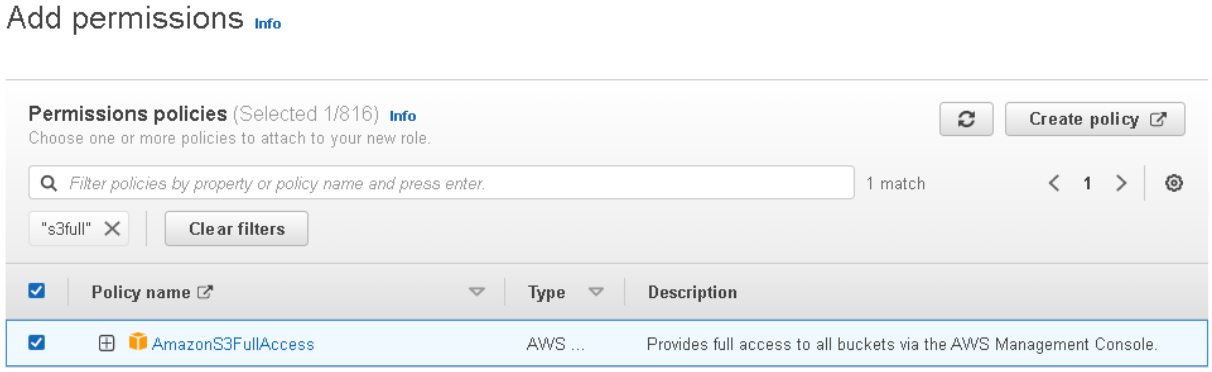
Go to IAM > Role > Create Role :

Select AWS Service >

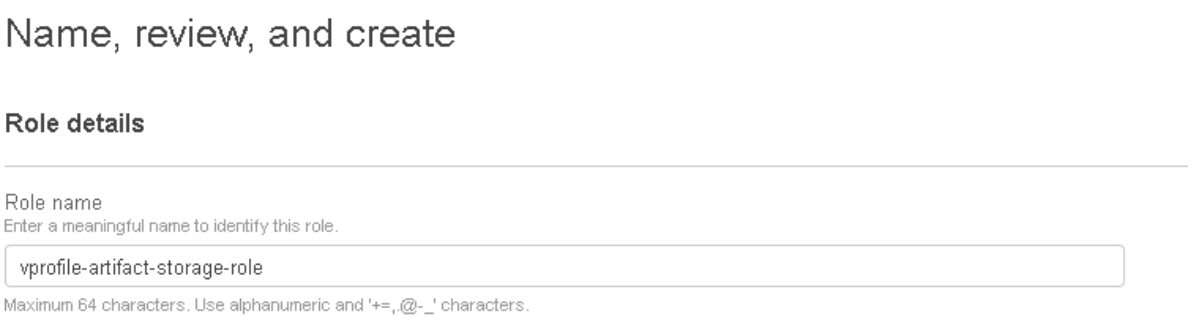
Use Case: EC2



Add permissions > s3fullaccess



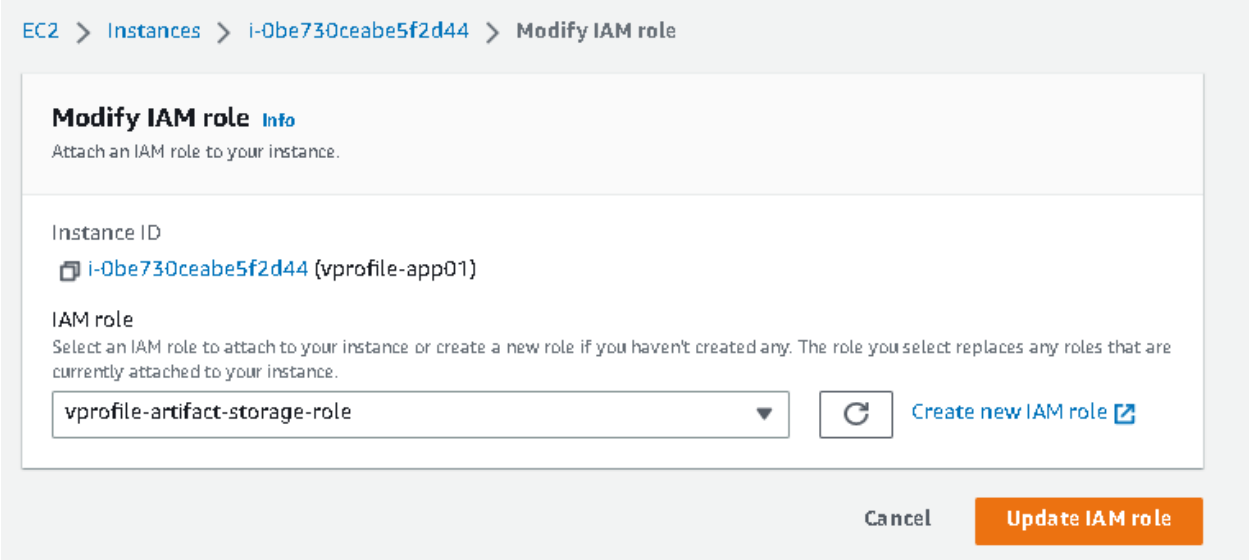
Role Name: vprofile-artifact-storage-role



Click **Create Role**

Now go to EC2 > select app01 server > Actions > Security > Modify IAM Role >

Select our Role (vprofile-artifact-storage-role)



Click Update IAM Role

8. Download Artifact to Tomcat Ec2 instance:-

Now connect app01 server with ssh >

```
Hussain@DESKTOP-572PBGQ MINGW64 ~/OneDrive/Desktop
$ ssh -i vprofile-key.pem ubuntu@34.203.33.172
```

Now go to /var/lib/tomcat8/webapps

Stop the tomcat server: - systemctl stop tomcat

Remove ROOT directory :- rm –rf ROOT

```
root@ip-172-31-62-187:/var# ls
backups  cache  crash  lib  local  lock  log  mail  opt  run  snap  spool  tmp
root@ip-172-31-62-187:/var# clear
root@ip-172-31-62-187:/var# ls
backups  cache  crash  lib  local  lock  log  mail  opt  run  snap  spool  tmp
root@ip-172-31-62-187:/var# cd lib
root@ip-172-31-62-187:/var/lib# ls
AccountsService    dpkg            man-db     python                  ucf
amazon             git             misc       shim-signed             unattended-upgrades
apport             grub            mlocate    snapd                   update-manager
apt                initramfs-tools os-prober  sudo                    update-notifier
cloud              landscape       pam        systemd                 ureadahead
command-not-found  logrotate       plymouth   tomcat8                 usbutils
dbus               lxcfs           polkit-1   ubuntu-advantage        vim
dhcp               lxd             private    ubuntu-release-upgrader
root@ip-172-31-62-187:/var/lib# cd tomcat8
root@ip-172-31-62-187:/var/lib/tomcat8# ls
conf  lib  logs  policy  webapps  work
root@ip-172-31-62-187:/var/lib/tomcat8# cd webapps
root@ip-172-31-62-187:/var/lib/tomcat8/webapps# ls
ROOT
root@ip-172-31-62-187:/var/lib/tomcat8/webapps# systemctl stop tomcat8
root@ip-172-31-62-187:/var/lib/tomcat8/webapps# ls
ROOT
root@ip-172-31-62-187:/var/lib/tomcat8/webapps# rm -rf ROOT
root@ip-172-31-62-187:/var/lib/tomcat8/webapps# ls
```

Now Install AWSCLI , (No need to Configure)

```
root@ip-172-31-62-187:/var/lib/tomcat8/webapps# apt install awscli -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

aws s3 ls s3://my-vprofile-artifact

Download the artifact from S3 to our tomcat server in /tmp directory

aws s3 cp s3://my-vprofile-artifact/vprofile-v2.war /tmp/vprofile-v2.war

```
root@ip-172-31-62-187:~# aws s3 ls s3://my-vprofile-artifact
2023-02-28 14:53:27   48450938 vprofile-v2.war
root@ip-172-31-62-187:~# aws s3 cp s3://my-vprofile-artifact/vprofile-v2.war /tmp/vprofile-v2.war
download: s3://my-vprofile-artifact/vprofile-v2.war to ../tmp/vprofile-v2.war
root@ip-172-31-62-187:~#
```

Now copy the artifact to /var/lib/tomcat8/ROOT.war

```
root@ip-172-31-62-187:/tmp# cp vprofile-v2.war /var/lib/tomcat8/webapps/ROOT.war
```

Start Tomcat8 :- systemctl start tomcat8

```
root@ip-172-31-62-187:/tmp# systemctl start tomcat8
```

Now go to the path as given below:-

cd /var/lib/tomcat8/webapps/ROOT/WEB-INF/classes

ls

cat application.properties

```
root@ip-172-31-62-187:/var/lib/tomcat8/webapps/ROOT/WEB-INF/classes# ls
application.properties  com  db_backup.sql  logback.xml  validation.properties
root@ip-172-31-62-187:/var/lib/tomcat8/webapps/ROOT/WEB-INF/classes# cat application.properties
#JDBC Configutation for Database Connection
jdbc.driverClassName=com.mysql.jdbc.Driver
jdbc.url=jdbc:mysql://db01.vprofile.in:3306/accounts?useUnicode=true&characterEncoding=UTF-8&ze
roDateTimeBehavior=convertToNull
jdbc.username=admin
jdbc.password=admin123

#Memcached Configuration For Active and StandBy Host
#For Active Host
memcached.active.host=mc01.vprofile.in
memcached.active.port=11211
#For StandBy Host
memcached.standBy.host=127.0.0.2
memcached.standBy.port=11211

#RabbitMq Configuration
rabbitmq.address=rmq01.vprofile.in
rabbitmq.port=5672
irabbitmq.username=test
rabbitmq.password=test

#Elasticesearch Configuration
elasticsearch.host =192.168.1.85
elasticsearch.port =9300
elasticsearch.cluster=vprofile
elasticsearch.node=vprofilenode
root@ip-172-31-62-187:/var/lib/tomcat8/webapps/ROOT/WEB-INF/classes#
```

Then type vi /etc/hosts > add the private IP's of backend servers(db01, mc01, rmq01)

```
127.0.0.1 localhost
172.31.0.93 db01.vprofile.in
172.31.54.65 mc01.vprofile.in
172.31.56.24 rmq01.vprofile.in

# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
~
```

Save & Exit

Now, type telnet db01.vprofile.in 3306 -> to confirm that our backend servers are connecting to tomcat server

```
root@ip-172-31-18-202:~# telnet db01.vprofile.in 3306
Trying 172.31.0.93...
Connected to db01.vprofile.in.
Escape character is '^]'.
R
5.5.68-MariaDB
              @+S@&;9CSYE,LzD%(8$mysql_native_password
```

9. Setup ELB with HTTP:-

Go to Load Balancer > Target Groups > Create Target Groups >

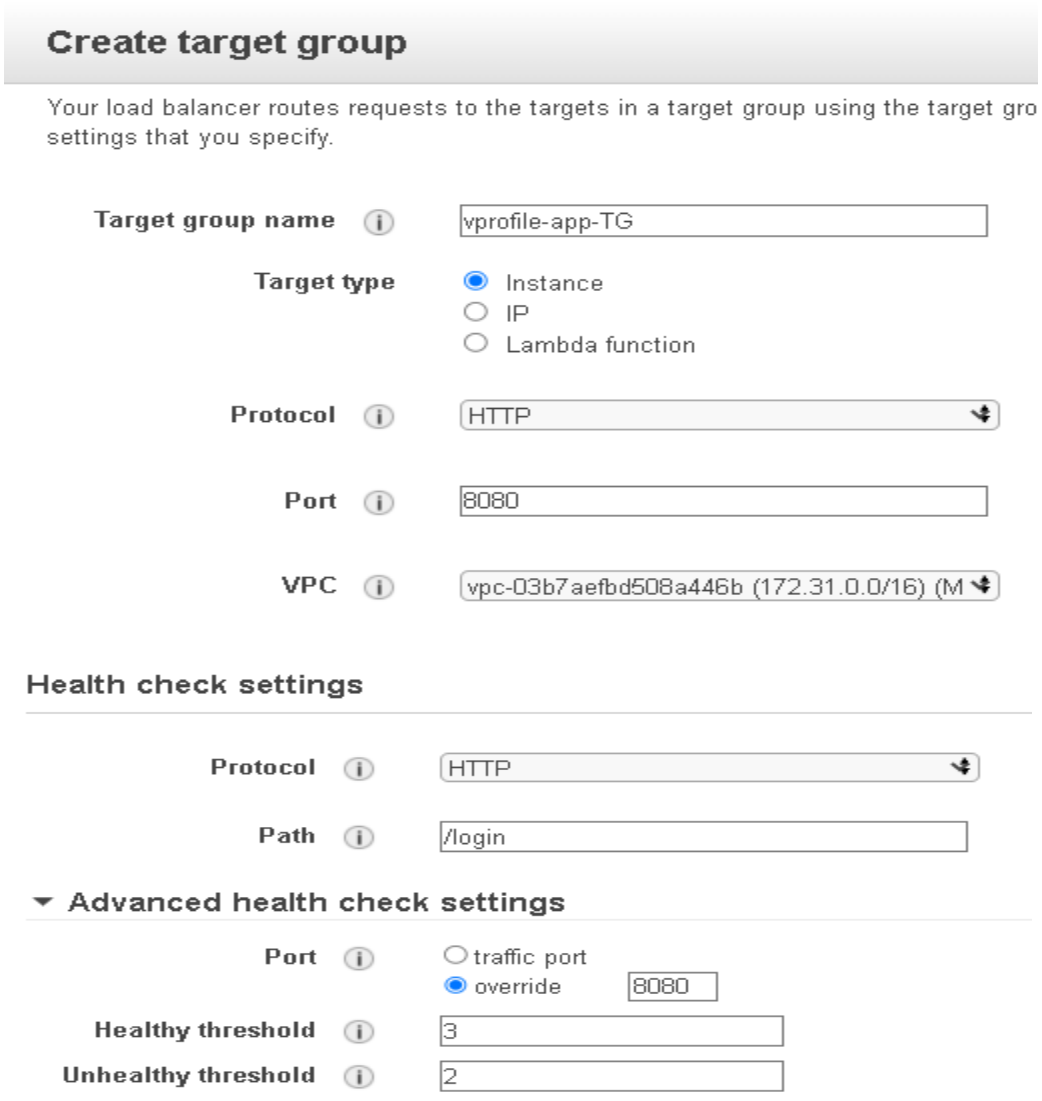
Target Group Name : vprofile-app-TG

Target Type: Instance

Protocol : HTTP

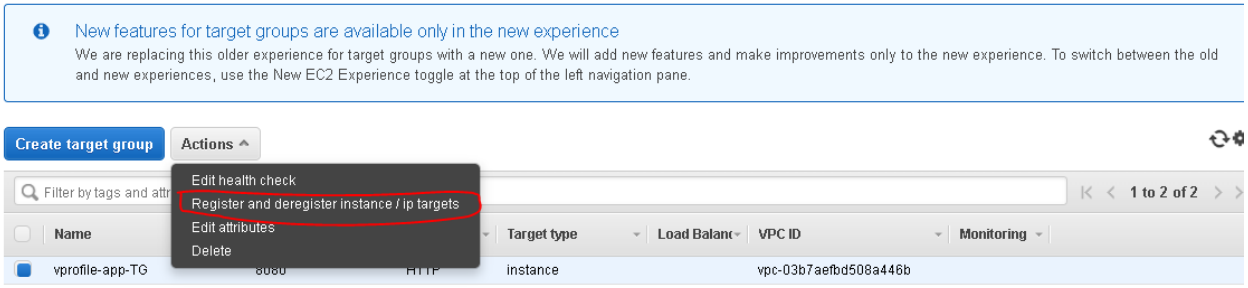
Port: 8080

Path : /login

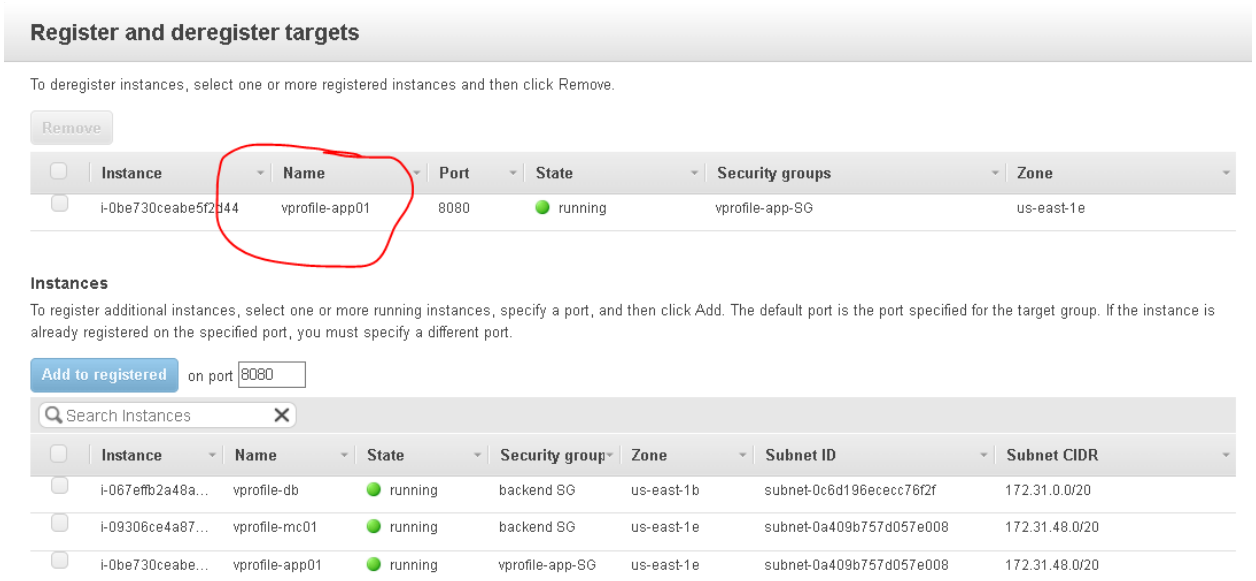


Click **Create**

Now, Select our Target Group: vprofile-app-TG >



Select the Target Group > Register Instances > Register Instances > Select our Tomcat App Instance > Add > Save Changes

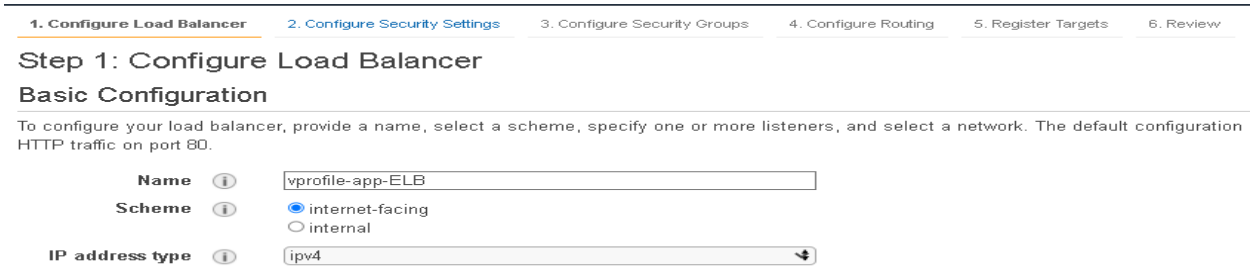


Now Go to Load Balancer > Create Load Balancer > Select Application Load Balancer > Create

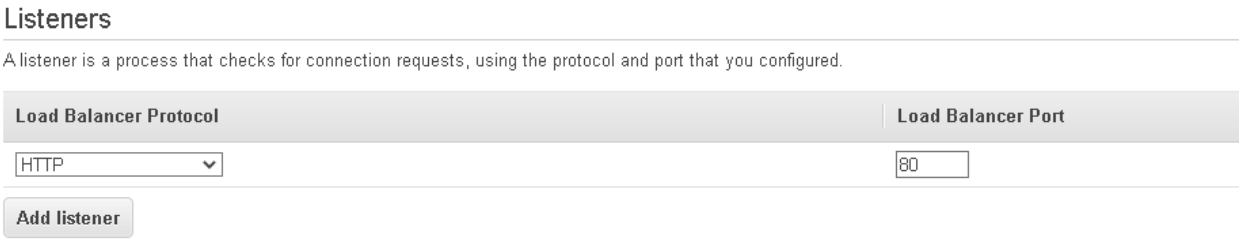
Name: vprofile-app-ELB

Scheme : Internet Facing

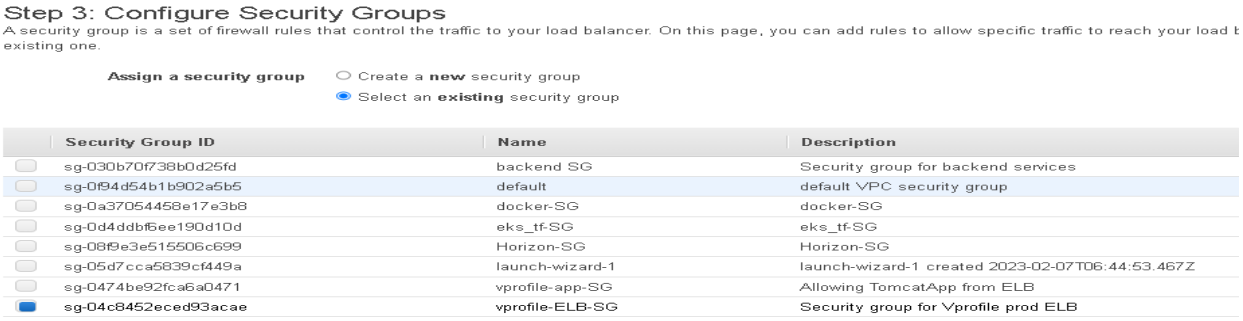
Ip type: ipv4



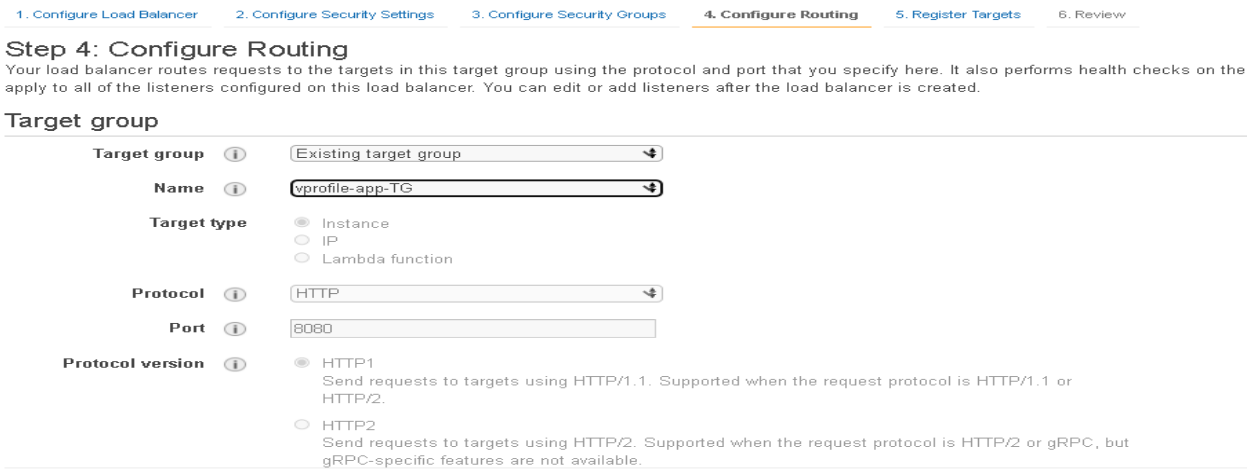
Listener: HTTP : 80



Select ELB Security Group



Select our Target Group : vprofile-app-TG

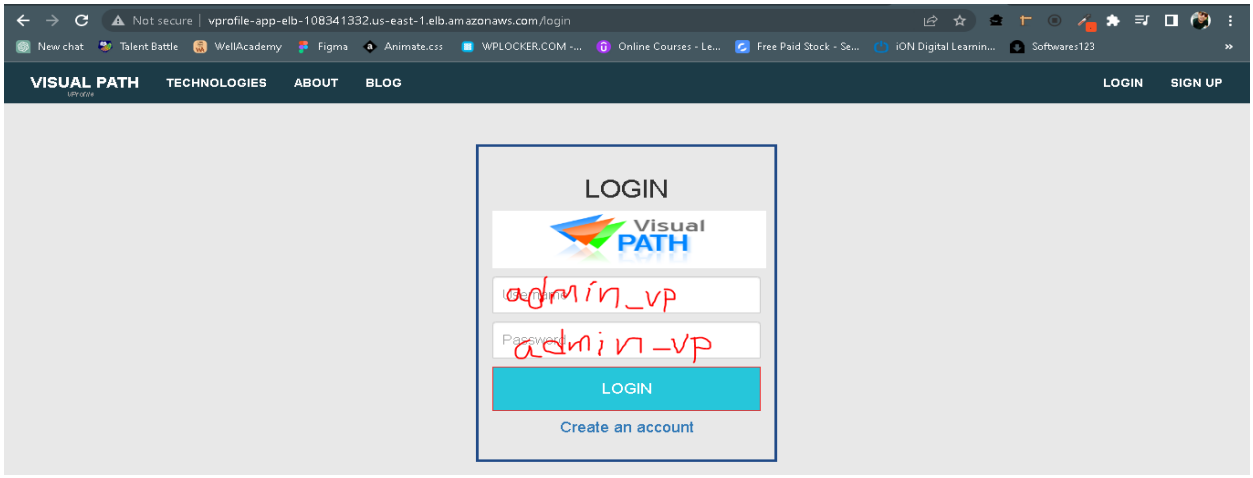


Click **Register Targets** > Click **Review**> **Next** > Click on **Create**

Then Copy the DNS Name & paste it in the URL address

Username: admin_vp

Password: admin_vp

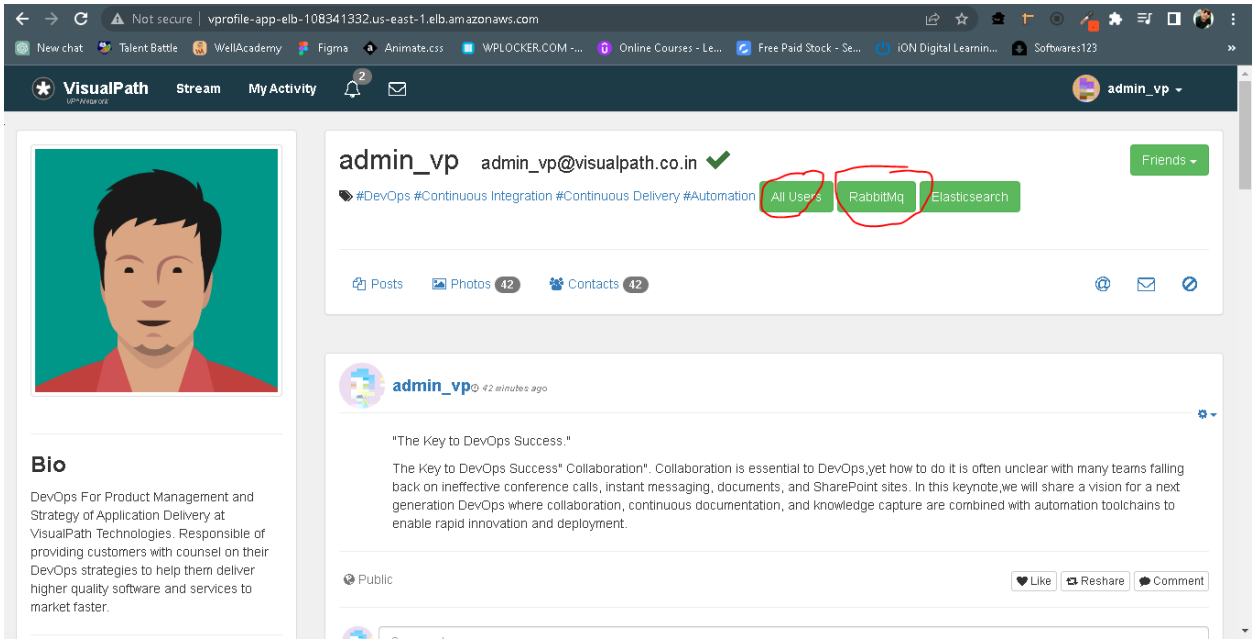


Click > Login

Once you successfully logged in, check for backend servers are working are not.

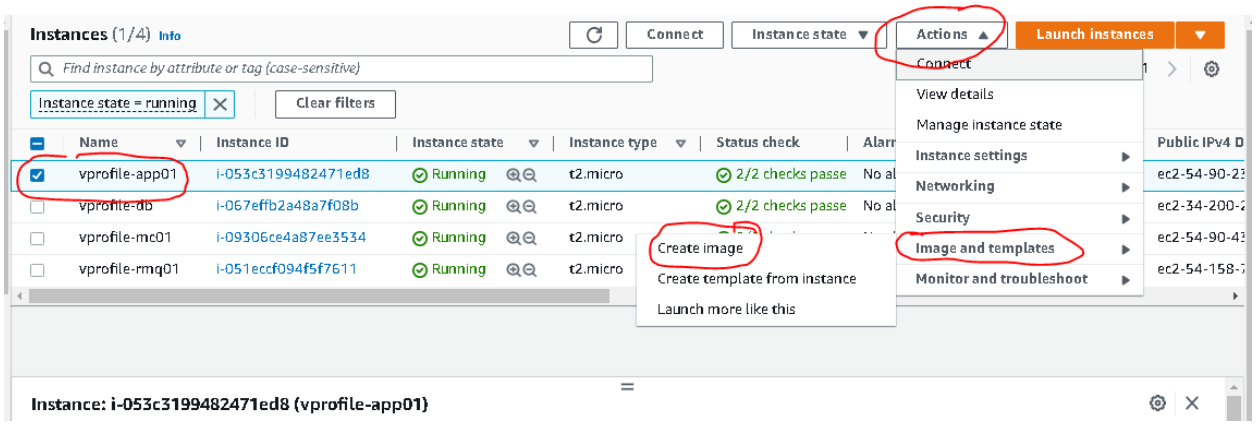
Click on All Users – If it opens a new webpage then our memcache is working

Click on RabbitMq – If it opens a new webpage then our rabbitmq is working

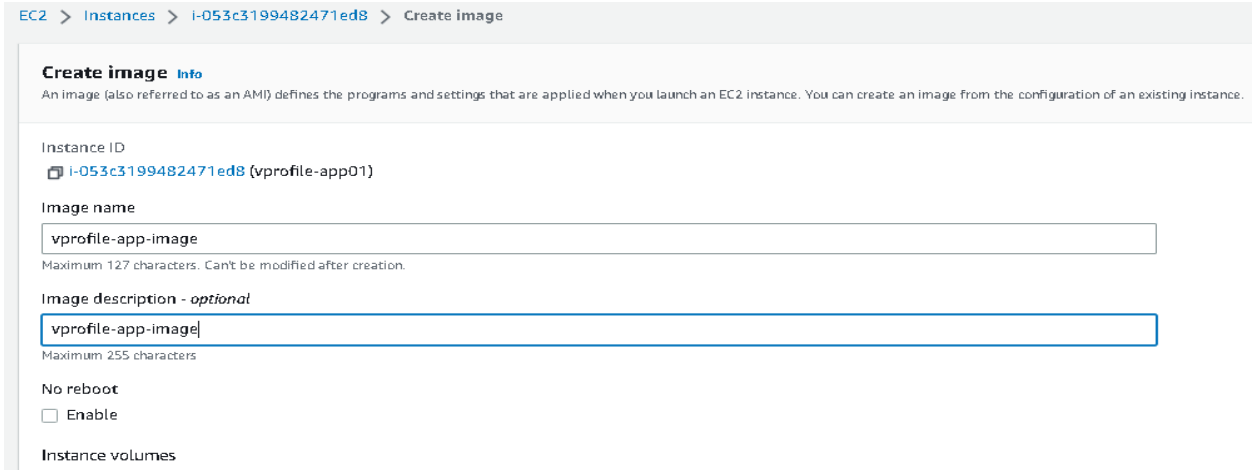

Not secure | vprofile-app-elb-108341332.us-east-1.elb.amazonaws.com
New chat
Talent Battle
WellAcademy
Figma
Animate.css
WPLOCKER.COM -...
Online Courses - Le...
Free Paid Stock - Se...
iON Digital Learnin...
Softwares123
VisualPath
Stream
My Activity
2
admin_vp
admin_vp
admin_vp@visualpath.co.in
Friends
#DevOps #Continuous Integration #Continuous Delivery #Automation
All Users
RabbitMq
Elasticsearch
Posts
Photos 42
Contacts 42
Bio
DevOps For Product Management and Strategy of Application Delivery at VisualPath Technologies. Responsible of providing customers with counsel on their DevOps strategies to help them deliver higher quality software and services to market faster.
admin_vp 42 minutes ago
"The Key to DevOps Success."
The Key to DevOps Success" Collaboration". Collaboration is essential to DevOps,yet how to do it is often unclear with many teams falling back on ineffective conference calls, instant messaging, documents, and SharePoint sites. In this keynote,we will share a vision for a next generation DevOps where collaboration, continuous documentation, and knowledge capture are combined with automation toolchains to enable rapid innovation and deployment.
Public
Like
Reshare
Comment

10. Build Autoscaling Group for Tomcat Instances:-

G0 to EC2 > select our app01 server > Actions > Image and templates > Create Image

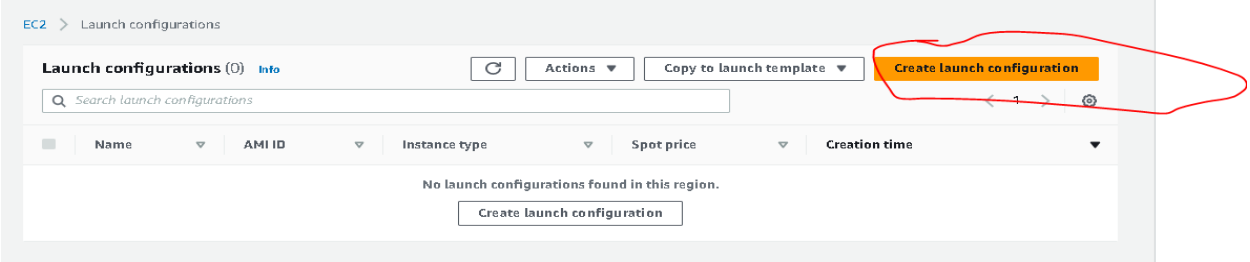


Name : vprofle-app-image



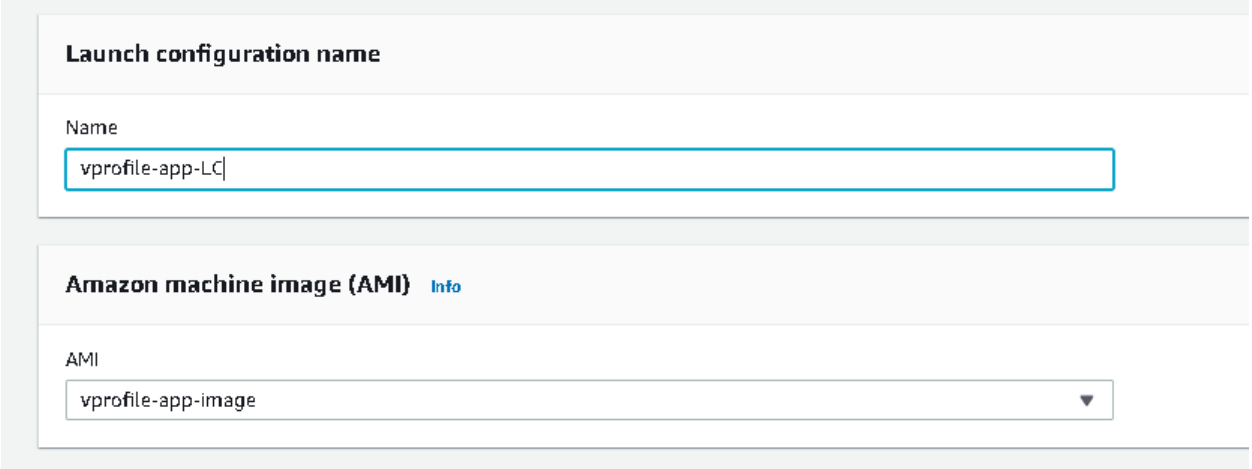
Click create

Then Go to Launch Configurations > Create Launch Configuration



Name : vprofile-app-LC

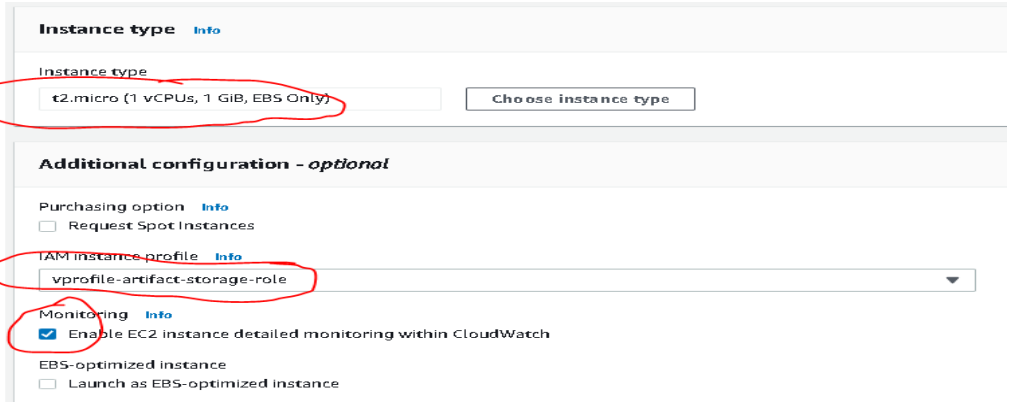
AMI : select our image (vprofile-app-image)



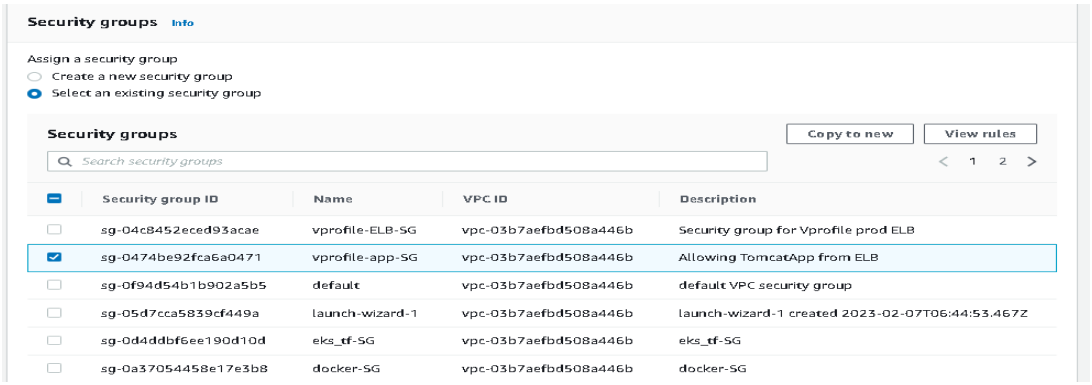
Insatnce Type : t2 micro (it is free tier)

IAM Instance Profile : select our Role (vprofile-artifact-storage-role)

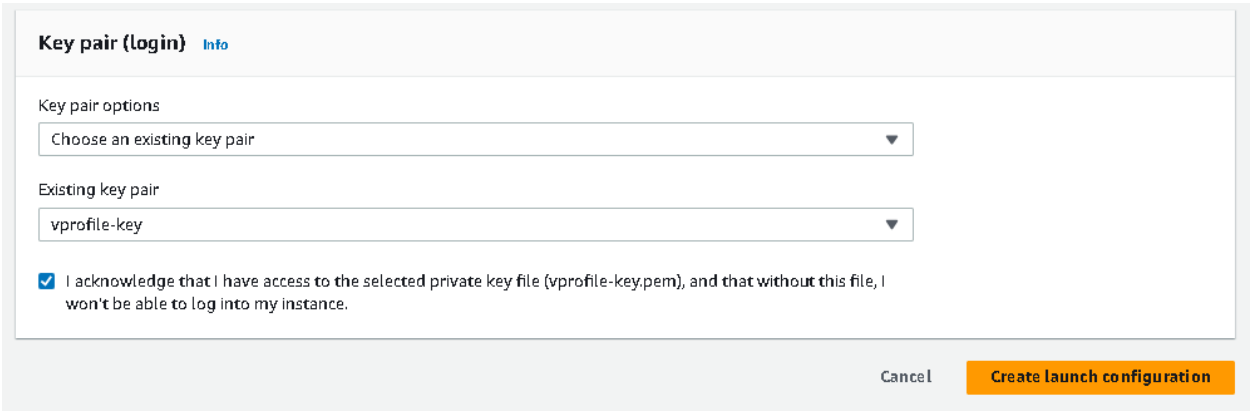
Monitoring : enable to get the logs in cloudwatch



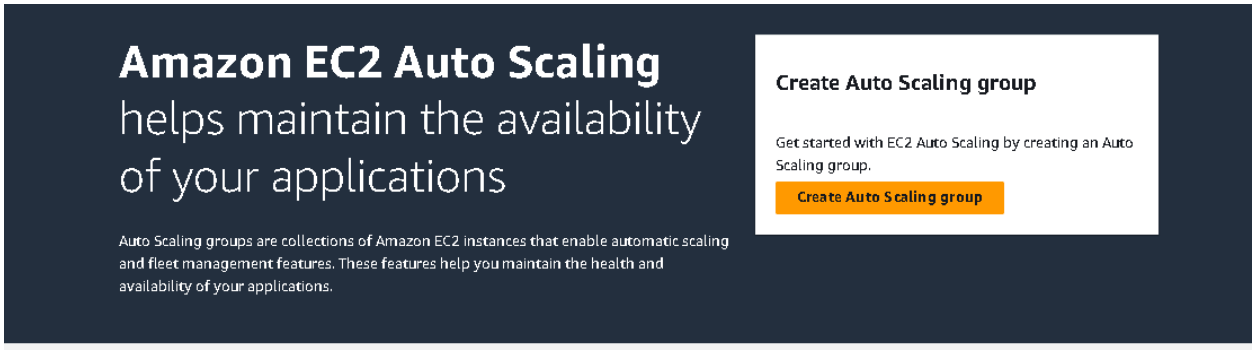
Security Group : select the **app-SG**



Keypair : vprofile-key > click **create Launch Configuration**



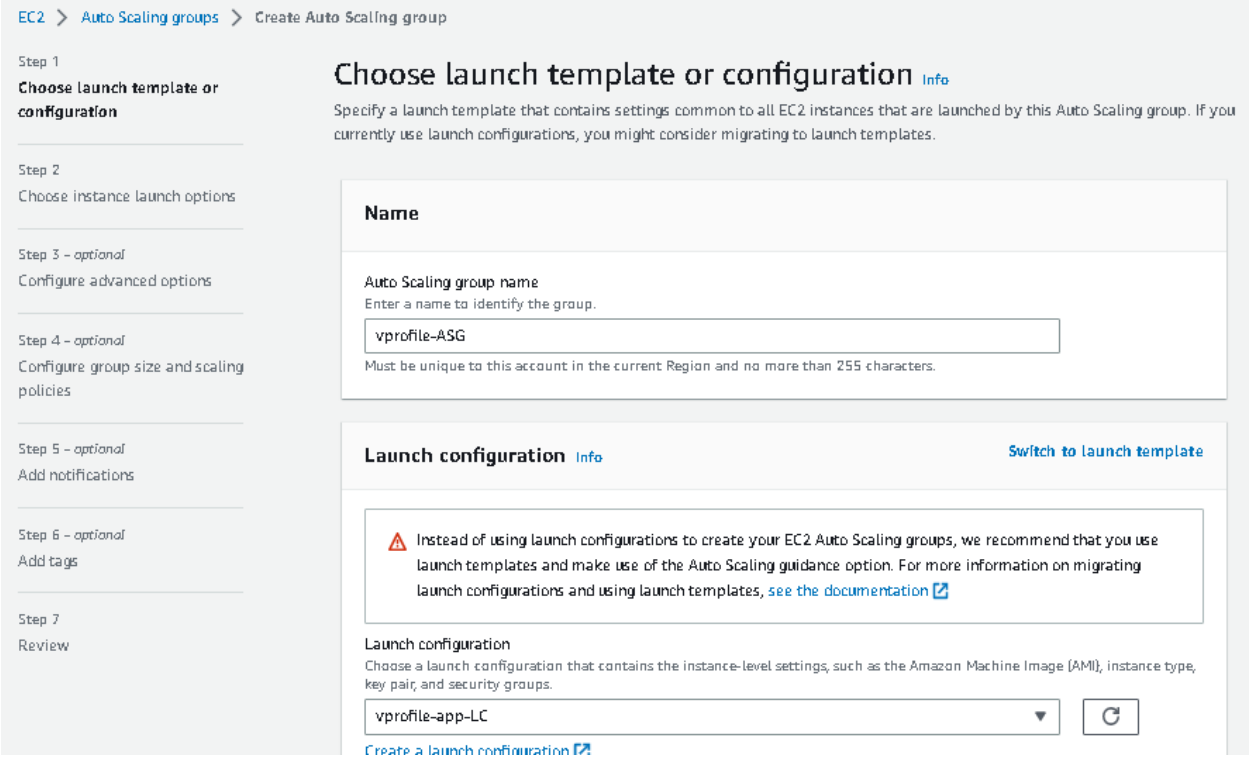
Now go to Autoscaling Groups > click on create Autoscaling Group



Name : vprofile-ASG

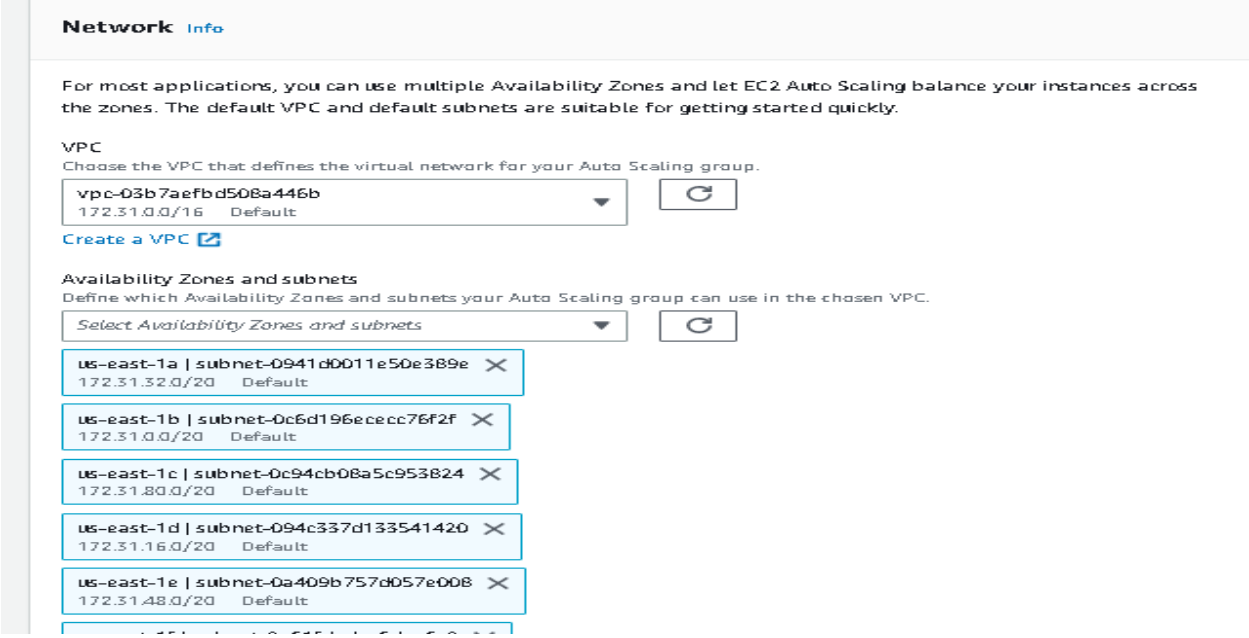
Click on **Switch to launch configuration**

Launch Configuration : vprofile-app-LC



Click Next

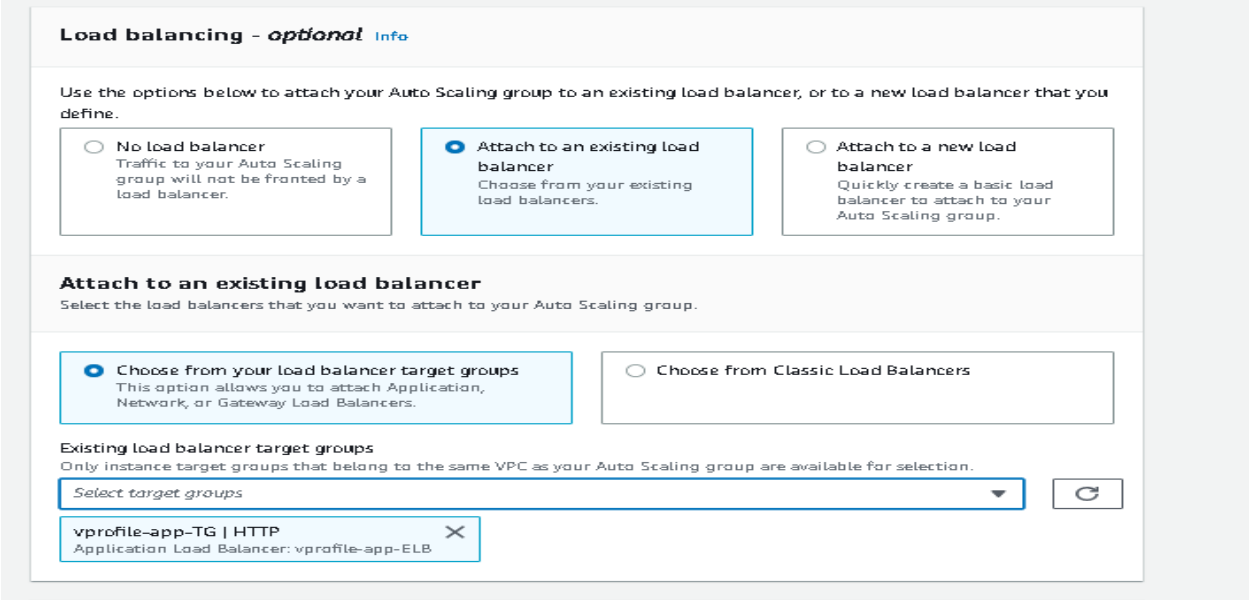
Select the VPC(in my case I'm using default VPC) & select all the Availabilty Zones



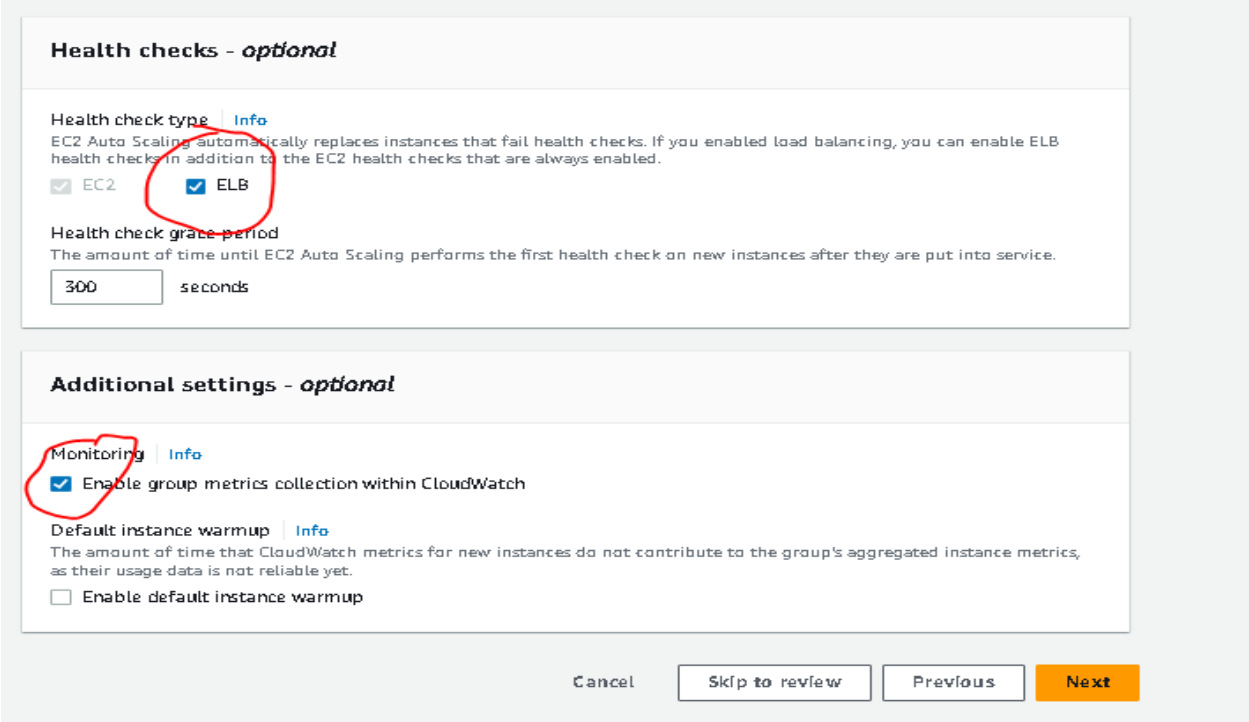
Click **Next**

Select our existing Load Balancer

Select vprofile-app-TG



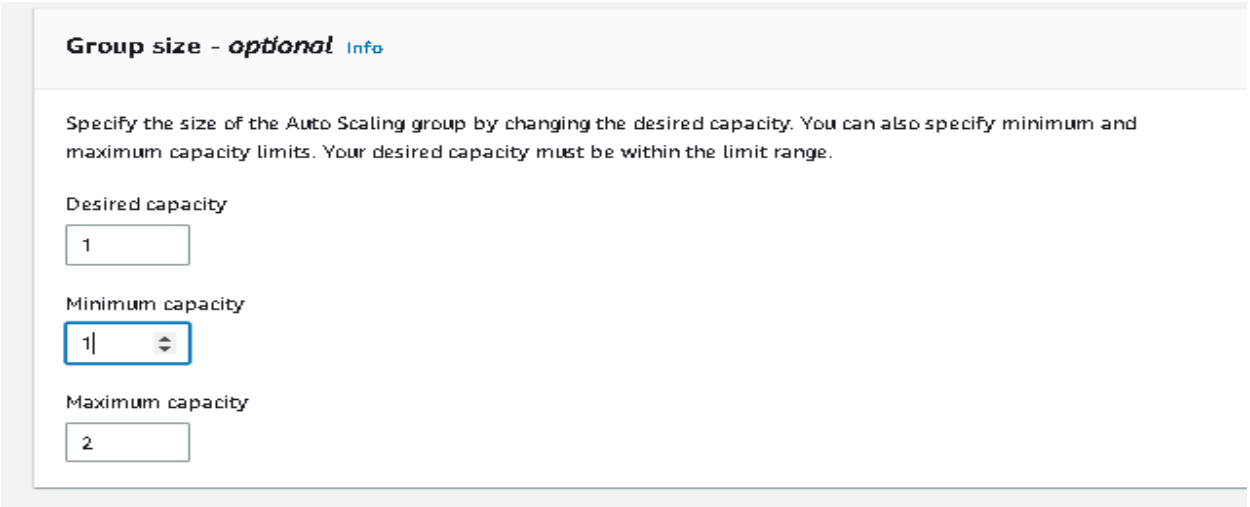
Enable **ELB** & **Monitoring**

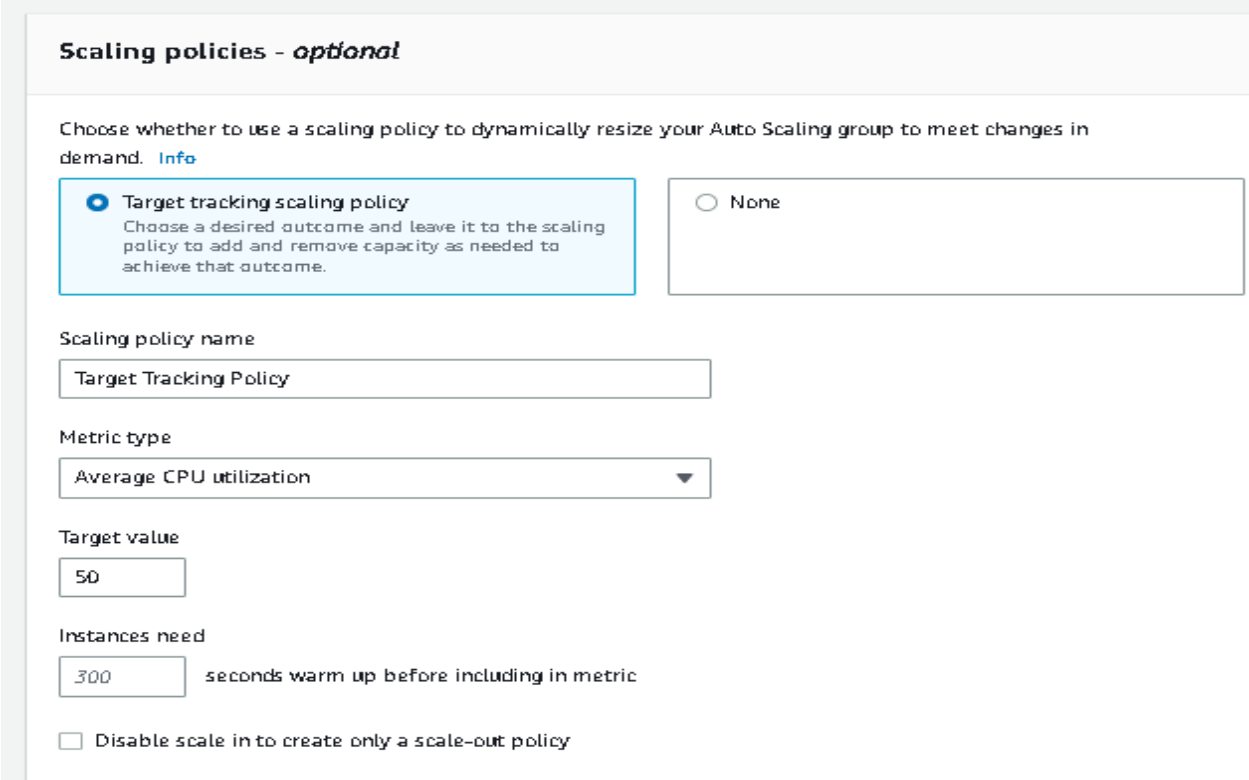


Click **Next**

Select Desired Capacity : I choose 1

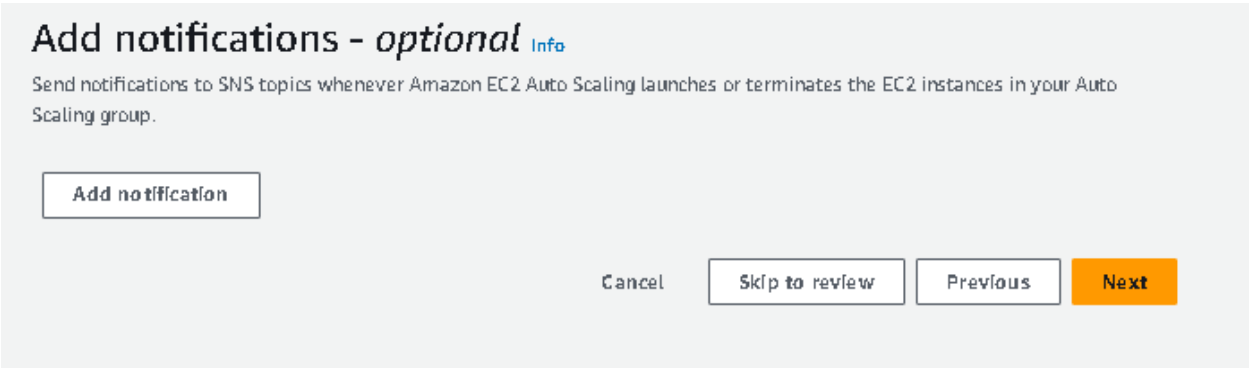
Minimum & maximum as per the requirement





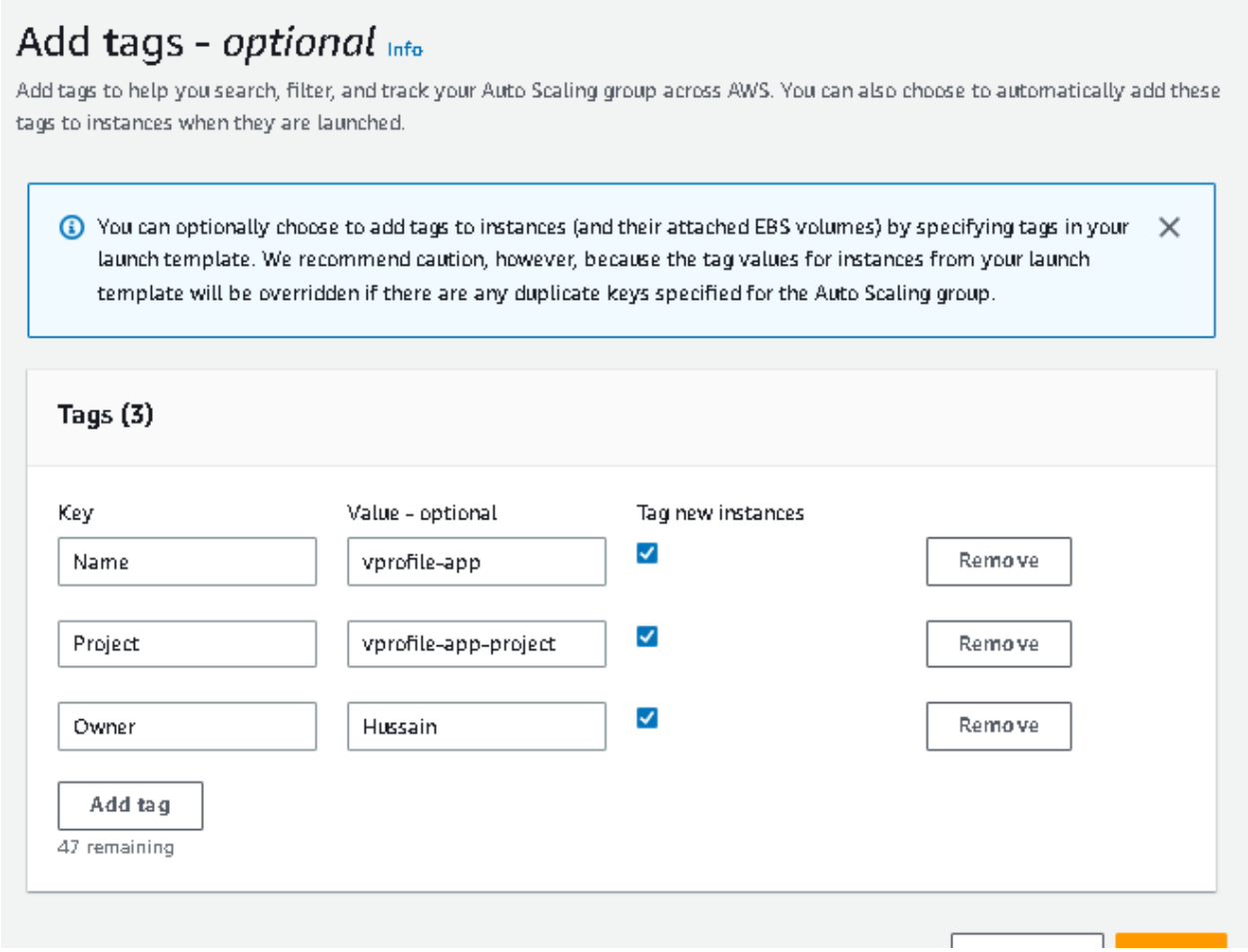
Click **Next**

Add Notification by creating SNS Topic



Click **Next**

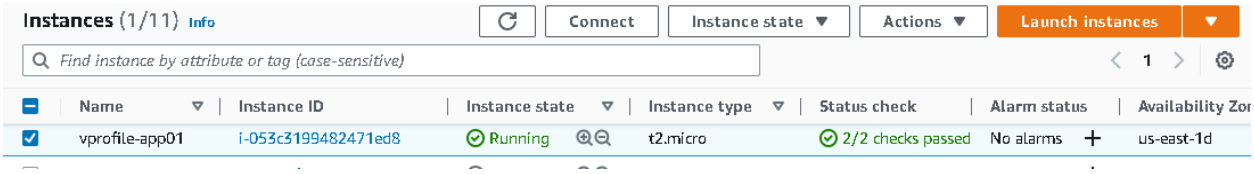
Give Some Tags as per your choice



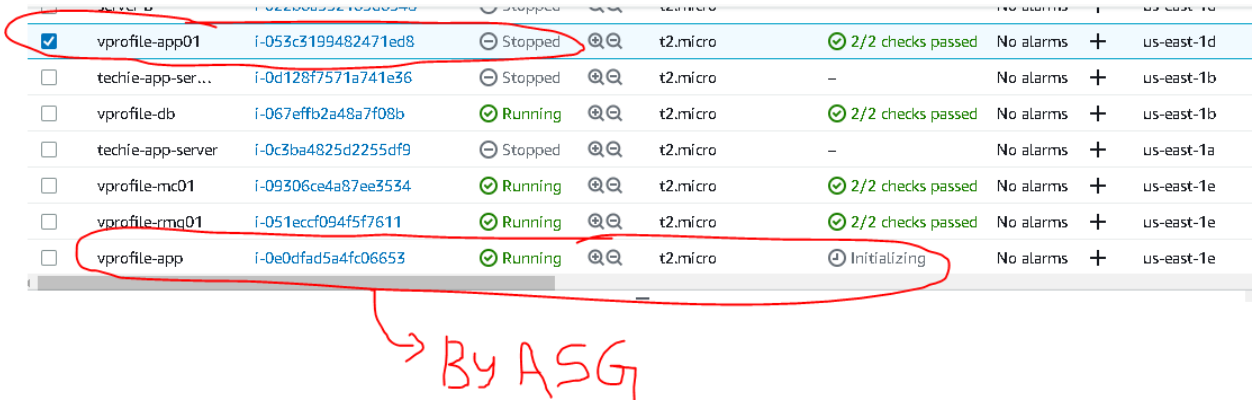
Click **Next**

Click **Create Auto Scaling Group ->**

Now if your app01 get terminated or go down then your autoscaling group will scale up the instance with your application



I have stopped the app01 wantedly > see our ASG is launching app server automatically



Note: If you want to delete the entire project, then first delete autoscaling group > Launch Configurations > Target Groups > Load Balancer > Instances > Keypairs & Security Groups

If You don't delete autoscaling first, then it will go on launching new instances when you terminate the instance

Congratulations

you have successfully deployed the java application with ELB & Autoscaling..!

~~~THANK YOU~~~