VPC Assignment

Q1. When to use Elastic IP over Public IP

Q2. Valid IP Ranges for LAN, Implication of using Public IP ranges for Private Network.

Q3. List down the things to keep in mind while VPC peering.

Q4. CIDR of a VPC is 10.0.0.0/16, if the subnet mask is /20 calculate the number of subnets that could be created from the VPC. Also find the number of IP in subnet.

Q5. Differentiate between NACL and Security Groups.

Q6. Implement a 2-tier vpc with following requirements:

1. **Create a private subnet, attach NAT, and host an application server(Tomcat)**

2. **Create a public subnet, and host a web server(Nginx), also proxypass to Tomcat from Nginx**

After Implementing this on AWS, create an architecture diagram for this use case.

Note: For hosting Nginx in public subnet, use Elastic IP.

Q1. When to use Elastic IP over Public IP

Elastic IP is used when you are working on long time project and configuration of IP sometimes consumes more time and you don't want your IP to change.

Q2. Valid IP Ranges for LAN, Implication of using Public IP ranges for Private Network.

Valid IP ranges of LAN:

- **192.168.0.0 - 192.168.255.255** (65,536 IP addresses)
- **172.16.0.0 - 172.31.255.255** (1,048,576 IP addresses)
- **10.0.0.0 - 10.255.255.255** (16,777,216 IP addresses)

It is public global addresses that are used in the Internet. A public IP address is an IP address that is used to access the Internet. Public (global) IP addresses are routed on the Internet, unlike private addresses.

The presence of a public IP address on your private network will allow you to organize your own server (VPN, FTP, WEB, etc.), remote access to your computer, video surveillance cameras, and access them from anywhere in the global network.

Q3. List down the things to keep in mind while VPC peering.

- Choosing the proper VPC configuration for your organization's needs
- Choosing a CIDR block for your VPC implementation
- Isolating your VPC environments
- Creating your disaster recovery plan
- Traffic control and security
- Keep your data close
- Determining the NAT instance type
- IAM for your AWS VPC infrastructure

Q4. CIDR of a VPC is 10.0.0.0/16, if the subnet mask is /20 calculate the number of subnets that could be created from the VPC. Also find the number of IP in subnet.

No. of subnets created = 2pow4=16

No. of IPs in a subnet = 2pow12=4096-2(reserved)=4094

Q5. Differentiate between NACL and Security Groups.

Security Group	NACL (Network Access Control List)
It supports only **allow** rules, and by default, all the rules are denied. You cannot deny the rule for establishing a connection.	It supports both **allow and deny** rules, and by default, all the rules are denied. You need to add the rule which you can either allow or deny it.

It is a **stateful** means that any changes made in the inbound rule will be automatically reflected in the outbound rule. For example, If you are allowing an incoming port 80, then you also have to add the outbound rule explicitly.	It is a **stateless** means that any changes made in the inbound rule will not reflect the outbound rule, i.e., you need to add the outbound rule separately. For example, if you add an inbound rule port number 80, then you also have to explicitly add the outbound rule.
It is associated with an EC2 instance.	It is associated with a subnet.
All the rules are evaluated before deciding whether to allow the traffic.	Rules are evaluated in order, starting from the lowest number.
Security Group is applied to an instance only when you specify a security group while launching an instance.	NACL has applied automatically to all the instances which are associated with an instance.
It is the first layer of defense.	It is the second layer of defense.

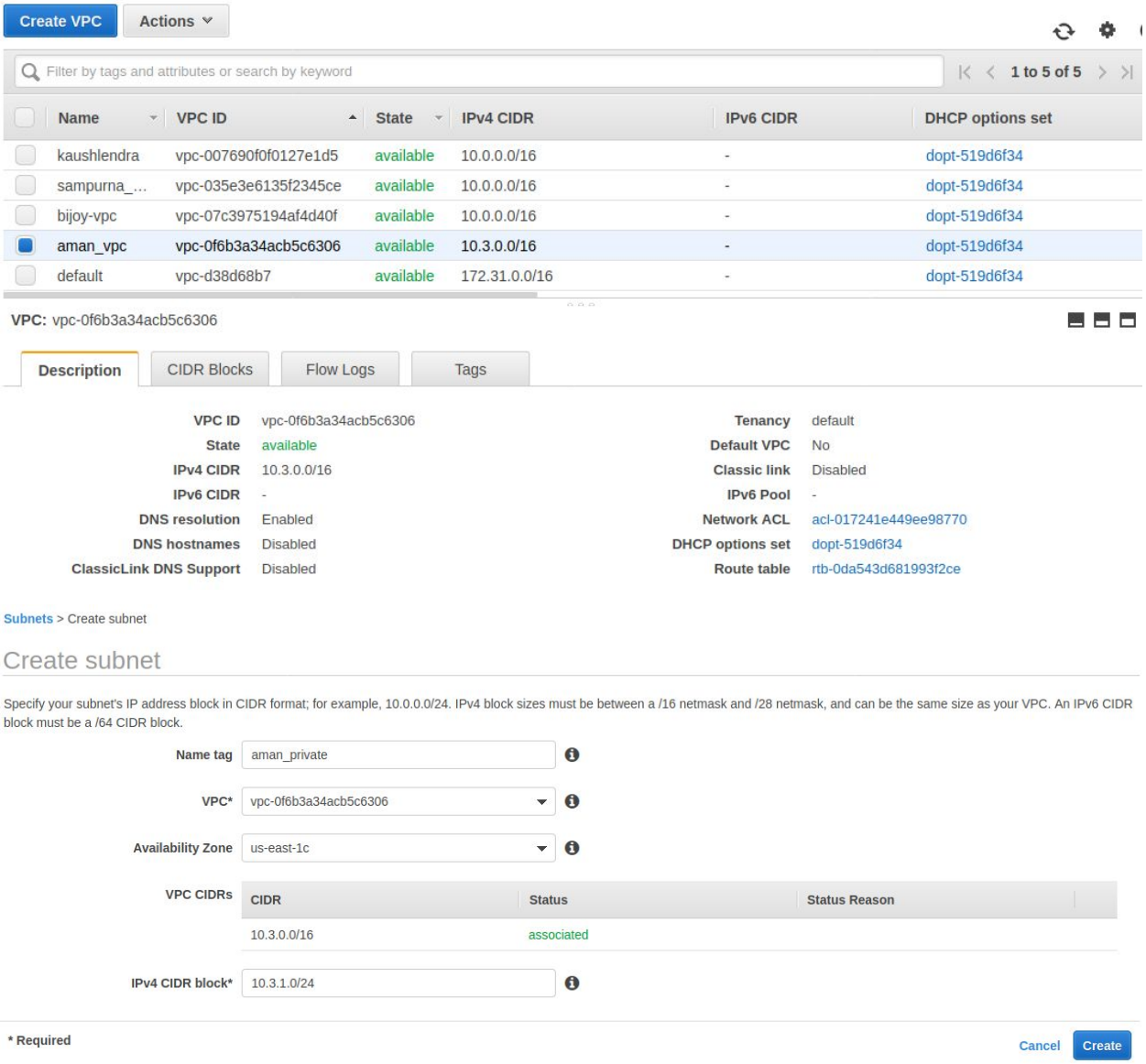
Q6. Implement a 2-tier vpc with following requirements:

1. Create a private subnet, attach NAT, and host an application server(Tomcat)

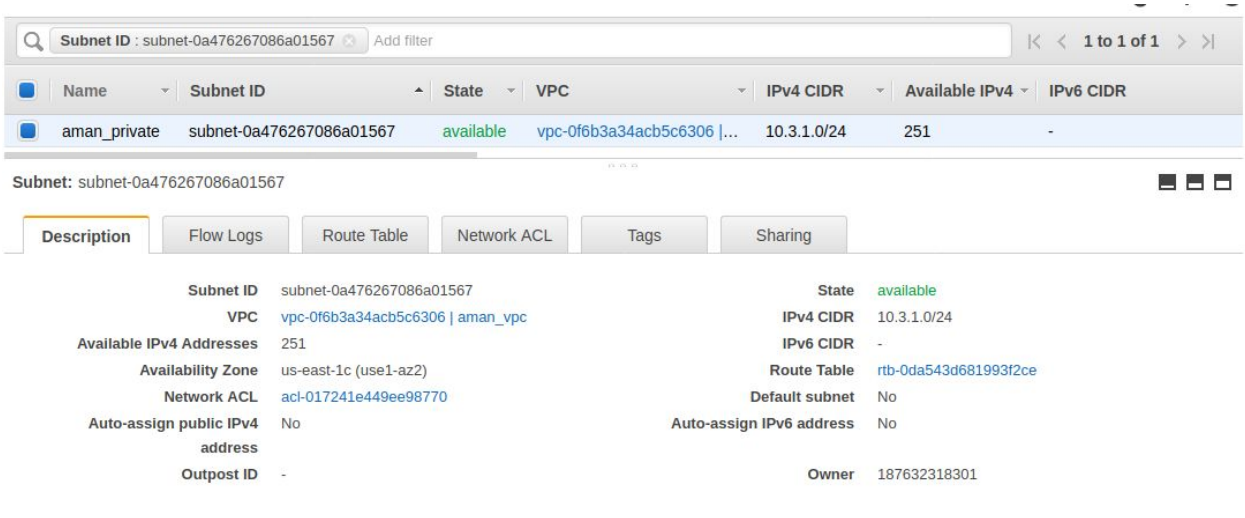
2. Create a public subnet, and host a web server(Nginx), also proxypass to Tomcat from Nginx

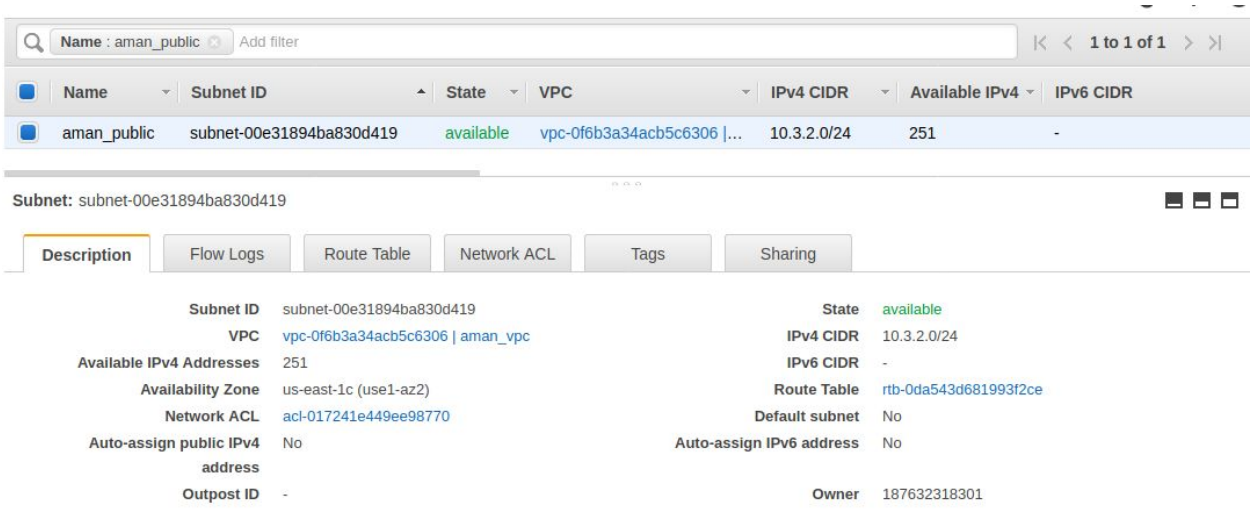
Answer

Step 1:- Create a VPC

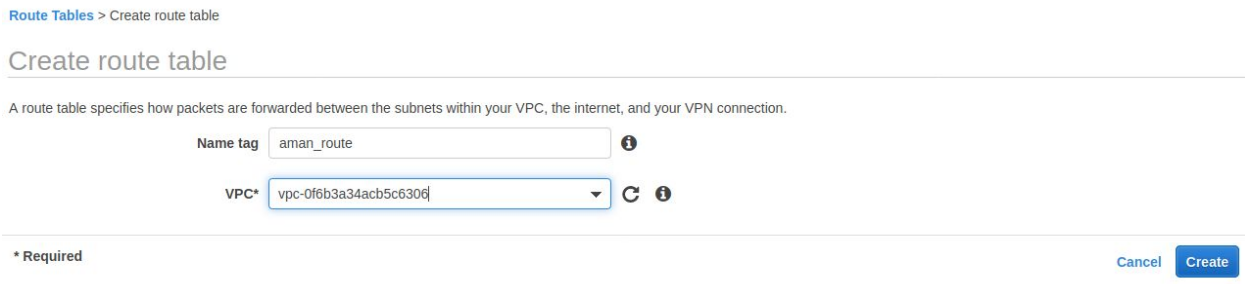


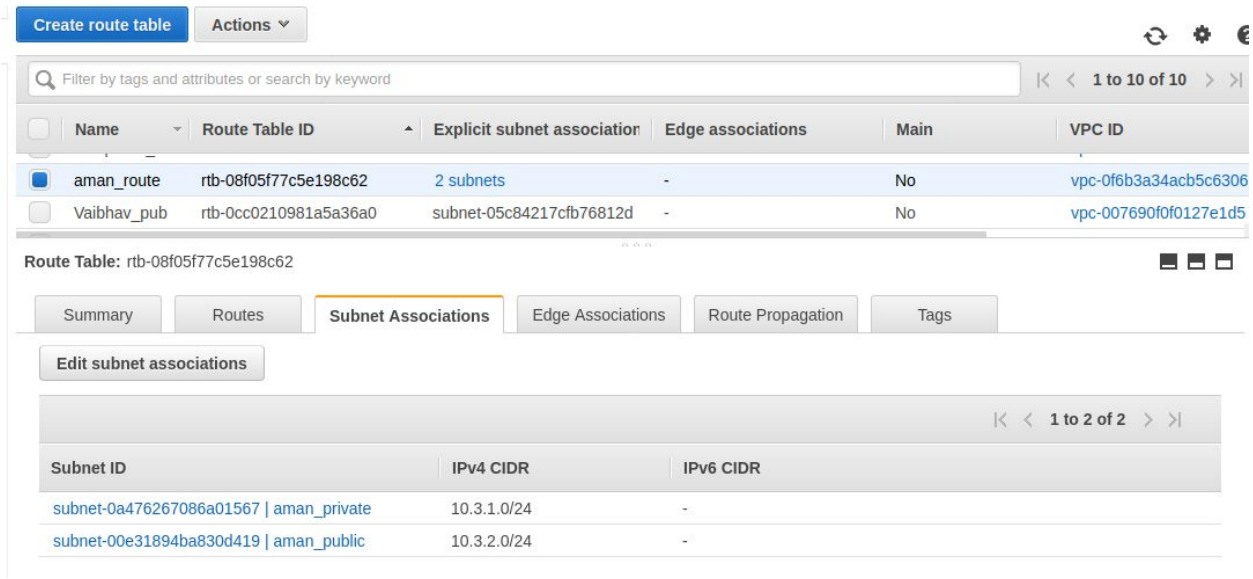
Step 2:- Create two Subnet (aman_private and aman_public)



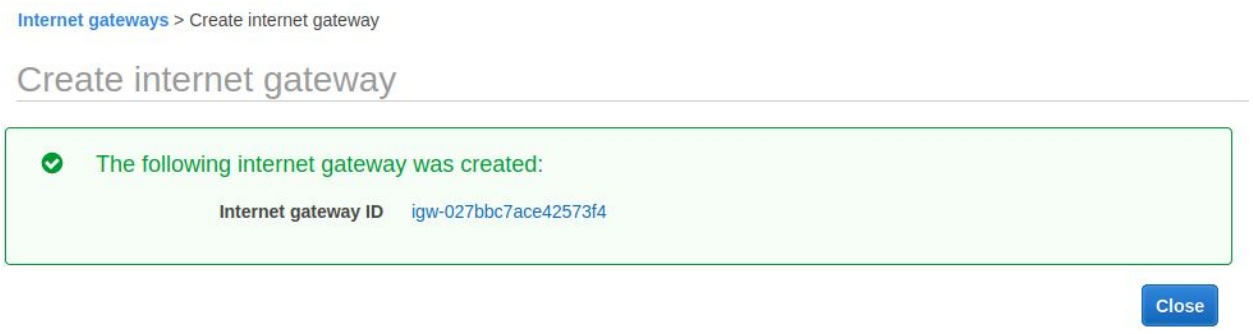


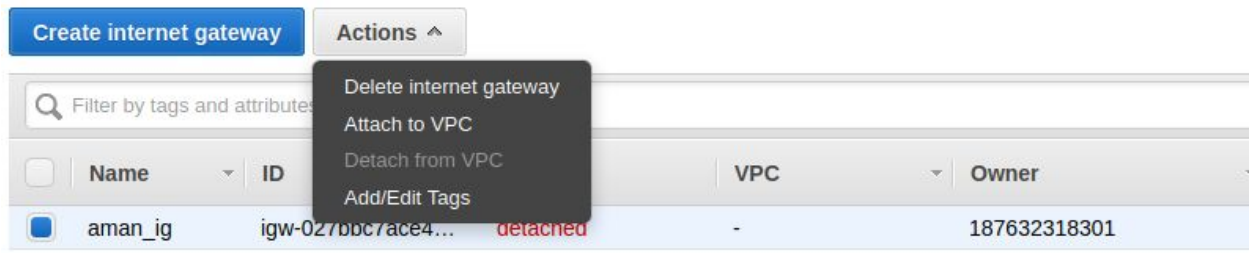
Step 3:- Create route tables





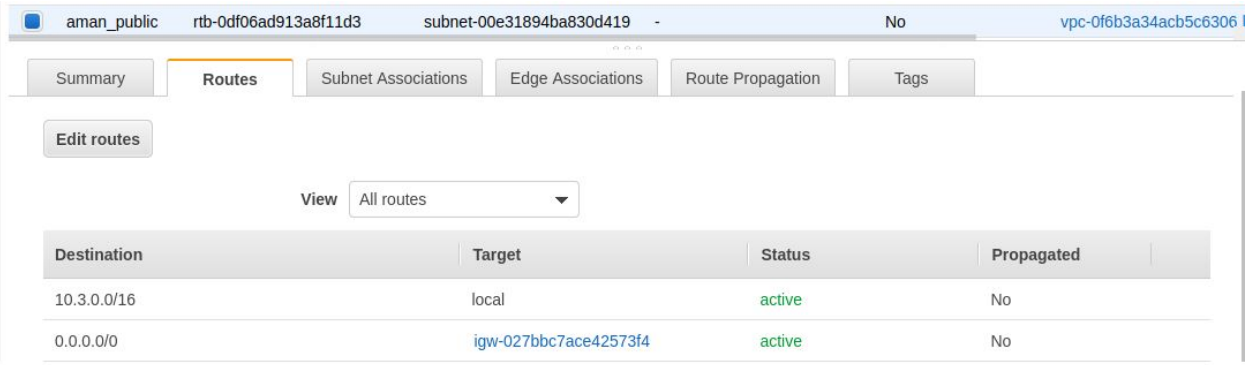
Step 4:- Create Internet Gateway (aman_ig)



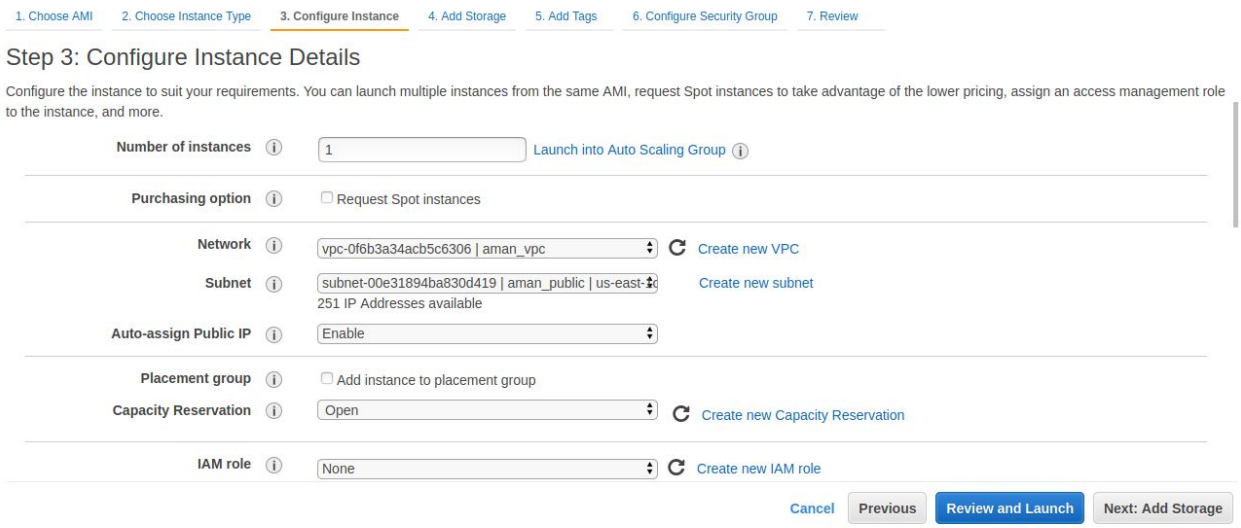


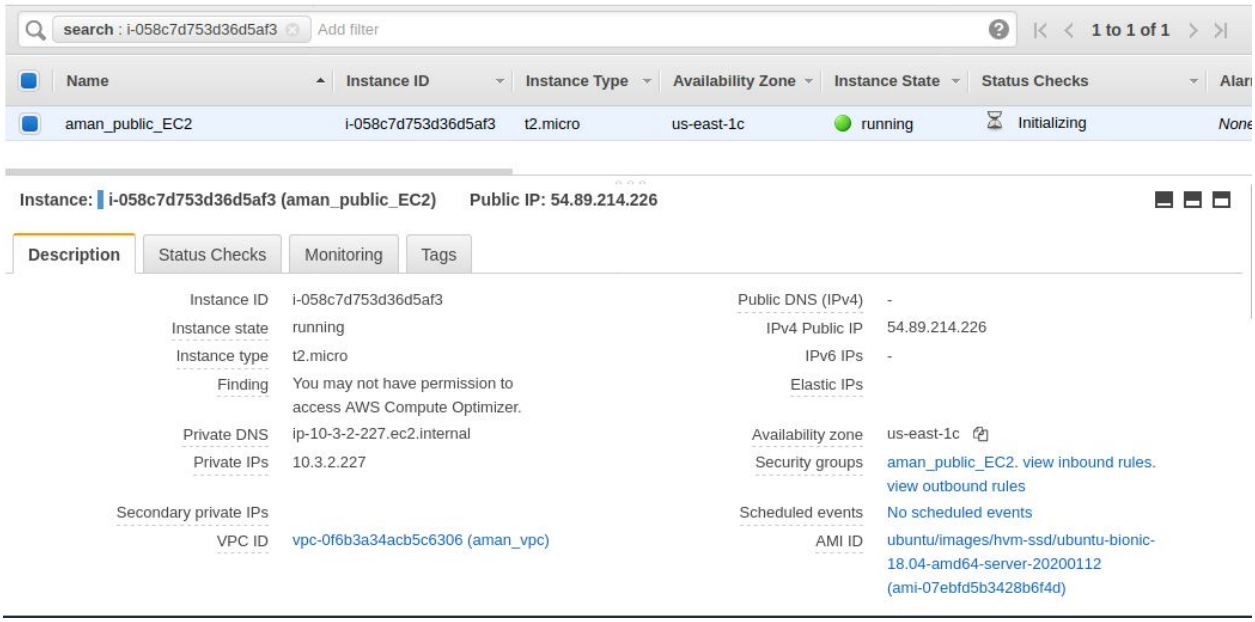
Step 5:- Attach Internet Gateway to VPC

Step 6:- Confiigure route table for public subnet and connect to Internet Gateway.

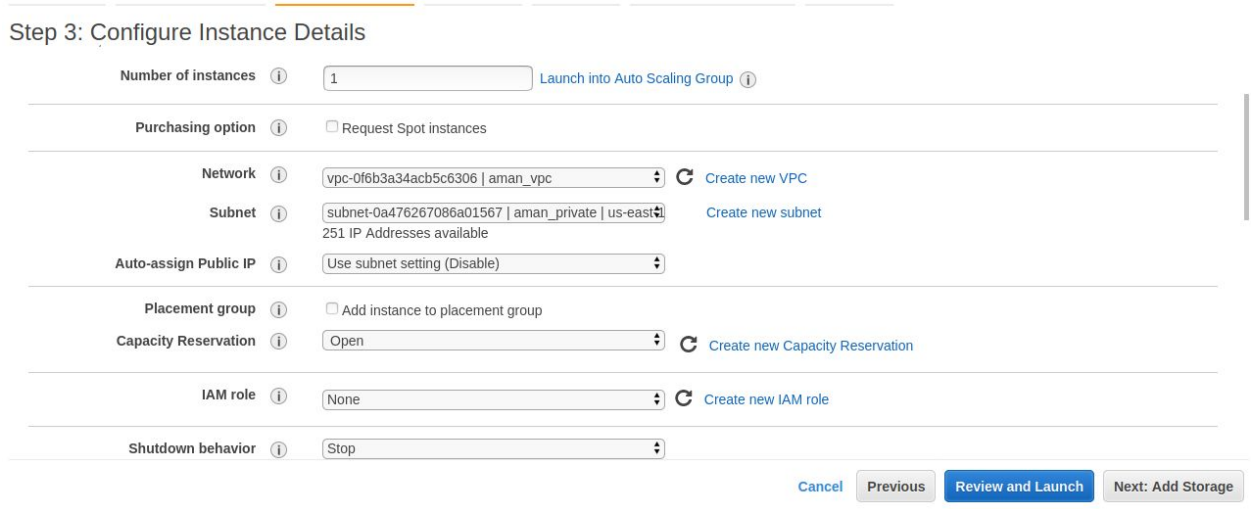


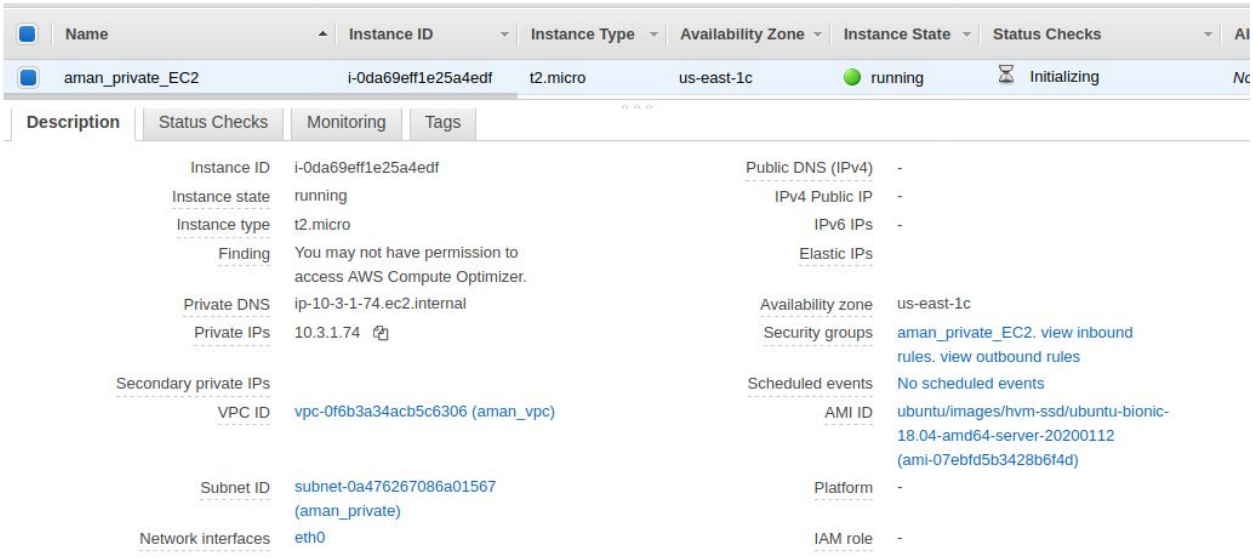
Step 7:- Create a Public Instance (aman_public_EC2)



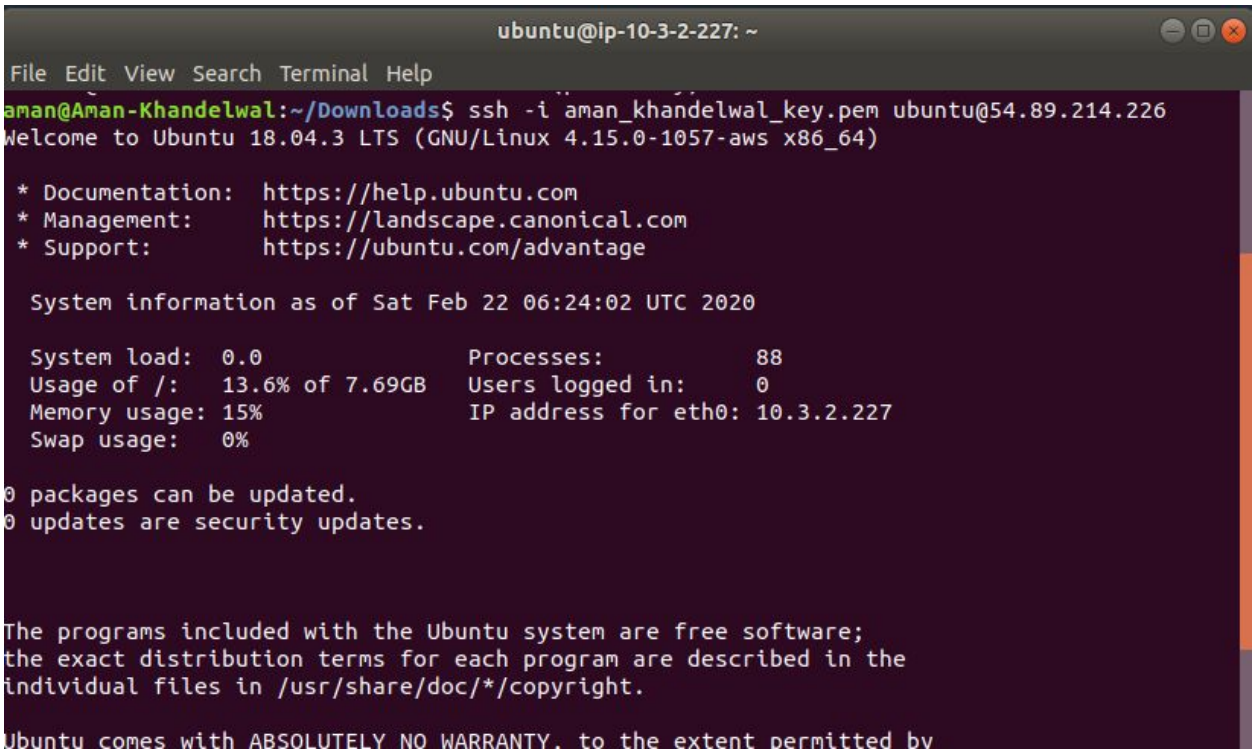


Step 8:- Create a Public Instance (aman_private_EC2)





Step 9:- Connect to aman_public_EC2 instance through ssh and configure nginx in it.



```
ubuntu@ip-10-3-2-227:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-backports InRelease [74.6 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/universe amd64 Packages [8570
kB]
Get:5 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/universe Translation-en [4941
kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/multiverse amd64 Packages [15
```

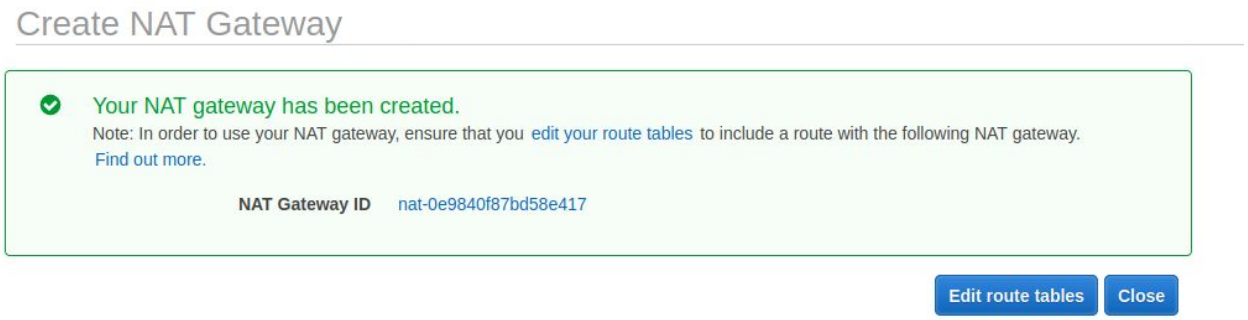
```
ubuntu@ip-10-3-2-227:~$ sudo apt-get install nginx
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  fontconfig-config fonts-dejavu-core libfontconfig1 libgd3 libjbig0 libjpeg-turbo8
  libjpeg8 libnginx-mod-http-geoip libnginx-mod-http-image-filter
  libnginx-mod-http-xslt-filter libnginx-mod-mail libnginx-mod-stream libtiff5 libwebp6
  libxpm4 nginx-common nginx-core
Suggested packages:
  libgd-tools fcgiwrap nginx-doc ssl-cert
```

Step 10:- Copy ssh key from local machine to aman_public_EC2 through SCP.

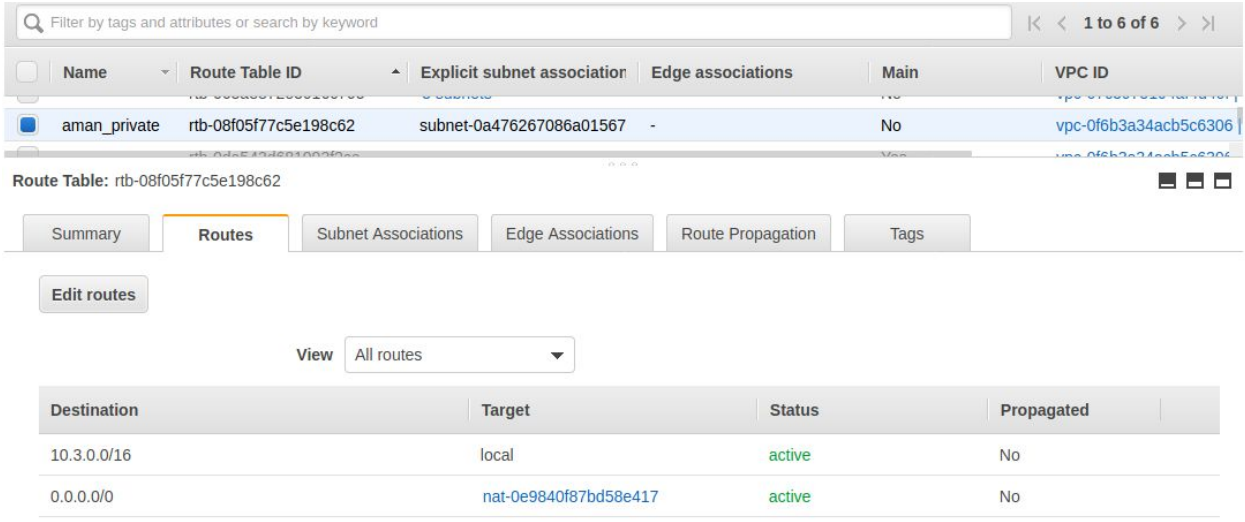
```
aman@Aman-Khandelwal:~/Downloads$ scp -i aman_khandelwal_key.pem aman_khandelwal_key.pem
 ubuntu@54.89.214.226:/home/ubuntu
aman_khandelwal_key.pem                       100% 1692     5.4KB/s   00:00
aman@Aman-Khandelwal:~/Downloads$
```

```
ubuntu@ip-10-3-2-227:~$ ls
aman_khandelwal_key.pem
ubuntu@ip-10-3-2-227:~$
```

Step 11:- Create NAT Gateway in Public Subnet.



Step 12:- Configure Nat Gateway Route in route table for private subnet.



Step 13:- Connect to Public Instance (aman_public_EC2) through SSH and from their connect to private instance (aman_private_EC2) in the private subnet.

```
ubuntu@ip-10-3-2-227:~$ ssh -i aman_khandelwal_key.pem ubuntu@10.3.1.74
The authenticity of host '10.3.1.74 (10.3.1.74)' can't be established.
ECDSA key fingerprint is SHA256:VMZSJxeCIgWIy3uSep/QBI4GCvtXNQfJXVTtQ2FDDcQ.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.3.1.74' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-1057-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sat Feb 22 06:49:13 UTC 2020

  System load:  0.0               Processes:           86
  Usage of /:   13.8% of 7.69GB   Users logged in:     0
  Memory usage: 17%               IP address for eth0: 10.3.1.74
  Swap usage:   0%

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

0 packages can be updated.
```

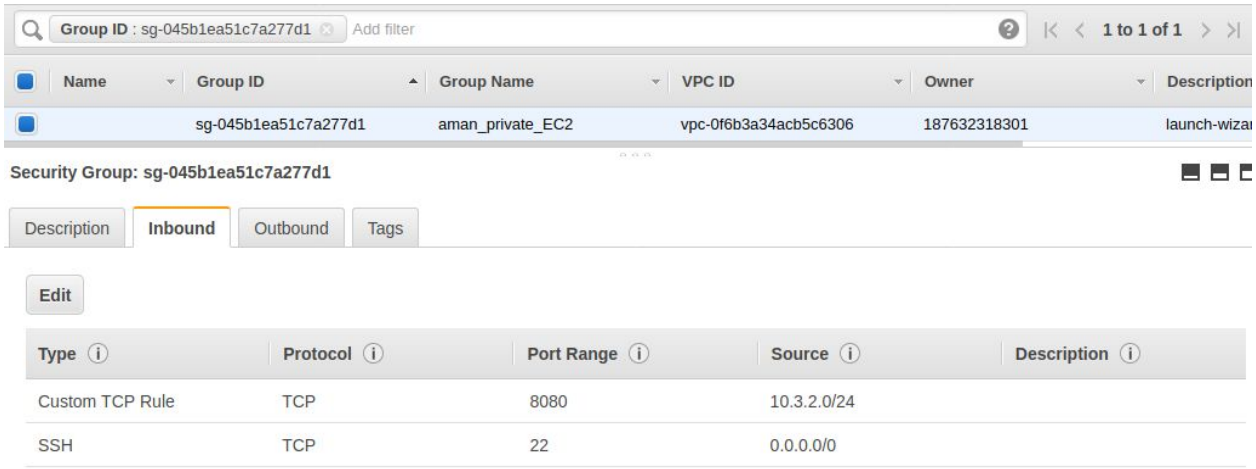
```
ubuntu@ip-10-3-1-74:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7
kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic-backports InRelease [74
6 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/universe amd64 Packages
[8570 kB]
Get:5 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu bionic/universe Translation-en
```

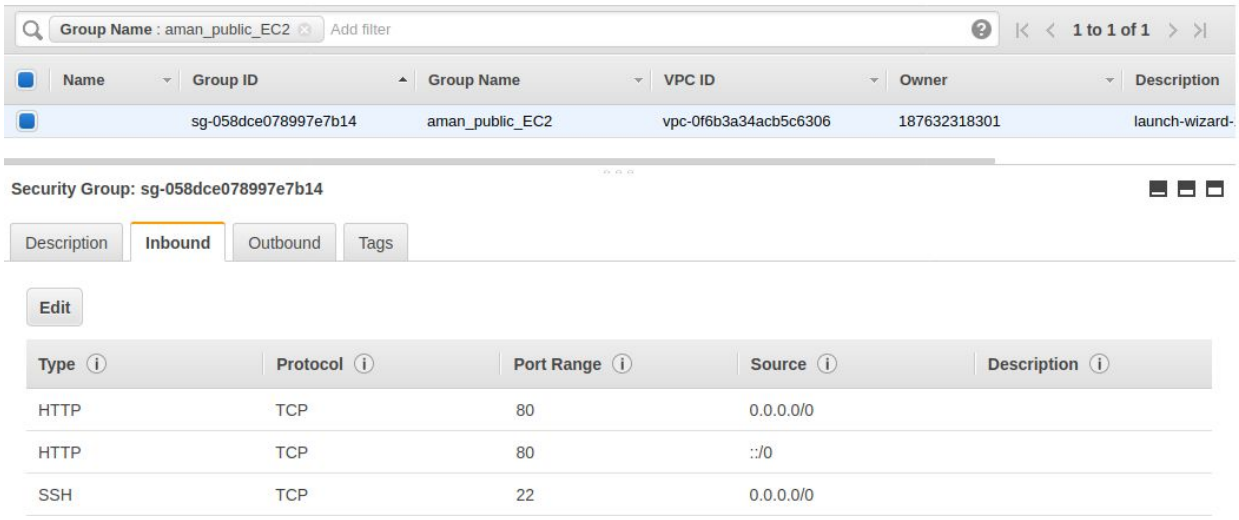
Step 14:- Install and configure tomcat9 on private subnet.

```
ubuntu@ip-10-3-1-74:~$ sudo apt-get install tomcat9
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java default-jre-headless fontconfig-config fonts-dejavu-core
  java-common libapr1 libasound2 libasound2-data libavahi-client3
  libavahi-common-data libavahi-common3 libcups2 libeclipse-jdt-core-java
  libfontconfig1 libjpeg-turbo8 libjpeg8 liblcms2-2 libnspr4 libnss3 libpcsclite1
  libtcnative-1 libtomcat9-java libxi6 libxrender1 libxtst6
  openjdk-11-jre-headless tomcat9-common x11-common
Suggested packages:
  default-jre libasound2-plugins alsa-utils cups-common liblcms2-utils pcscd
  libnss-mdns fonts-dejavu-extra fonts-ipafont-gothic fonts-ipafont-mincho
  fonts-wqy-microhei | fonts-wqy-zenhei fonts-indic tomcat9-admin tomcat9-docs
```

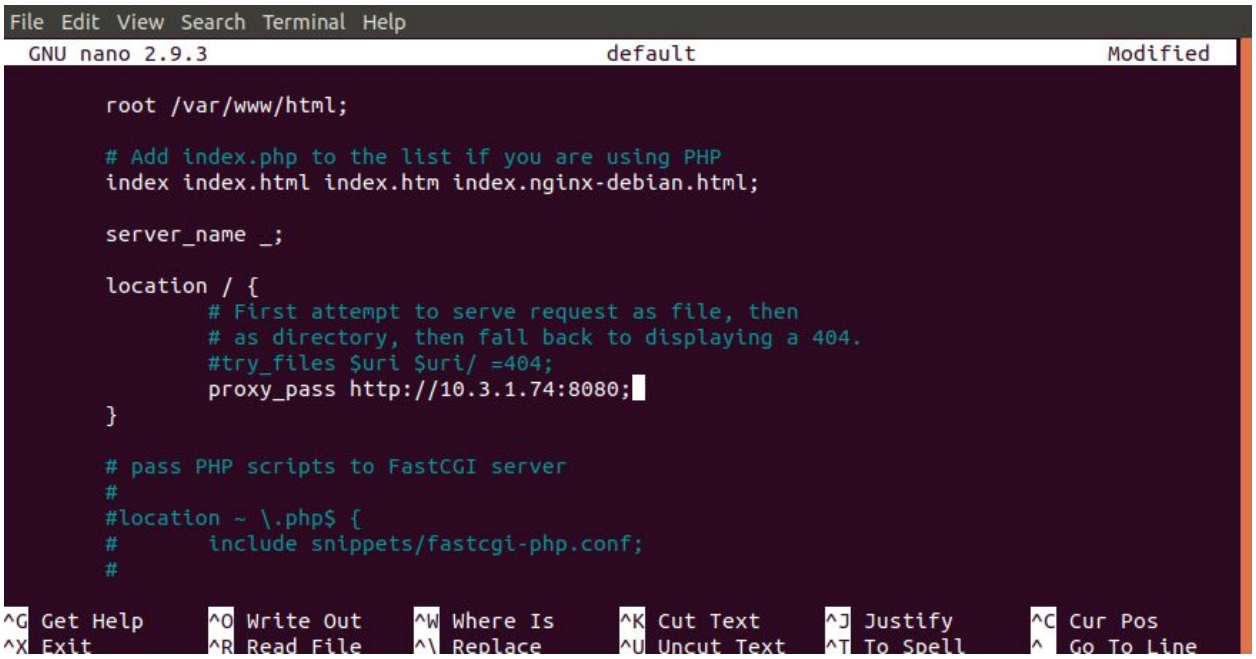
```
ubuntu@ip-10-3-1-74:~$ curl 127.0.0.1:8080
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
    <title>Apache Tomcat</title>
</head>

<body>
<h1>It works !</h1>

<p>If you're seeing this page via a web browser, it means you've setup Tomcat succes
sfully. Congratulations!</p>

<p>This is the default Tomcat home page. It can be found on the local filesystem at:
 <code>/var/lib/tomcat9/webapps/ROOT/index.html</code></p>

<p>Tomcat veterans might be pleased to learn that this system instance of Tomcat is
installed with <code>CATALINA_HOME</code> in <code>/usr/share/tomcat9</code> and <co
de>CATALINA_BASE</code> in <code>/var/lib/tomcat9</code>, following the rules from <
code>/usr/share/doc/tomcat9-common/RUNNING.txt.gz</code>.</p>
```

Step 15:- Configure Security Group of public instance to allow request on port no. 80, and for private instance to allow request on port 8080.





Step 16:- Configure nginx web server for proxy_pass request to the tomcat server at port 8080.



```
ubuntu@ip-10-3-2-227:~$ cd /etc/nginx/sites-available/
ubuntu@ip-10-3-2-227:/etc/nginx/sites-available$ sudo nano default
ubuntu@ip-10-3-2-227:/etc/nginx/sites-available$ sudo service nginx restart
ubuntu@ip-10-3-2-227:/etc/nginx/sites-available$
```

Step 17:- Now Open the public IP on browser.

It works !

If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

This is the default Tomcat home page. It can be found on the local filesystem at: `/var/lib/tomcat9/webapps/ROOT/index.html`

Tomcat veterans might be pleased to learn that this system instance of Tomcat is installed with `CATALINA_HOME` in `/usr/share/tomcat9` and `CATALINA_BASE` in `/var/lib/tomcat9`, following the rules from `/usr/share/doc/tomcat9-common/RUNNING.txt.gz`.

You might consider installing the following packages, if you haven't already done so:

tomcat9-docs: This package installs a web application that allows to browse the Tomcat 9 documentation locally. Once installed, you can access it by clicking here.

tomcat9-examples: This package installs a web application that allows to access the Tomcat 9 Servlet and JSP examples. Once installed, you can access it by clicking here.

tomcat9-admin: This package installs two web applications that can help managing this Tomcat instance. Once installed, you can access the manager webapp and the host-manager webapp.

NOTE: For security reasons, using the manager webapp is restricted to users with role "manager-gui". The host-manager webapp is restricted to users with role "admin-gui". Users are defined in `/etc/tomcat9/tomcat-users.xml`.