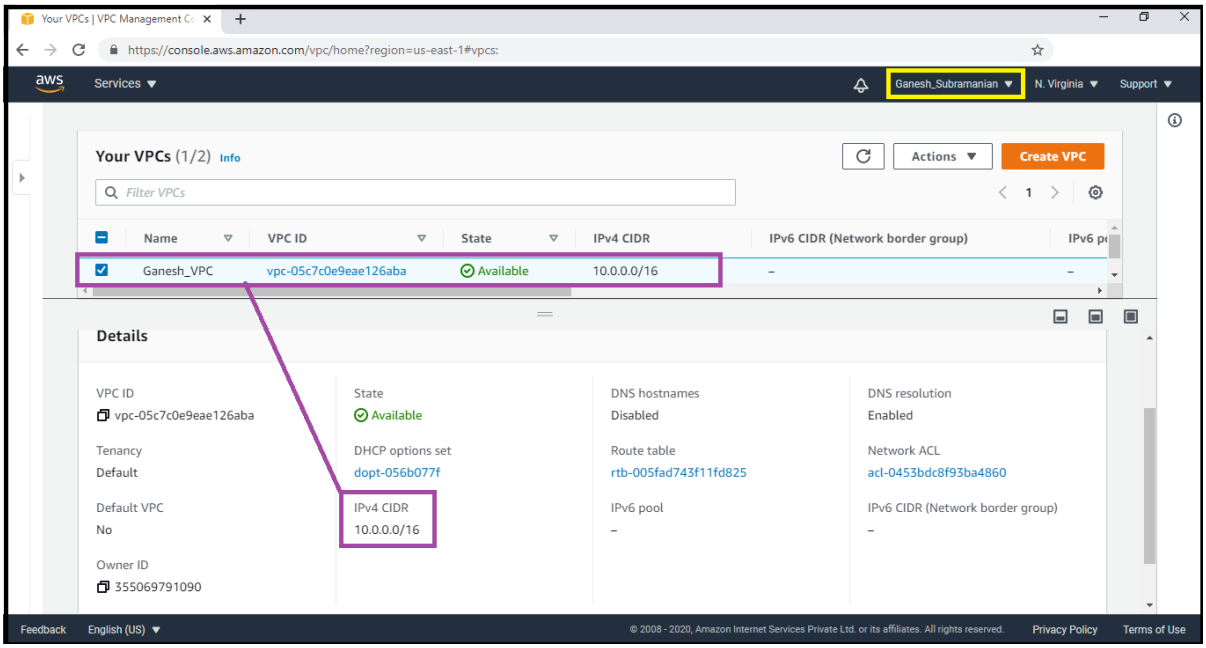
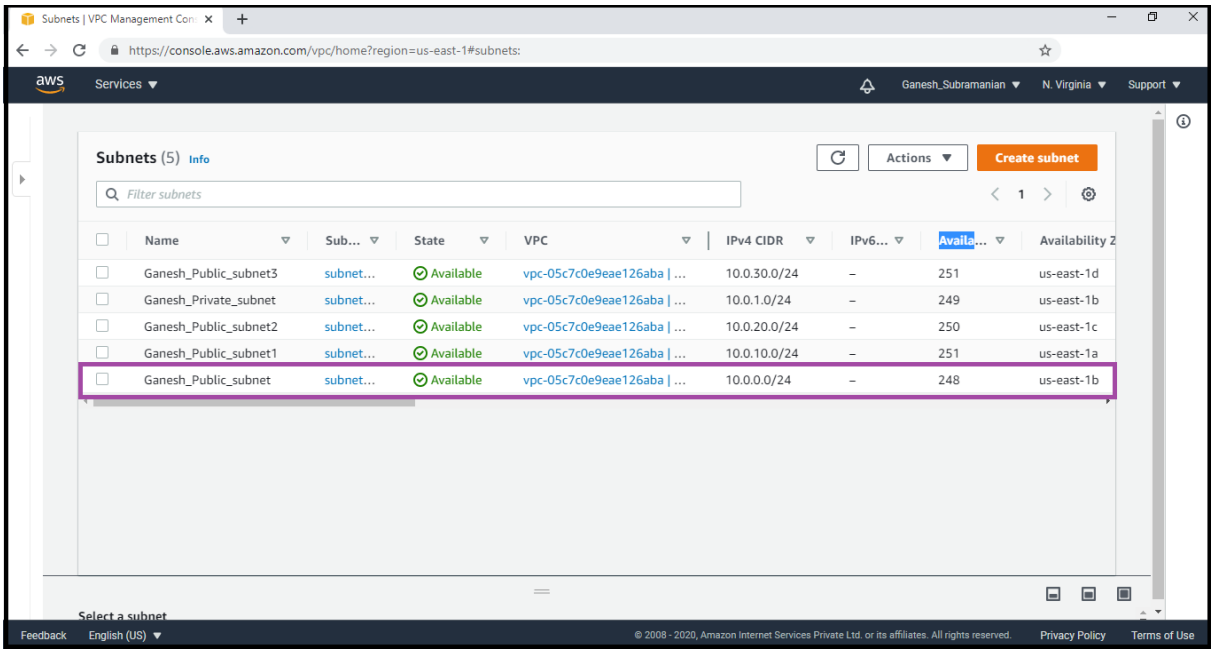
Project 1

➔ **Create VPC: -**



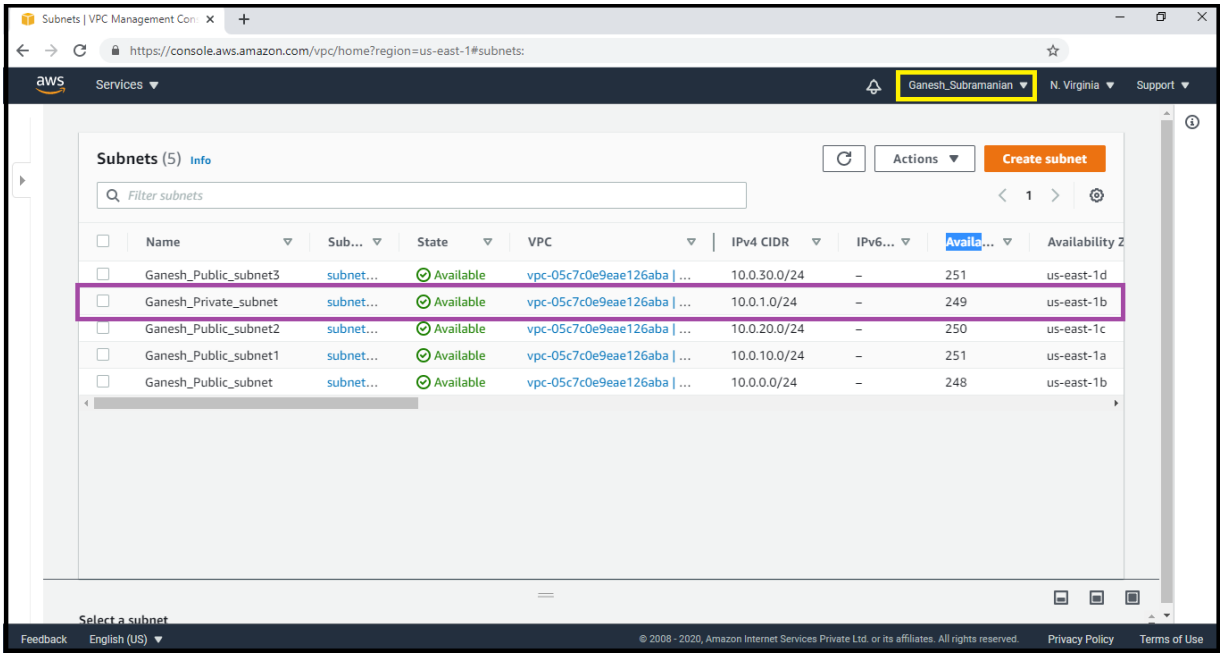
VPC Name: - "**Ganesh_VPC**"
IPv4 CIDR Block: - **10.0.0.0/16**

➔ **Create Subnet (Public): -**



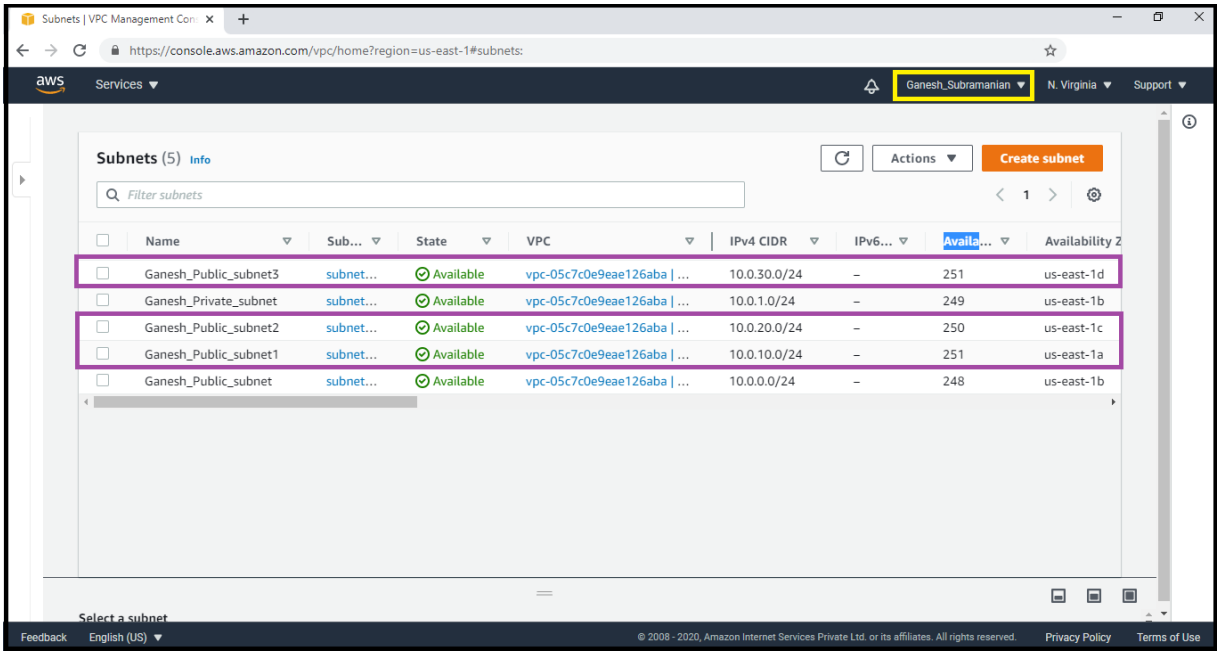
Public Subnet Name: - "**Ganesh_Public_subnet**"
VPC: - "**Ganesh_VPC**"
IPv4 CIDR Block: - "**10.0.0.0/24**"

➔ **Create Subnet (Private): -**



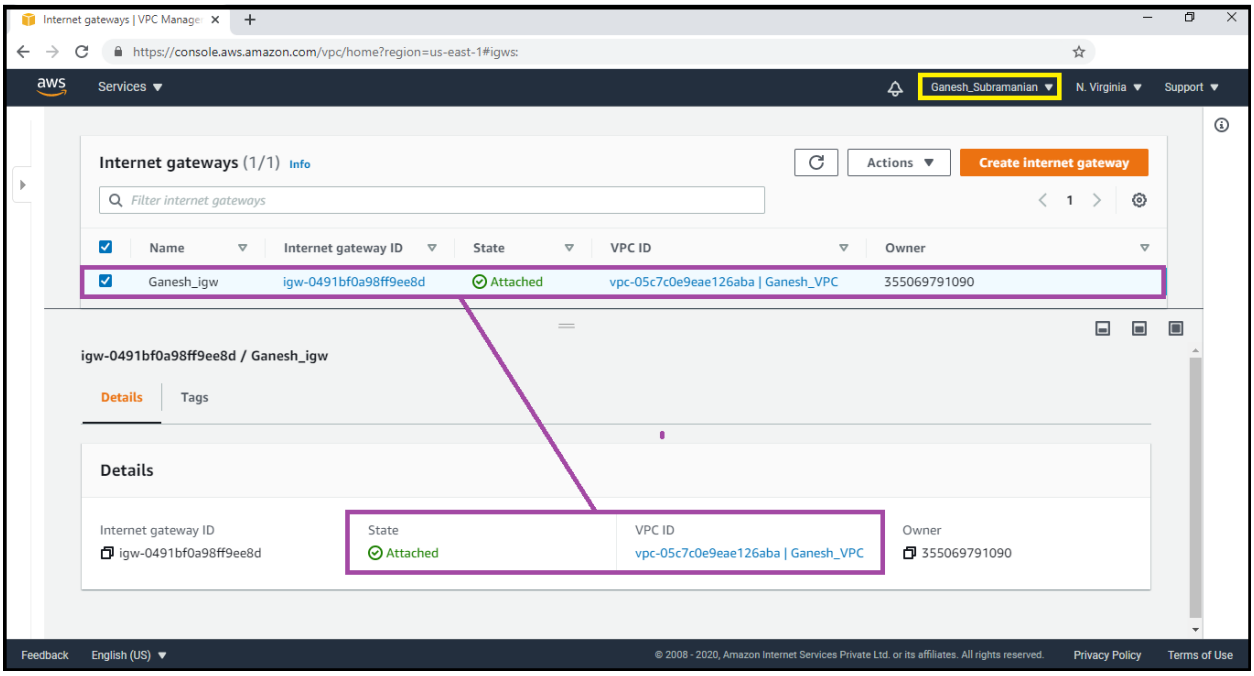
Private Subnet Name: - "**Ganesh_Private_subnet**"
VPC: - "**Ganesh_VPC**"
IPv4 CIDR Block: - "**10.0.1.0/24**"

➔ **Create 3 Public subnets in 3 different Availability Zone's in the same VPC: -**



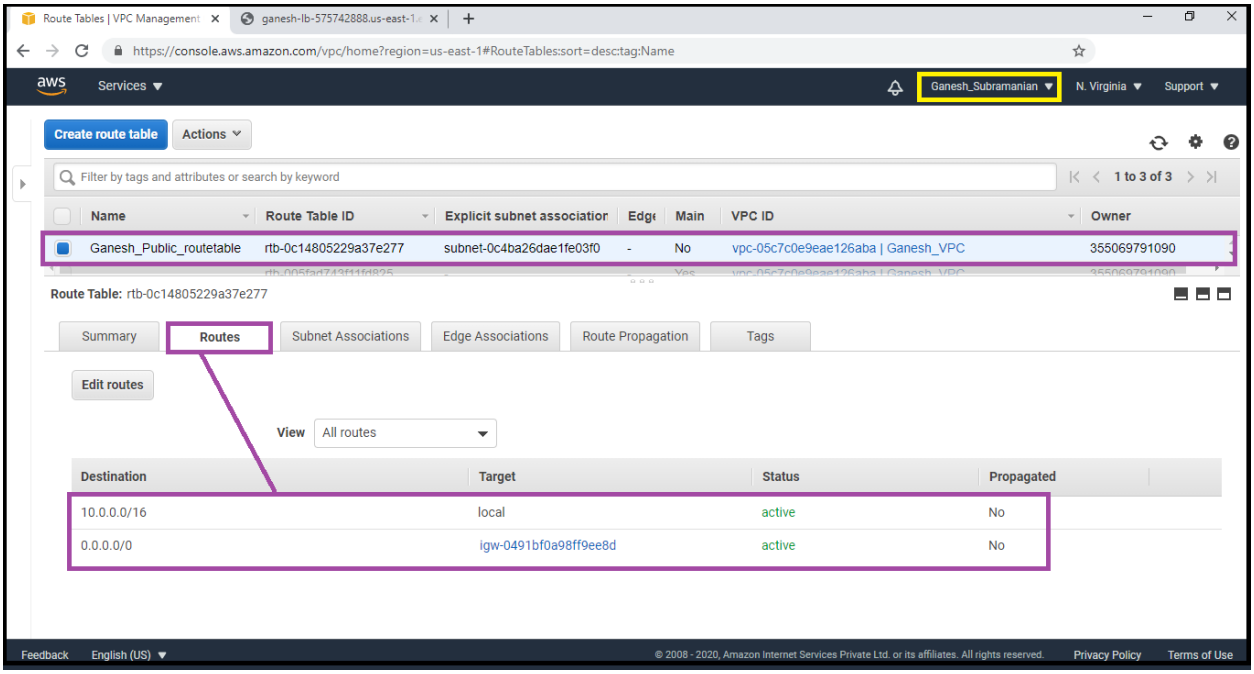
Public Subnet 1 (us-east-1a): - "**Ganesh_Public_subnet1**" IPv4 CIDR - **10.0.10.0/24**
Public Subnet 2 (us-east-1c): - "**Ganesh_Public_subnet2**" IPv4 CIDR - **10.0.20.0/24**
Public Subnet 3 (us-east-1d): - "**Ganesh_Public_subnet3**" IPv4 CIDR - **10.0.30.0/24**

➔ **Create a IGW and associate with the Public Subnet: -**

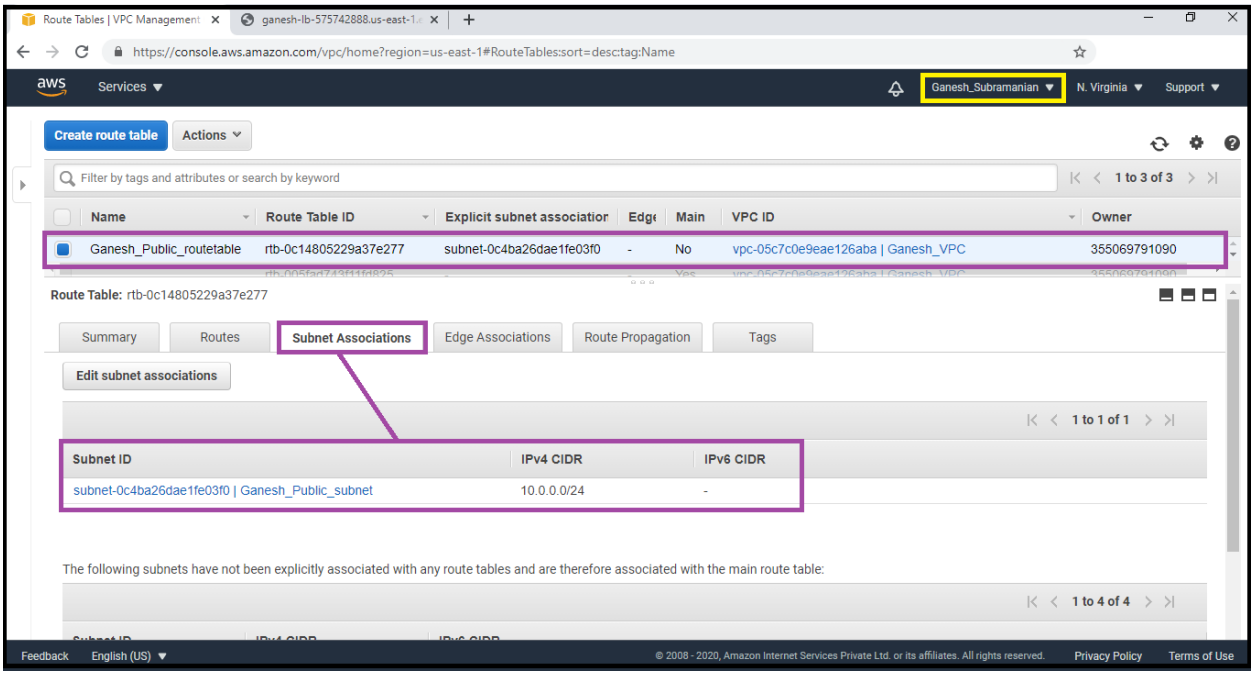


IGW Name: – "**Ganesh_igw**"
Attached to VPC: - "**Ganesh_VPC**"

➔ **Create Route Table: -**

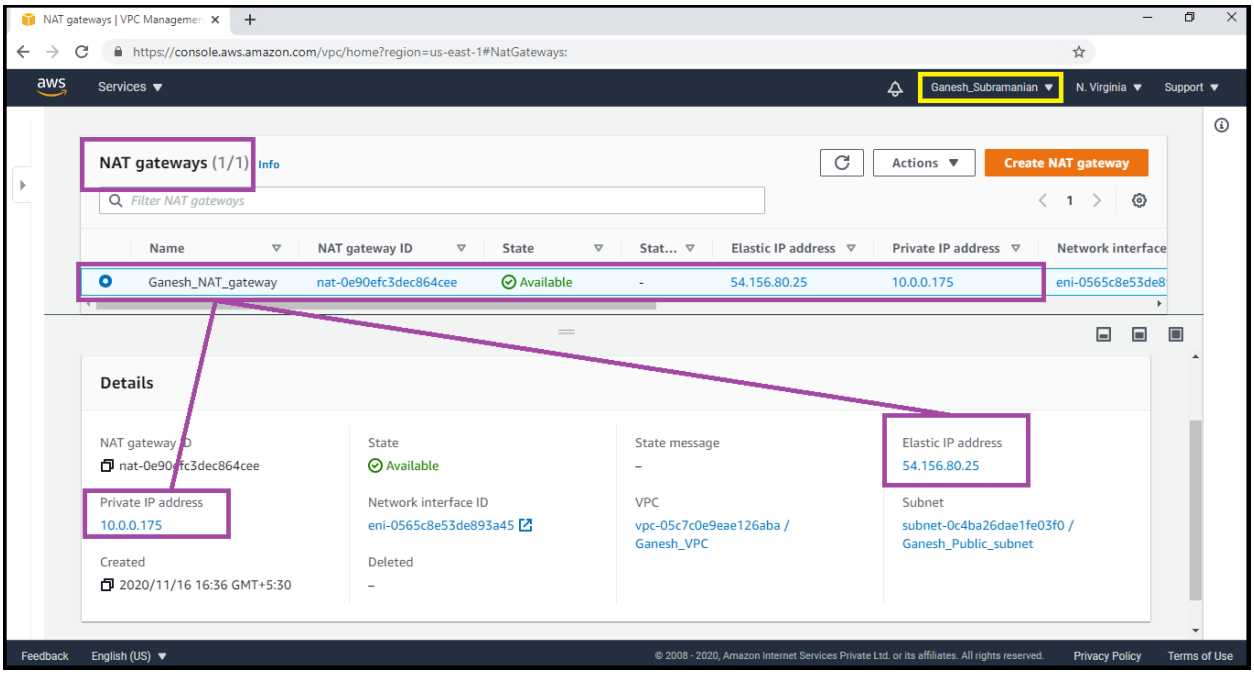


Route Table Name: - "**Ganesh_Public_routetable**"
Routes / Edit Routes / Add Routes: - Destination: - "**0.0.0.0/0**" & Target: - "**igw-0491bf0a98ff9ee8d**"



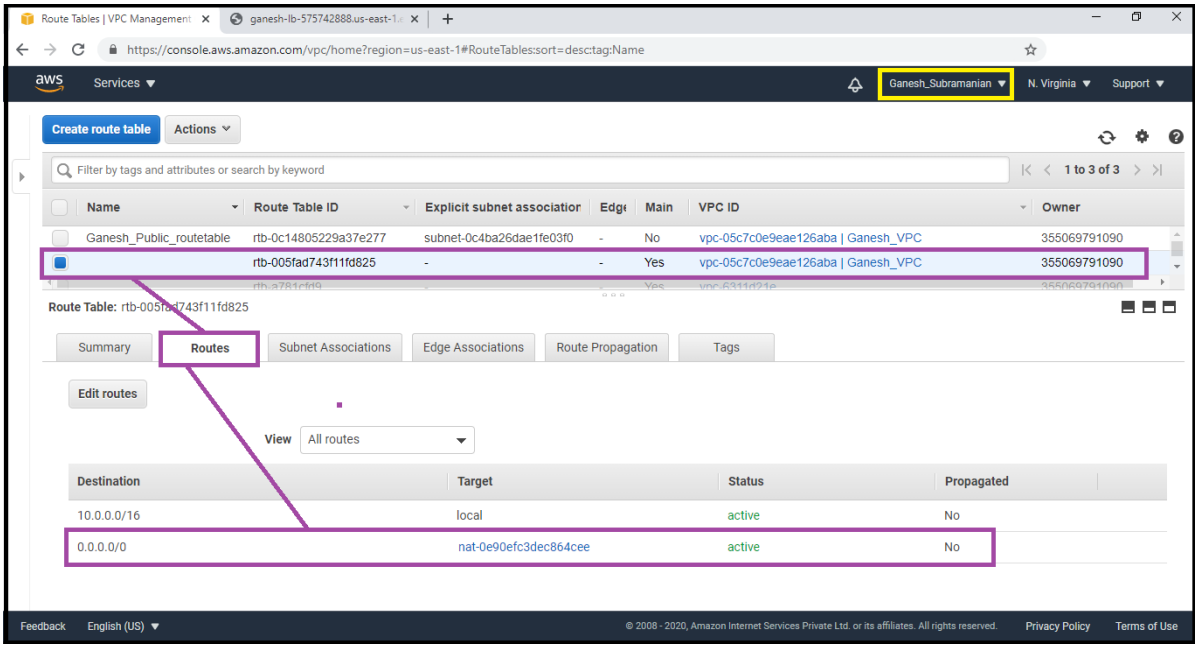
Edit Subnet Associations: - Select "**Ganesh_Public_subnet**"
Make Sure This Route **Should NOT be the Main** Route Table
The IGW Status is **Active**

➔ **Create a NATGateway and associate with public subnet: -**



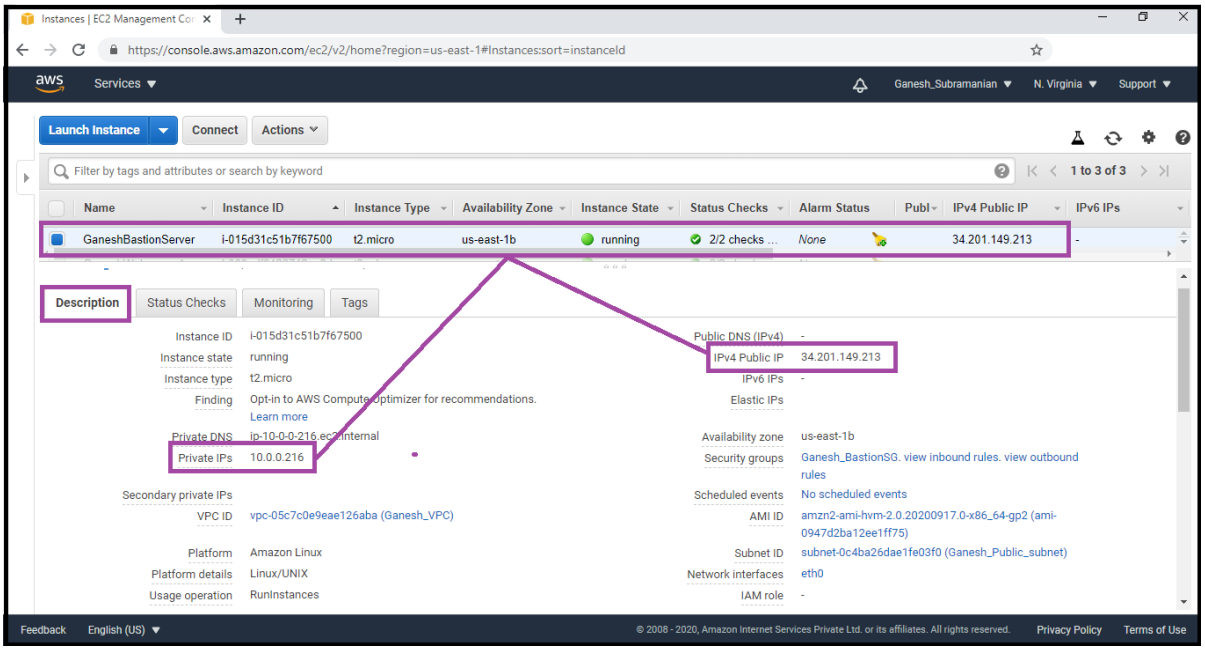
NAT Gateway Name: - **"Ganesh_NAT_gateway"**
Allocated Elastic IP: - "**54.156.80.25**"

➔ **Edit Route Table: -**



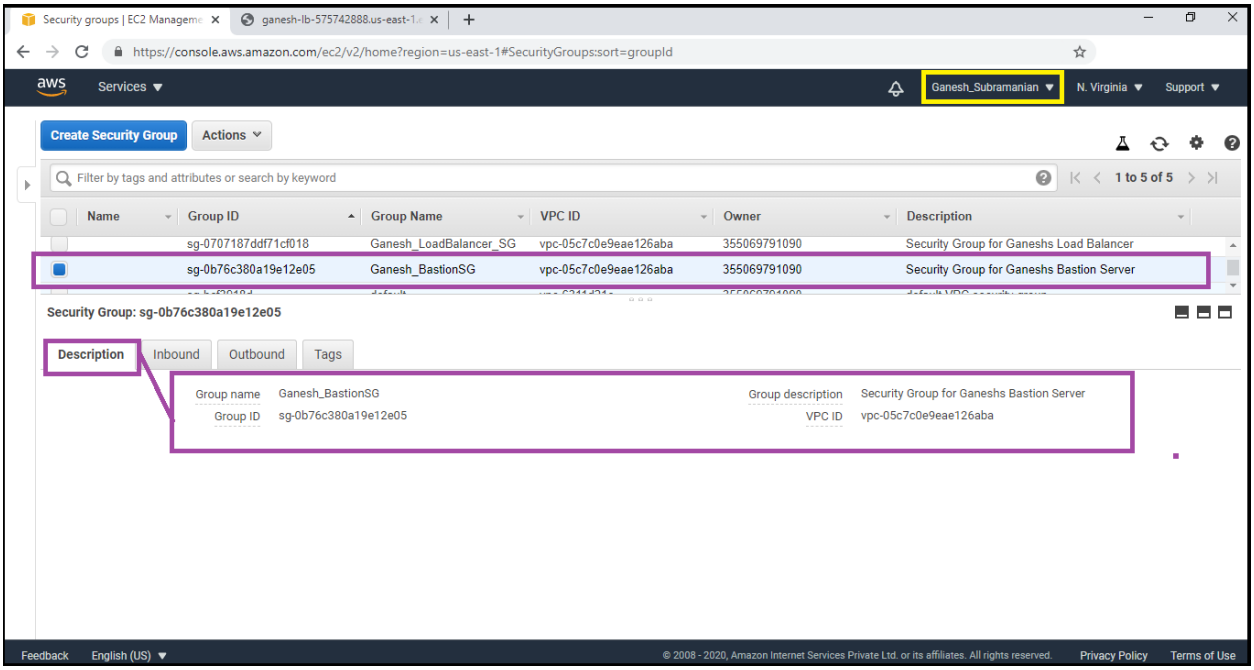
Edited Route Table: - "**Main Route Table**"
Routes / Edit Routes / Add Routes: - Destination: - "**0.0.0.0/0"** & Target: - "**nat-0e90efc3dec864cee"**
The NATGateway Status is **Active**

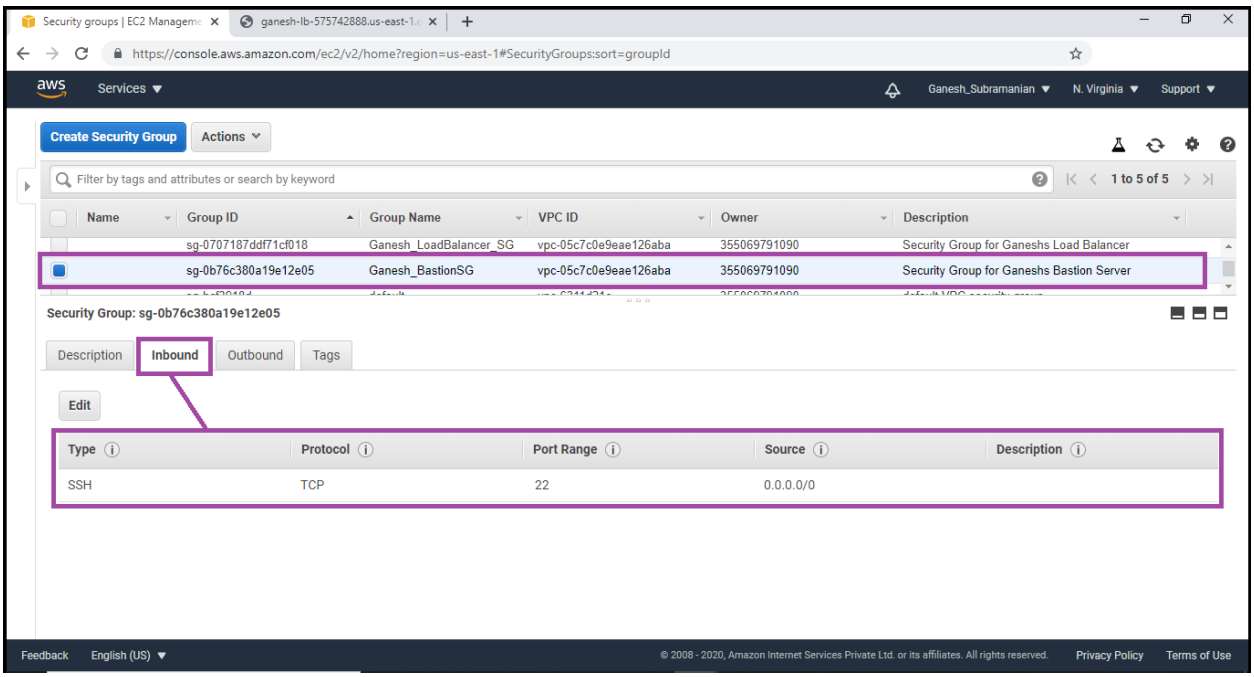
➔ **Create a Bastion Server Instance in Public Subnet: -**



Bastion Server Name: - **"GaneshBastionServer"**
Private IP: - **10.0.0.216**
Public IP: - **34.201.149.213**

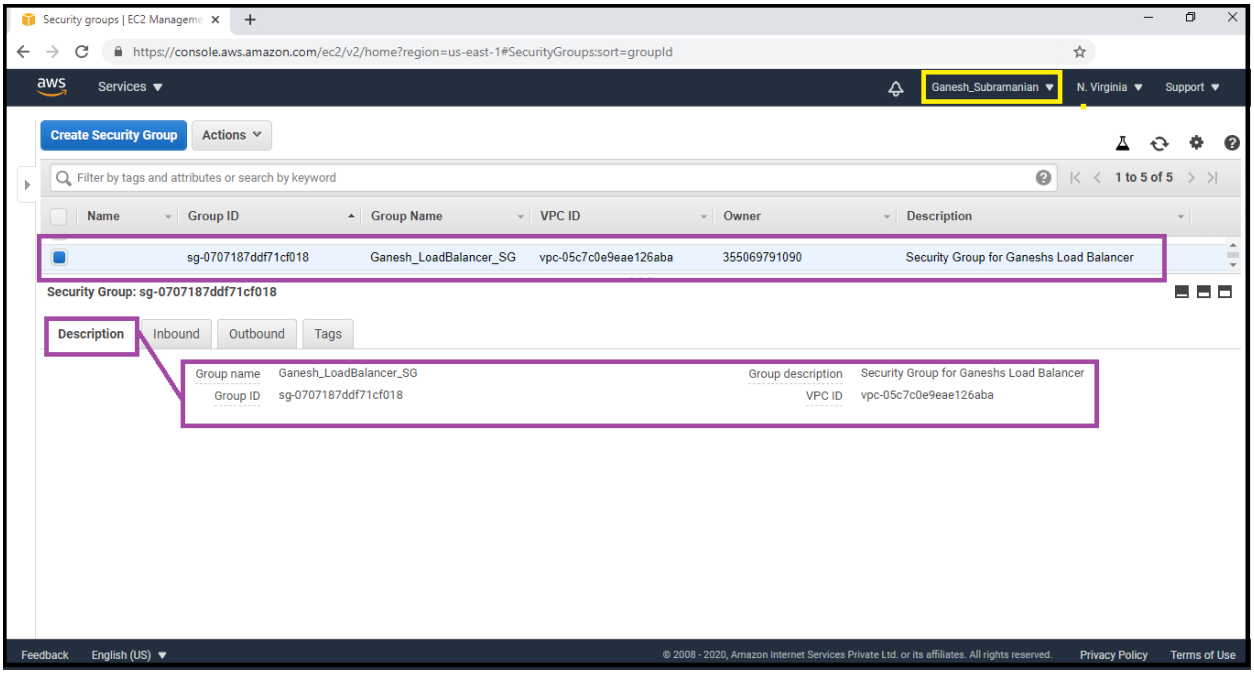
➔ **Create a New Security Group for Bastion Server: -**

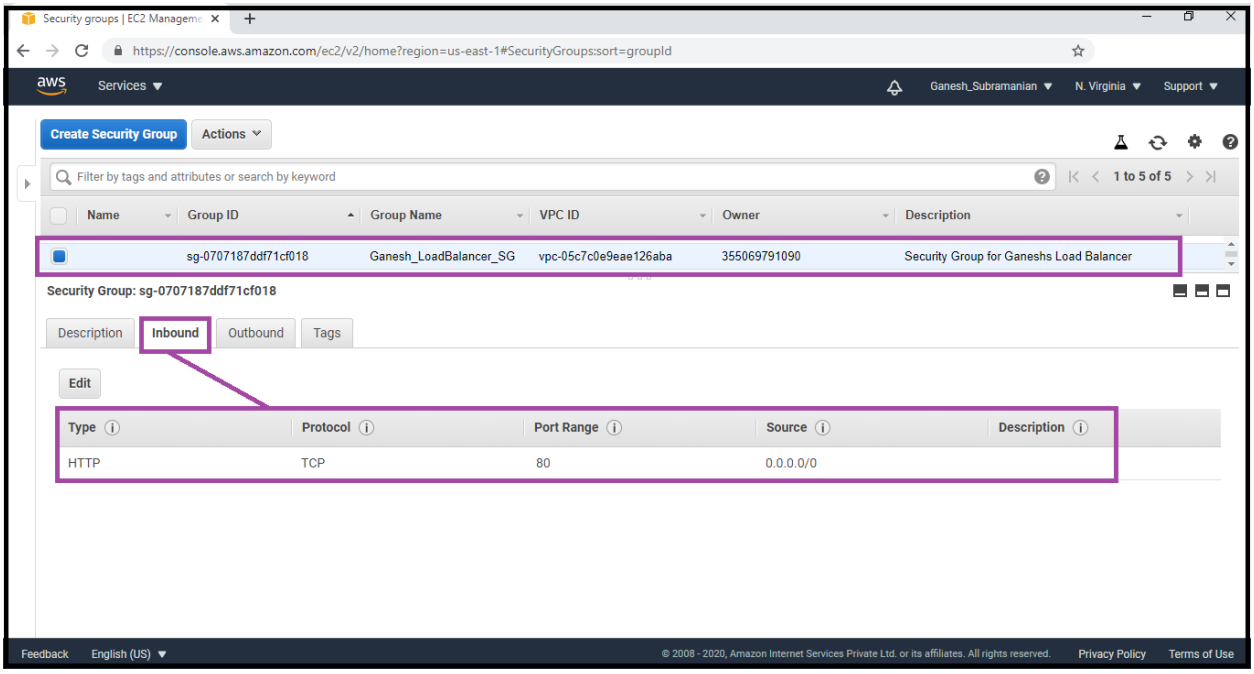




Security Group Name: - **"Ganesh_BastionSG"**
Inbound Rules: - Type – **SSH** & Source – **0.0.0.0/0**

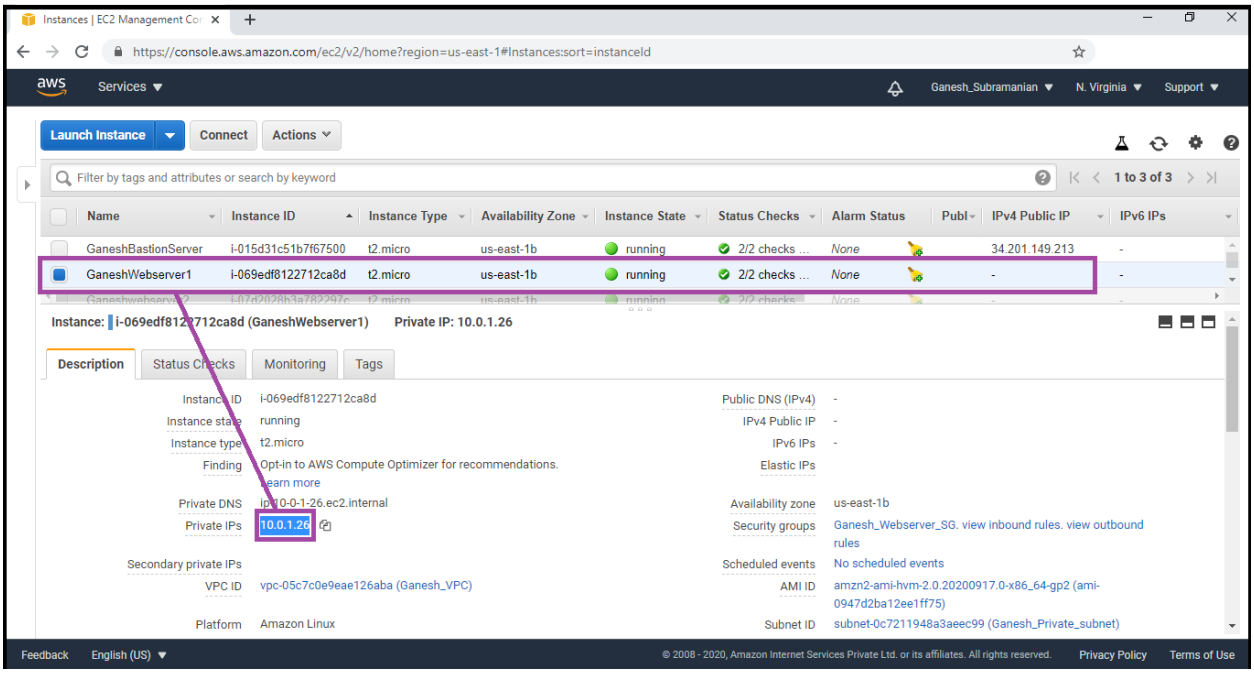
➔ **Create Security group for Load Balancer: -**



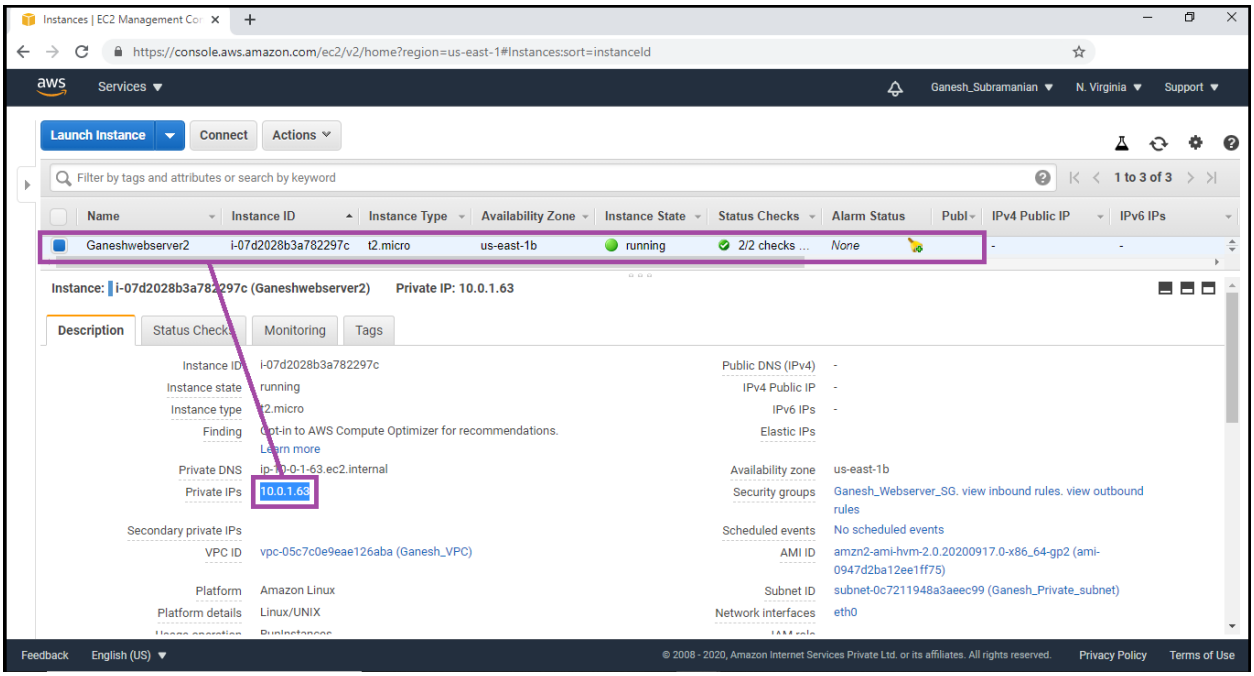


Security Group Name: - **"Ganesh_LoadBalancer_SG"**
Inbound Rules: - Type – **HTTP** & Source – **0.0.0.0/0**

➔ **Launch 2 Webservers in the Private Subnet: -**

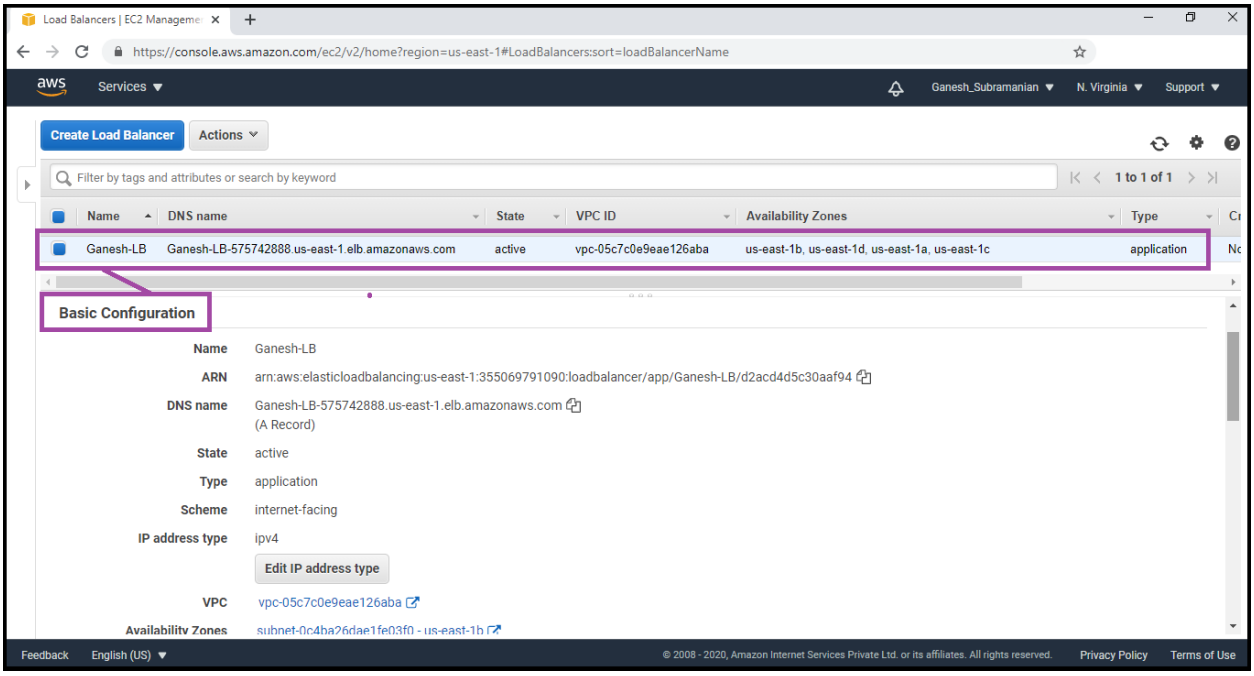


Webserver1 Name: - "**GaneshWebsever1**"
Private IP: - **10.0.1.26**
Public IP: - **---------**



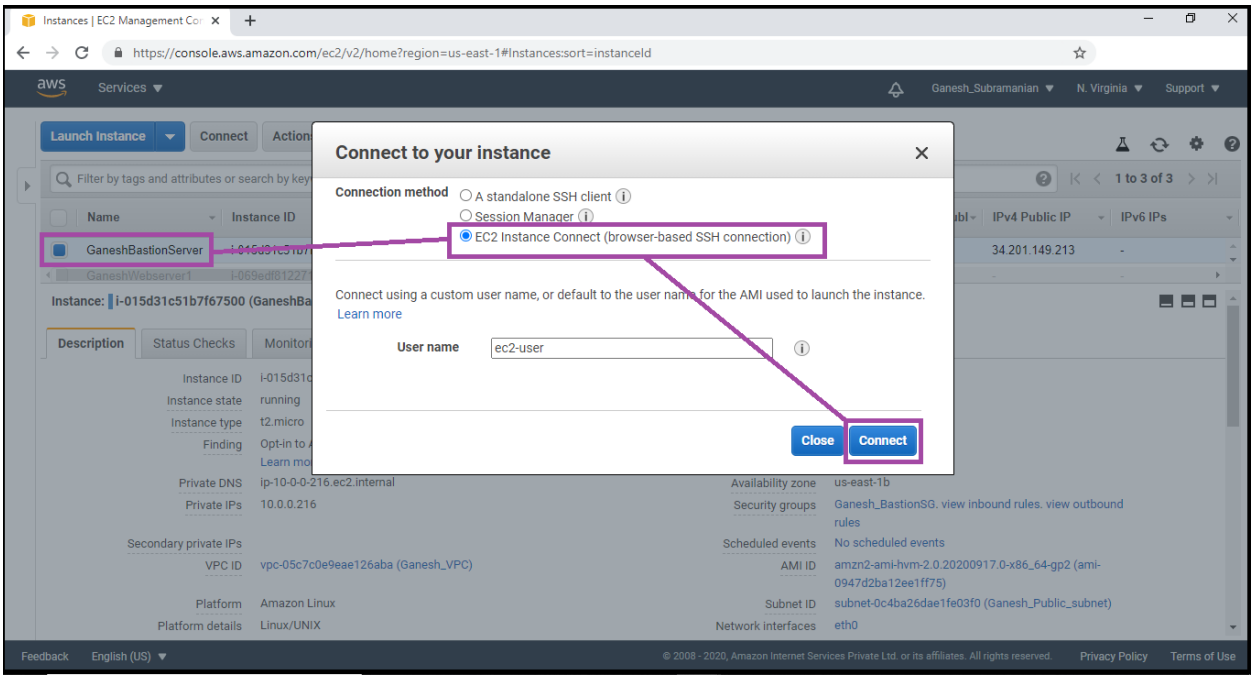
Webserver2 Name: - "**Ganeshwebsever2**"
Private IP: - **10.0.1.63**
Public IP: - **---------**

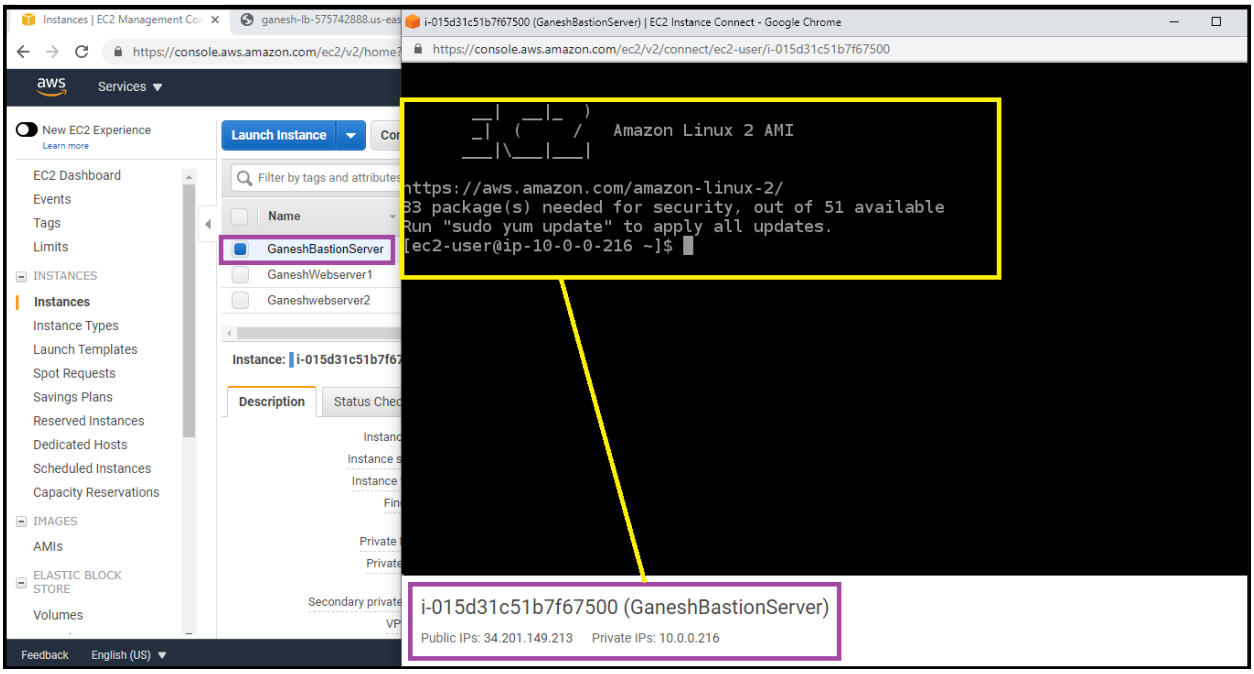
➔ **Create Load Balancer: -**



Load Balancer Name: – "**Ganesh-LB**"

➔ **Launch bastion Server: -**

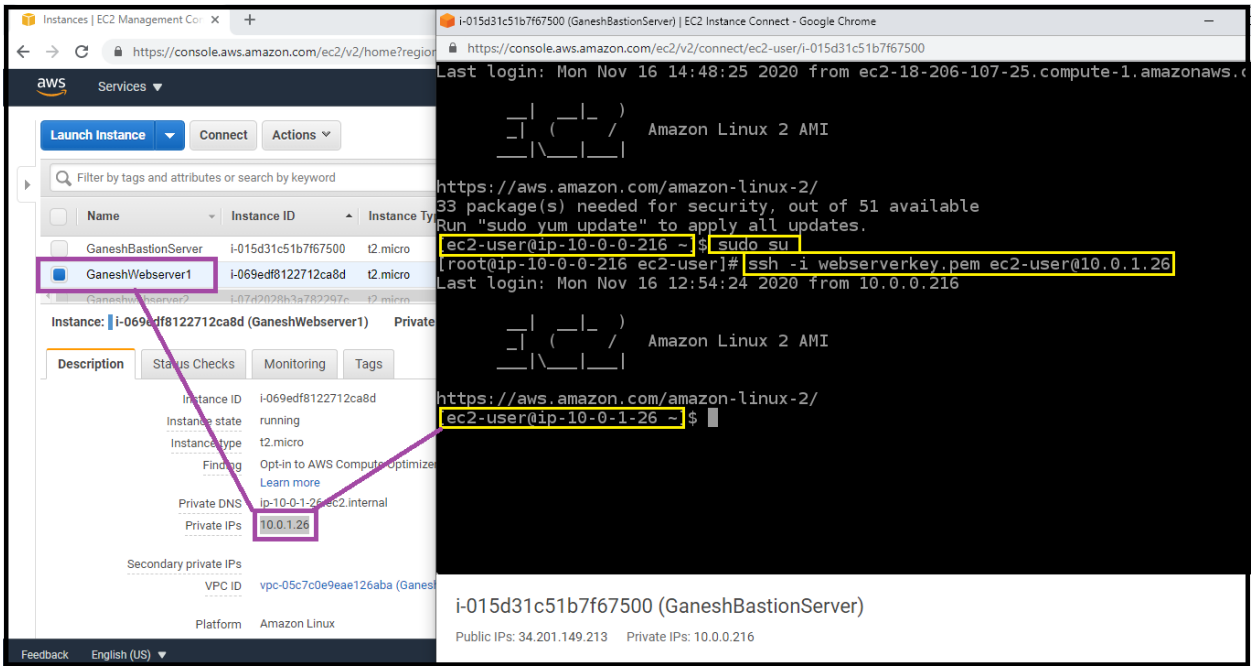




Successful Launching of Bastion Server "**GaneshBastionServer"**

Connecting to web server via Bastion

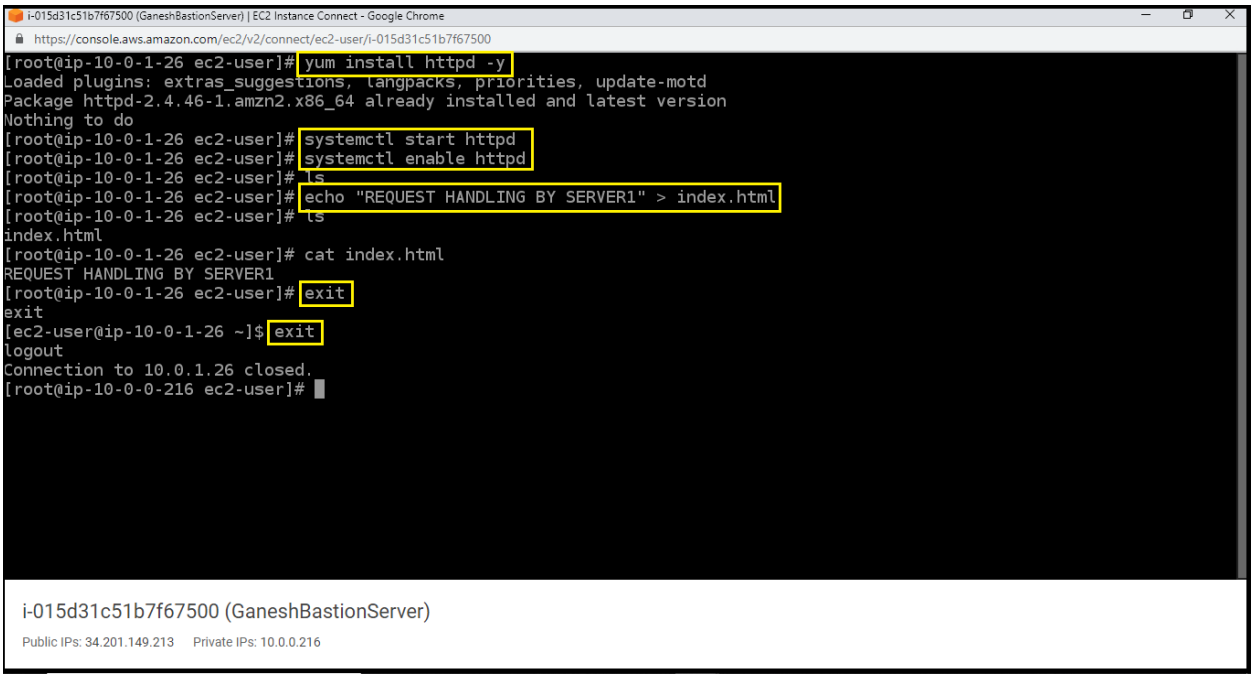
1. SSH into the Bastion server using the Bastion PEM key: **my-web-serverkey.pem**
2. Open the my-**web-serverkey.pem** file on your local system and then copy the text content.
3. Navigate to the Bastion server and create a file named **web-serverkey.pem** using below command:
    - **vi web-serverkey.pem**
5. Press "i" which enables the Insert mode in the vi editor and then paste the content and save it by pressing **Esc** key from keyboard and then Insert ":" by pre followed by "**:wq**" and then Press Enter to save your private key.
6. Change the permission of the file using below command:
    - **chmod 400 web-serverkey.pem**
7. Now you can log into the web servers using the private key copied to the bastion server with the help of below commands.



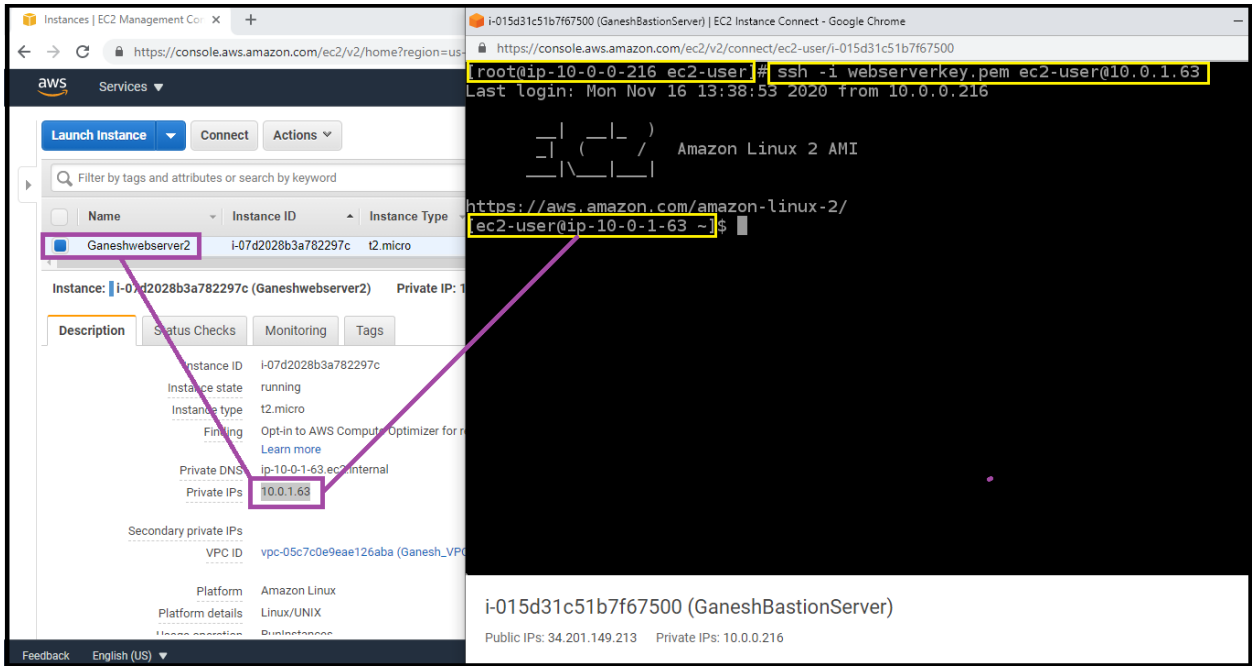
- **Note:** You don't have a public IPs for the web servers since we have them in a private subnet.
- Insert the following cmd: - **ssh -i web-serverkey.pem ec2-user@10.0.1.26**

8. Now install the apache service using the below commands and create a test **index.html** file, which will be used for a health check.

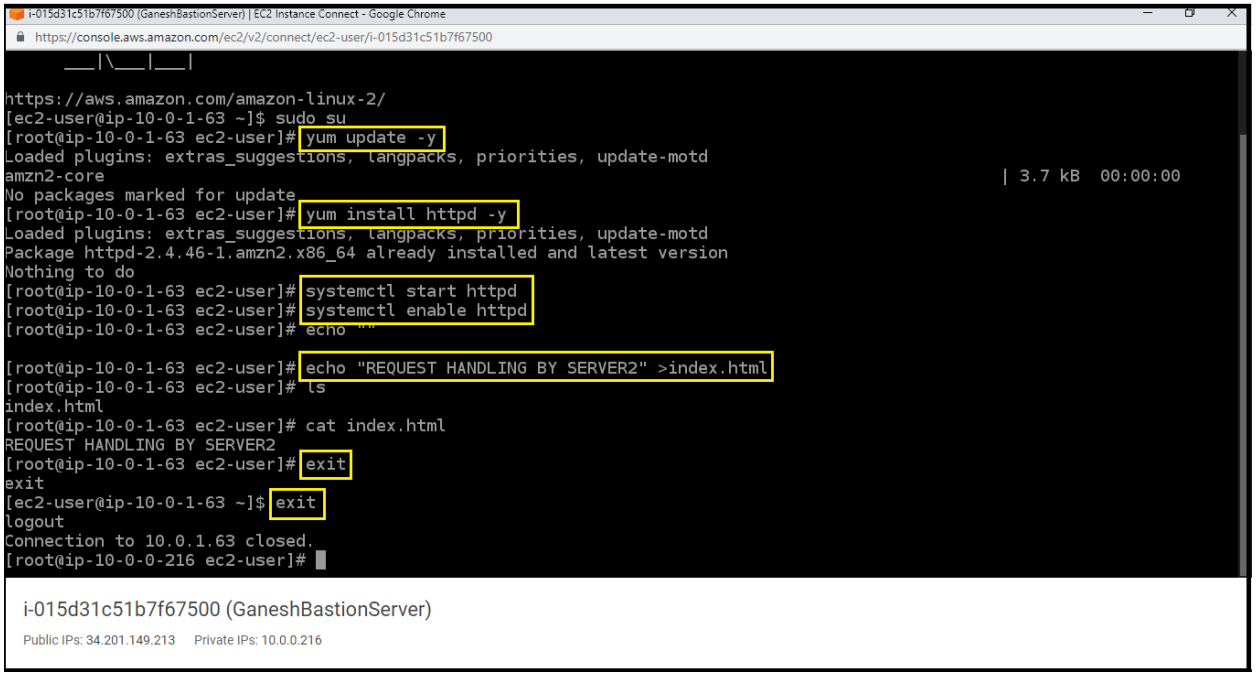
- **Installing Apache:**
    - sudo su
    - yum update –y



- yum install httpd -y
- systemctl start httpd
- systemctl enable httpd
- echo “REQUEST HANDLING BY SERVER 1” > index.html
- **Exit** – Exiting from “**GaneshWebserver1”**
- **Exit** - Come out of 2nd instance



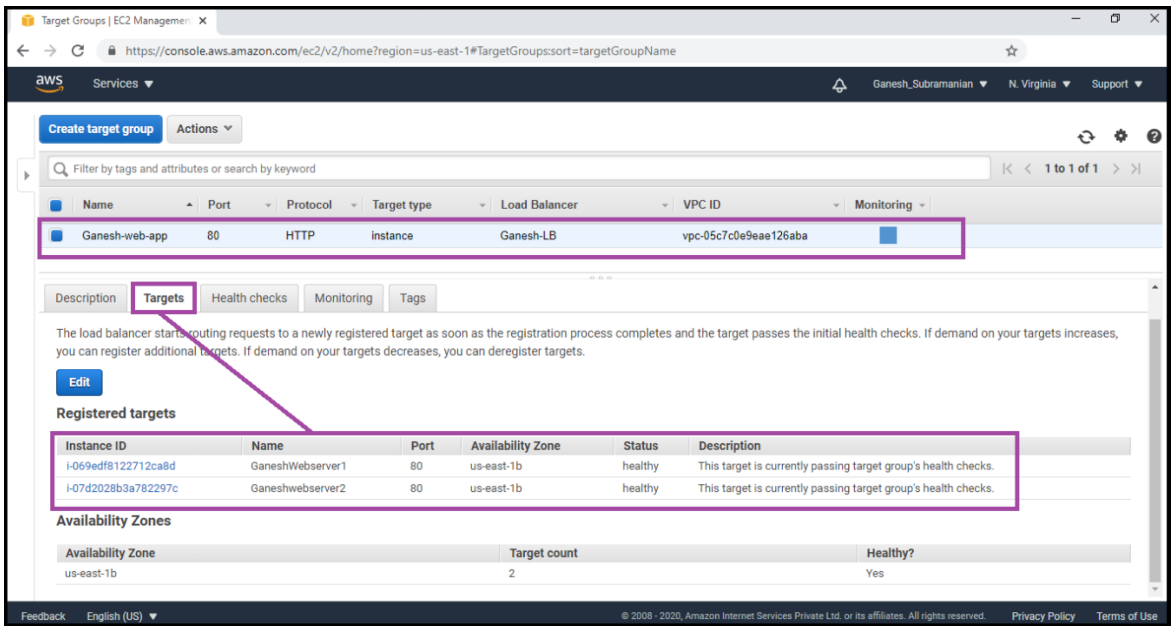
9. Repeat steps 7 & 8 for web server 2 with its respective private IP (**10.0.1.63**), making sure to change the content of index.html to "**REQUEST HANDLING BY SERVER 2**"



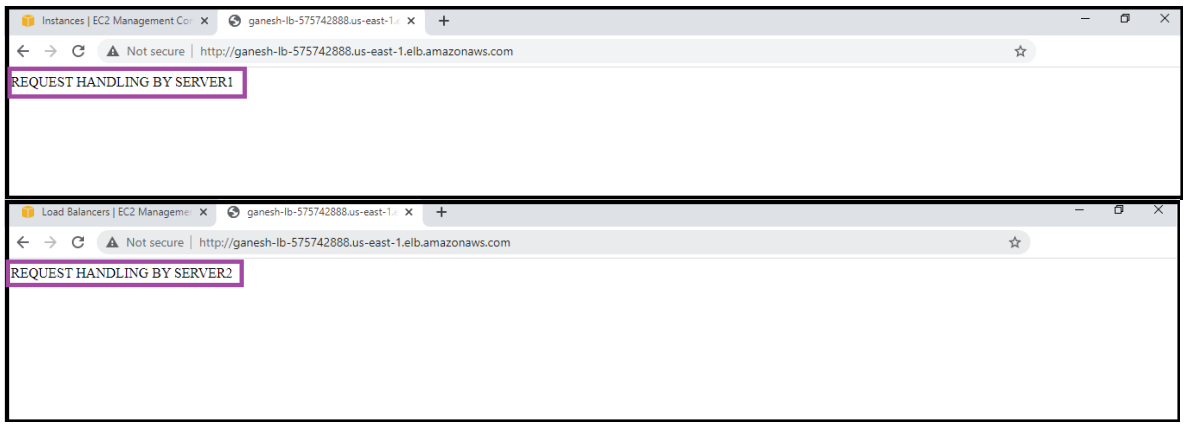
10. To come out of 1st instance, type **exit** command for coming out of root user, and **exit** command again for coming out of the instance.

Checking the health of the load balancer

1. Navigate to Target group
2. Select the target group "**Ganesh-web-app**" and then click on **Target** to see the **Status** of the attached targets.
3. It should show **Healthy** for the Load Balancer to work properly.



4. Now navigate to **Load Balancer** and select "**Ganesh-LB**" and then Click on DNS Name, copy the same and paste it into the browser.
   - DNS URL: Ganesh-LB-575742888.us-east-1.elb.amazonaws.com
5. Refresh the browser a couple of times to see the requests being served from both servers. Seeing output similar to **REQUEST HANDLING BY SERVER 1 & REQUEST HANDLING BY SERVER 2** implies that load is shared between the two web servers via Application Load Balancer.

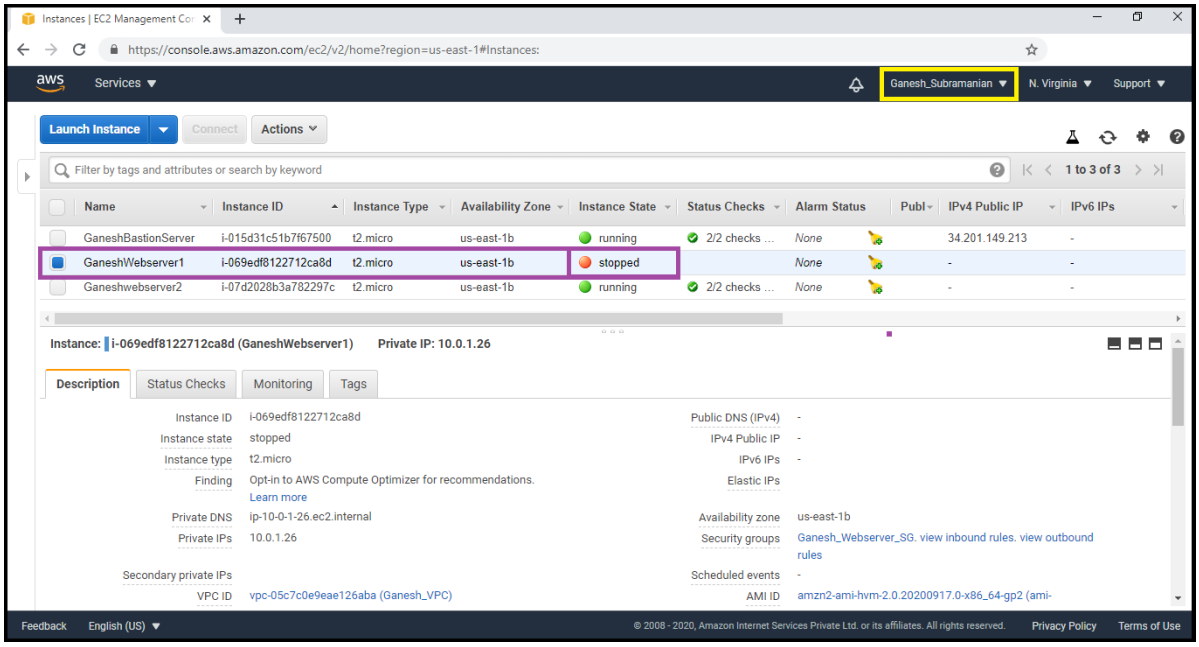


6. Now we have successfully created a bastion server, two web servers and an Application Load balancer, registered the targets to the load balancer and tested the working of Load Balancer.

Test case for High Availability

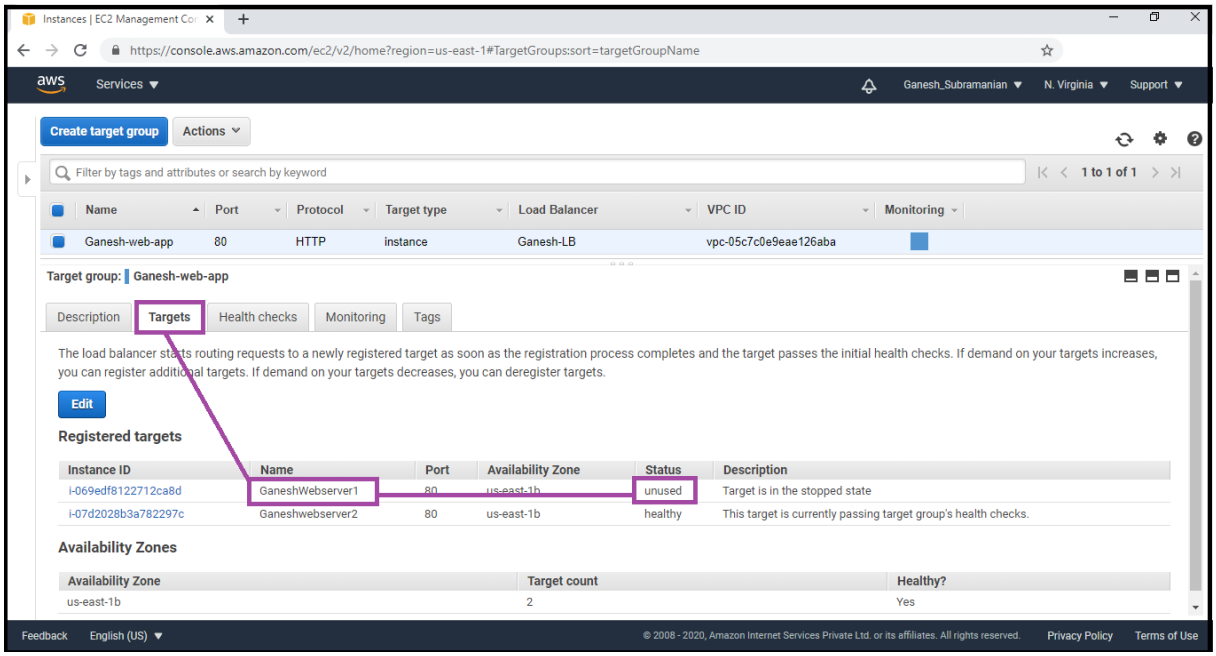
To check for high availability, we will make one of the instances unhealthy and test whether we get response from the other server.

1. Navigate to the EC2 dashboard and select "**GaneshWebserver1**".



Select Action / Instance State / **Stop**.

3. Navigate to **Target Groups** and click on **Targets**. Here you will find the status of "**GaneshWebserver1**" (which should be unhealthy / Unused because it is unused).



5. Navigate to **Load balancers-->Description-->DNS name**. Copy the DNS name and paste it into your browser. You should see the response "REQUEST HANDLING BY SERVER 2" from **GaneshWebserver2**.

6. If you refresh a few times, you will continue to see the response only from Web-server-2

7. 

8. Repeat step 3 by stopping "**GaneshWebserver2**" and starting "**GaneshWebserver1**" back up. This time you should see the response "REQUEST HANDLING BY SERVER1" from "**GaneshWebserver1**".

Completion and Conclusion

1. We have launched a Bastion server and two web-servers. We were able to SSH into the servers via Bastion Server successfully.

   ➔ Bastion Server "**GaneshBastionServer**" & 2 Web-Servers "**GaneshWebserver1**", "**GaneshWebserver2**"

2. We launched an Application Load Balancer and associated our web servers with the load balancer.

   ➔ Load Balancer "**Ganesh-LB**"

3. We tested the load sharing between web servers.

   ➔ Load Sharing Between Web Servers Successful

4. We successfully tested the high availability of the web application by making one of the web servers unhealthy.

   ➔ High Availability Test of the Web Application was Successful