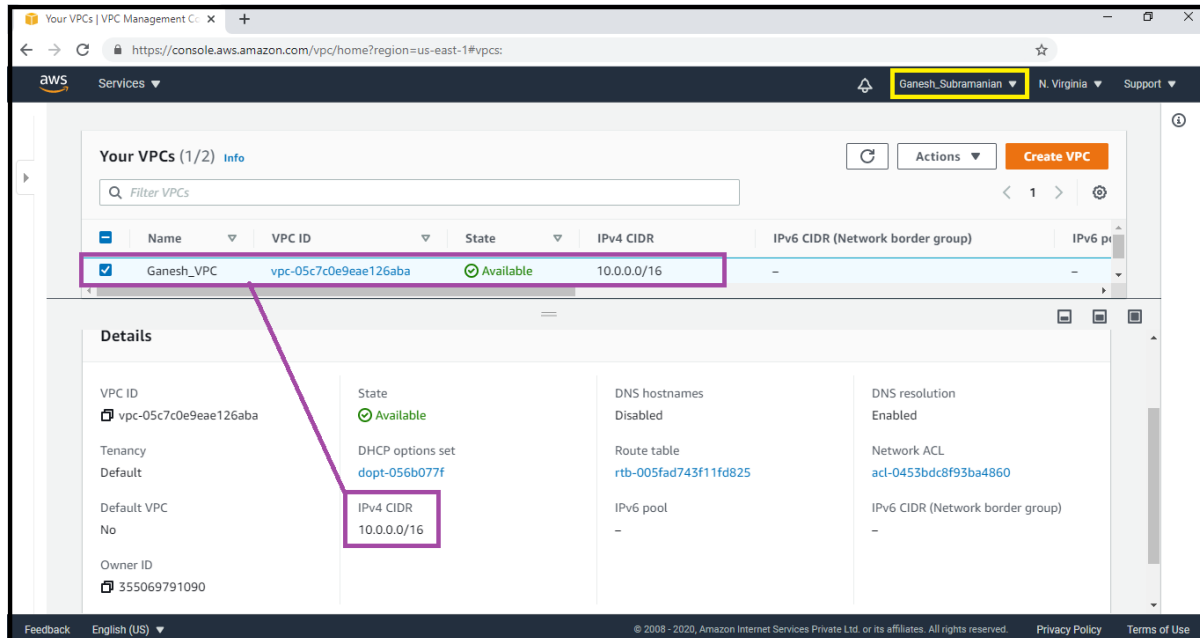


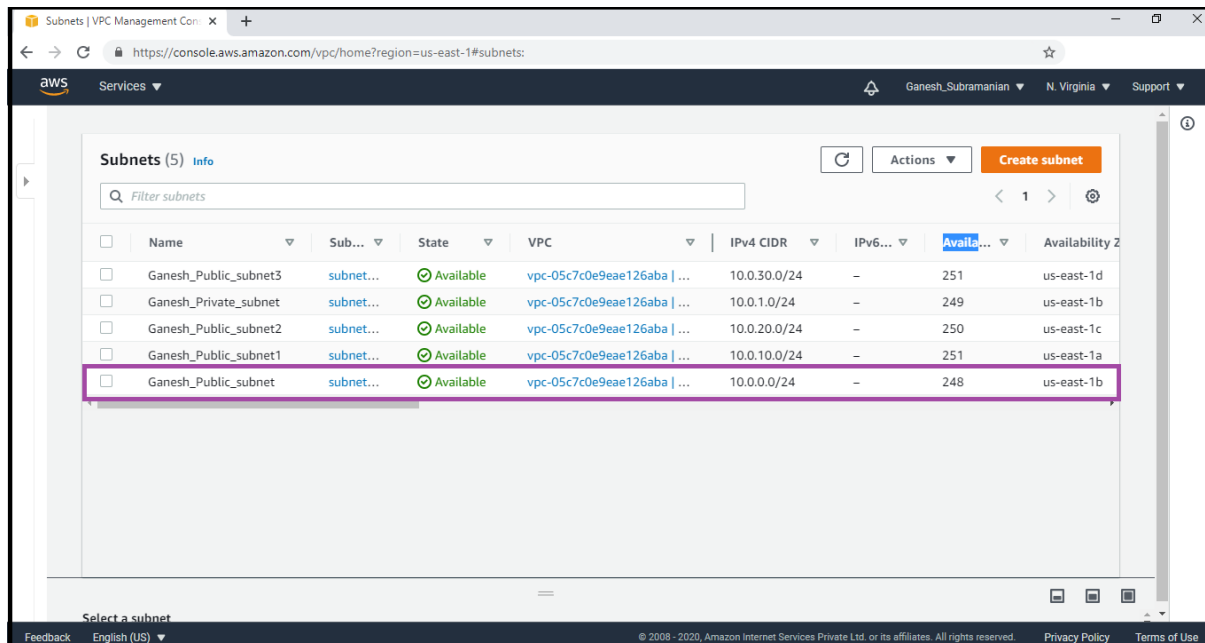
# Project 1

➔ Create VPC: -



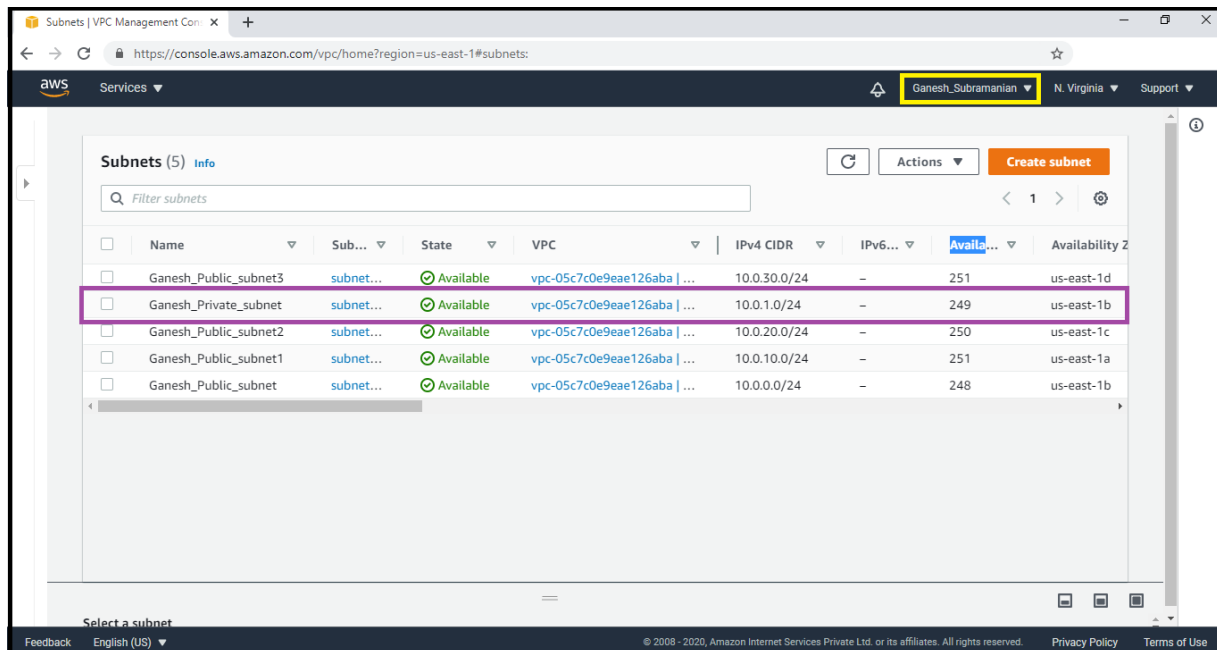
VPC Name: - "Ganesh\_VPC"  
IPv4 CIDR Block: - 10.0.0.0/16

➔ Create Subnet (Public): -



Public Subnet Name: - "Ganesh\_Public\_subnet"  
VPC: - "Ganesh\_VPC"  
IPv4 CIDR Block: - "10.0.0.0/24"

## ➔ Create Subnet (Private): -

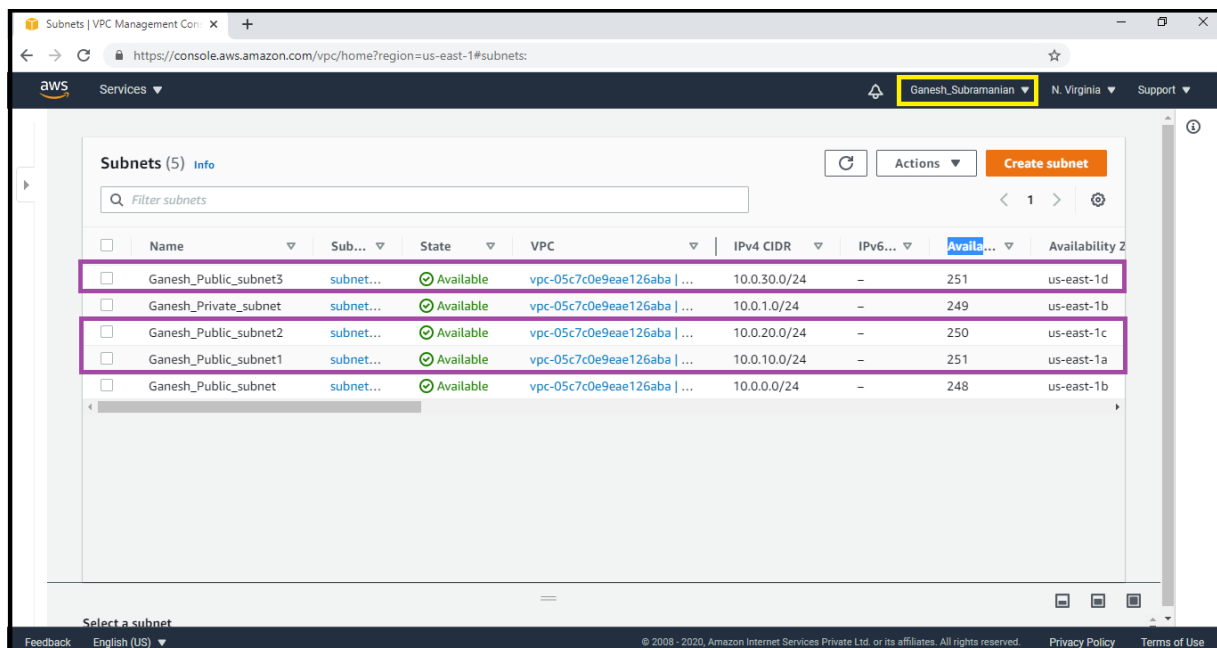


Private Subnet Name: - **"Ganesh\_Private\_subnet"**

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: -

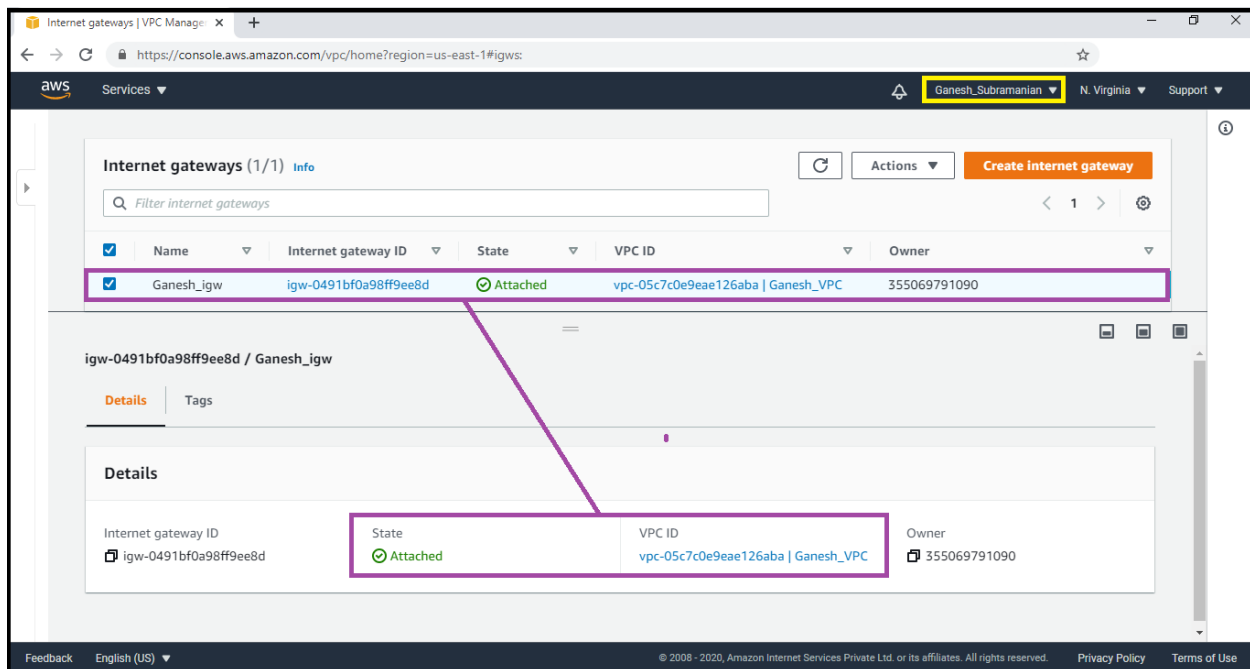


Public Subnet 1 (us-east-1a): - **"Ganesh\_Public\_subnet1"** IPv4 CIDR - **10.0.10.0/24**

Public Subnet 2 (us-east-1c): - **"Ganesh\_Public\_subnet2"** IPv4 CIDR - **10.0.20.0/24**

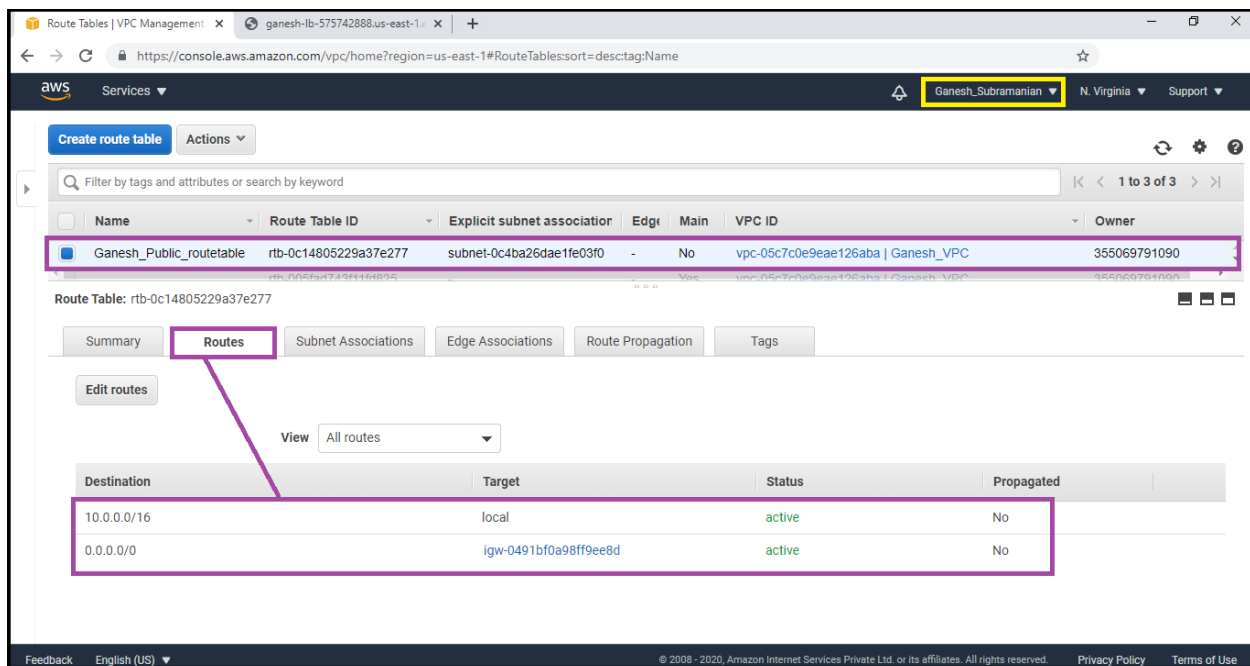
Public Subnet 3 (us-east-1d): - **"Ganesh\_Public\_subnet3"** IPv4 CIDR - **10.0.30.0/24**

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

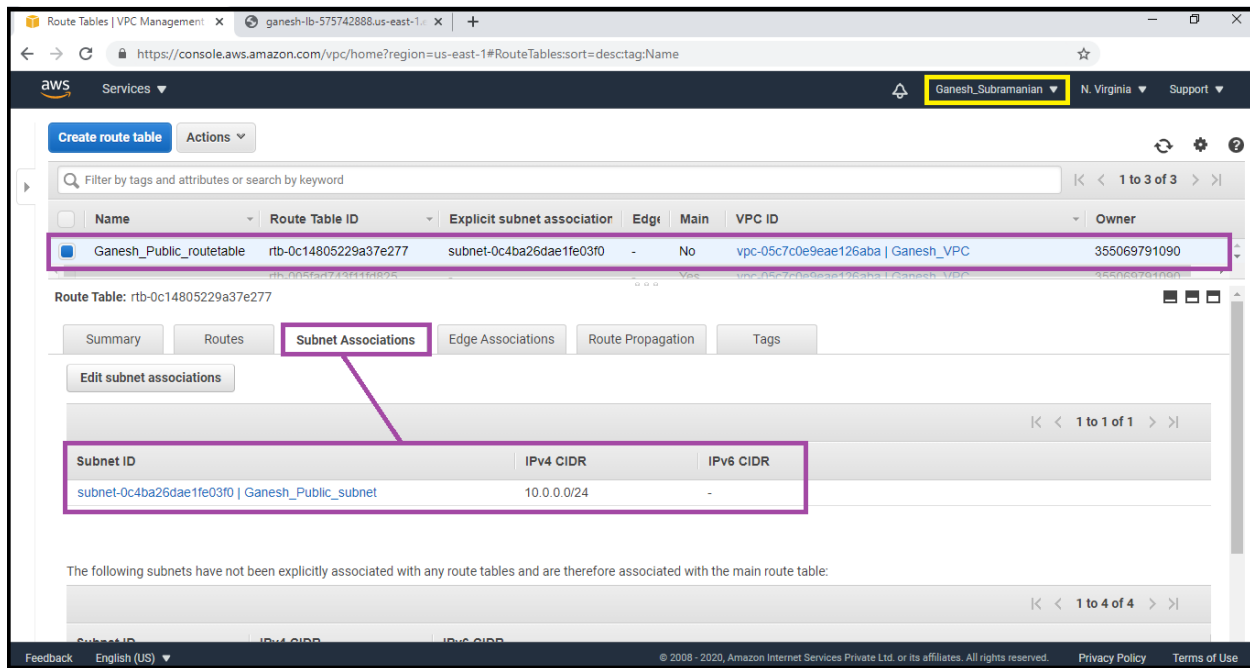


IGW Name: – **“Ganesh\_igw”**  
Attached to VPC: - **“Ganesh\_VPC”**

➔ **Create Route Table: -**

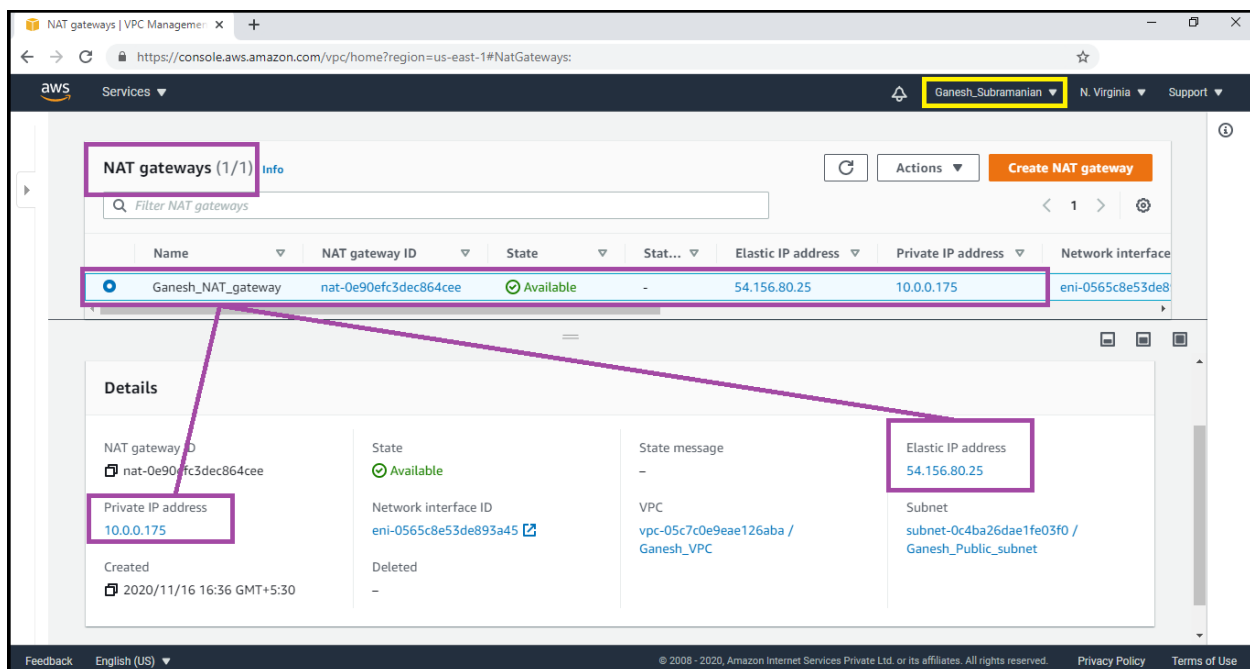


Route Table Name: - **“Ganesh\_Public\_routetable”**  
Routes / Edit Routes / Add Routes: - Destination: - **“0.0.0.0/0”** & Target: - **“igw-0491bf0a98ff9ee8d”**



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: -

Route Table: rtb-005fad743f11fd825

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
0.0.0.0/0	nat-0e90efc3dec864cee	active	No

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: -

Instance: GaneshBastionServer

Instance ID: i-015d31c51b7f67500

Instance Type: t2.micro

Availability Zone: us-east-1b

Instance State: running

Status Checks: 2/2 checks ...

Alarm Status: None

IPv4 Public IP: 34.201.149.213

Private DNS: ip-10-0-0-216.ec2.internal

Private IPs: 10.0.0.216

Public DNS (IPv4): -

IPv4 Public IP: 34.201.149.213

IPv6 IPs: -

Elastic IPs: -

Availability zone: us-east-1b

Security groups: Ganesh\_BastionSG, view inbound rules, view outbound rules

Scheduled events: No scheduled events

AMI ID: ami-zn2-ami-hvm-2.0.20200917.0-x86\_64-gp2 (ami-0947d2ba12ee1ff75)

Subnet ID: subnet-0c4ba26dae1fe03f0 (Ganesh\_Public\_subnet)

Network interfaces: eth0

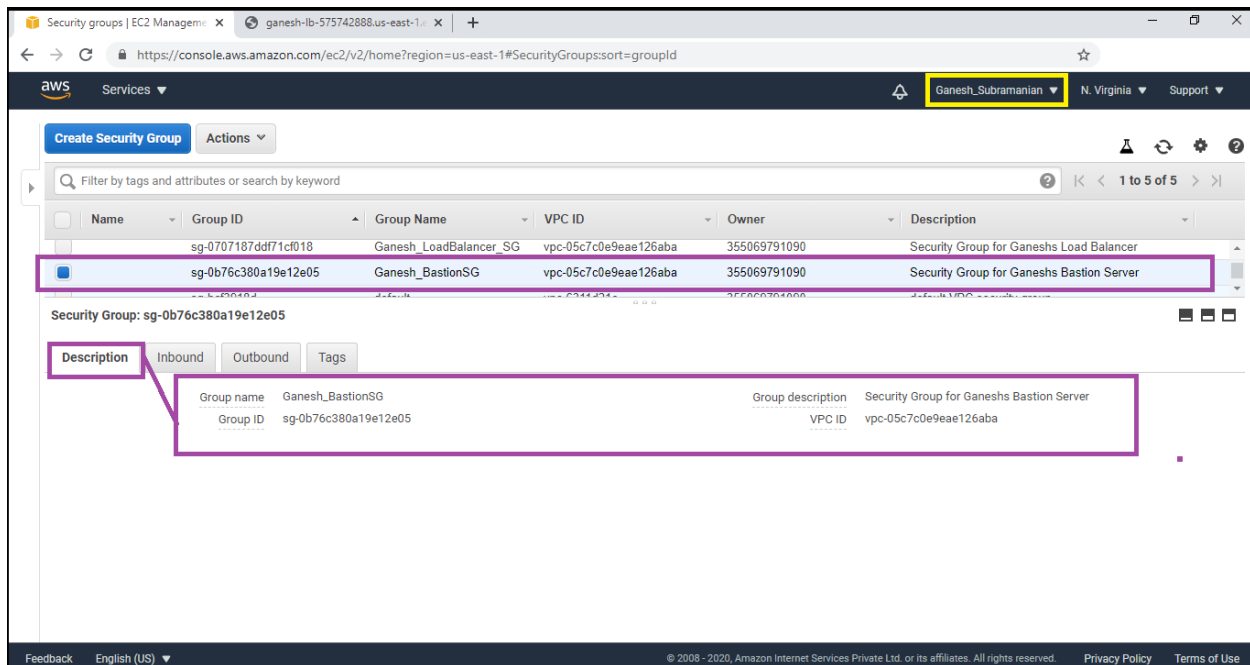
IAM role: -

Bastion Server Name: - **“GaneshBastionServer”**

Private IP: - **10.0.0.216**

Public IP: - **34.201.149.213**

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

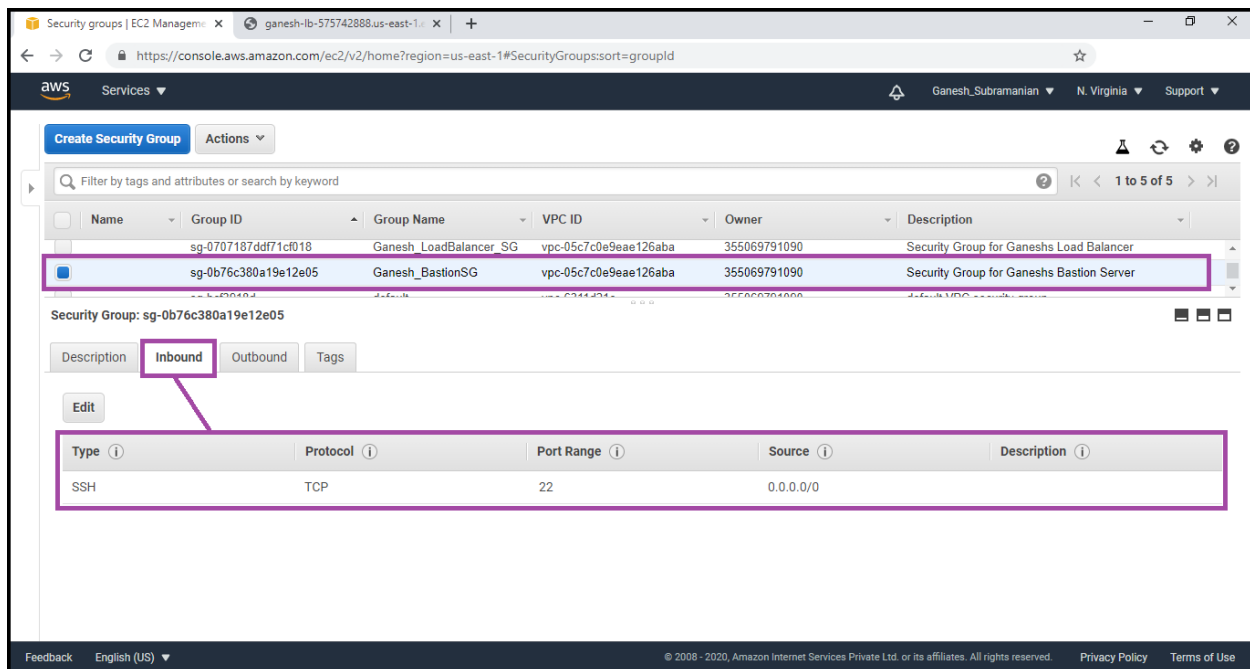


The screenshot shows the AWS Management Console for the 'Security groups' section. The 'Create Security Group' button is visible. A table lists existing security groups, with 'Ganesh\_BastionSG' highlighted. Below the table, the 'Description' tab is selected for the group 'sg-0b76c380a19e12e05'. The details shown are:

Group name	Group description
Ganesh_BastionSG	Security Group for Ganeshs Bastion Server

Additional details shown below the table:

Group ID	VPC ID
sg-0b76c380a19e12e05	vpc-05c7c0e9eae126aba

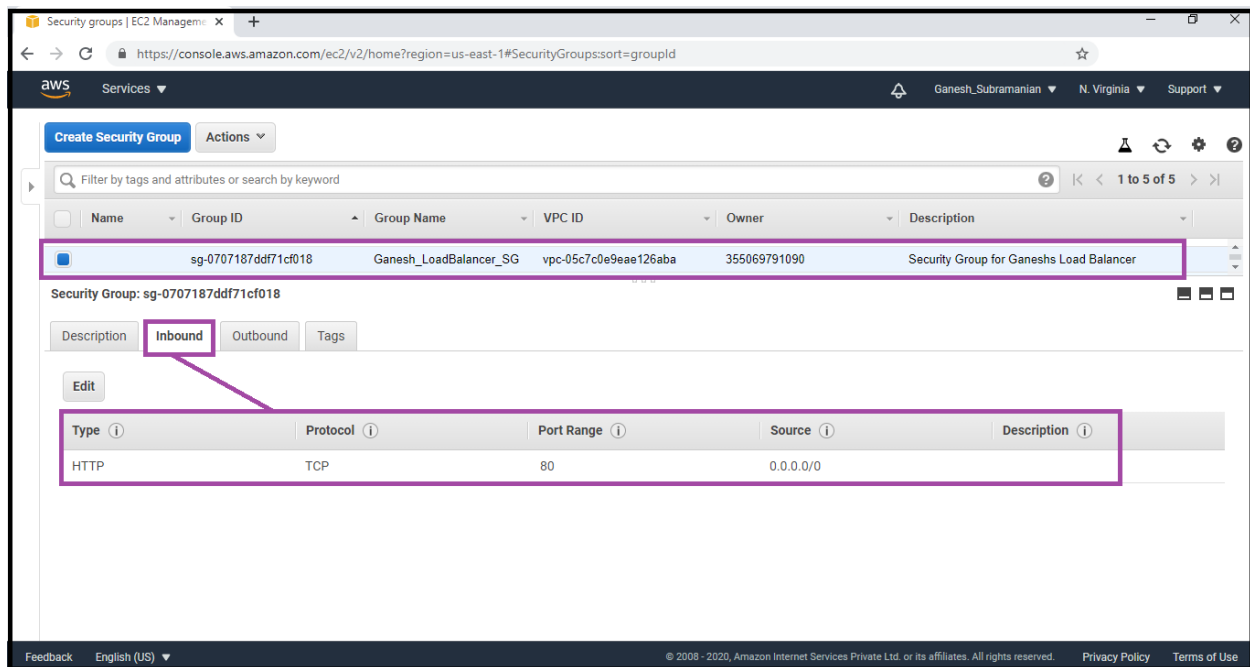
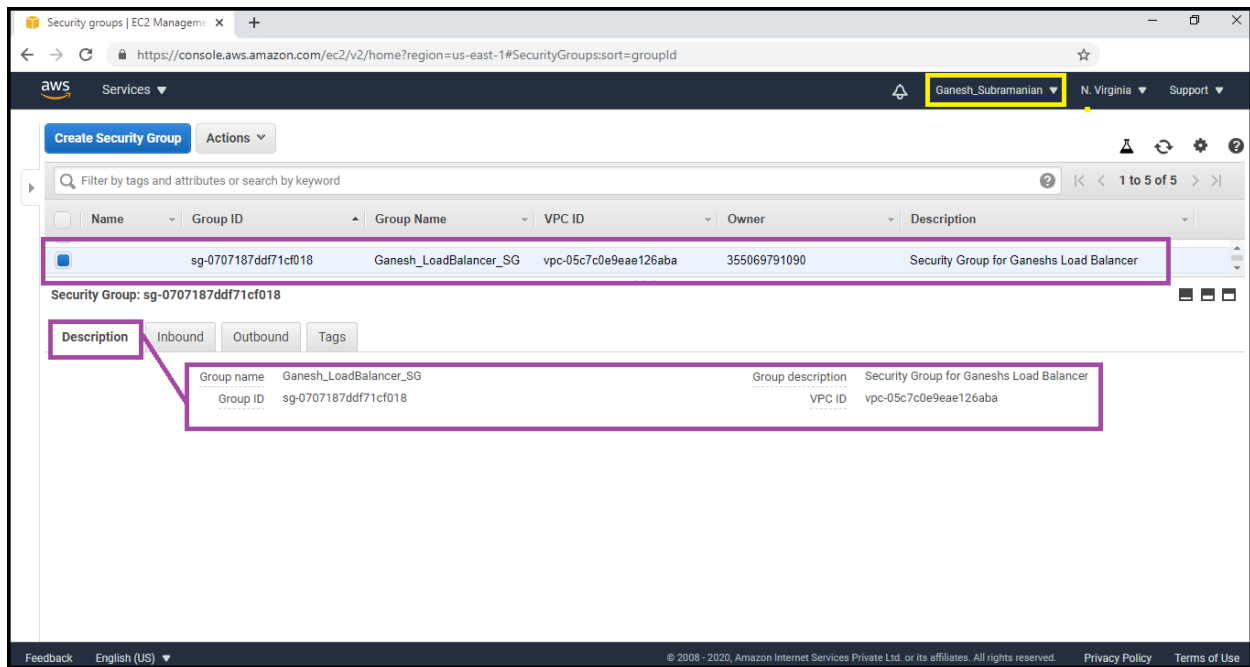


The screenshot shows the AWS Management Console for the 'Security groups' section. The 'Create Security Group' button is visible. A table lists existing security groups, with 'Ganesh\_BastionSG' highlighted. Below the table, the 'Inbound' tab is selected for the group 'sg-0b76c380a19e12e05'. The details shown are:

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	0.0.0.0/0	

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: -

The screenshot shows the AWS Management Console interface for the EC2 service. The top navigation bar includes the AWS logo, 'Services', and user information. The main content area displays a list of instances. The instance 'GaneshWebserver1' is highlighted with a purple box. Below the list, the details for this instance are shown, including its ID, type, state, and various network-related information. A purple box highlights the 'Private IP' field, which shows the value '10.0.1.26'.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Publ	IPv4 Public IP	IPv6 IPs
GaneshBastionServer	i-015d31c51b767500	t2.micro	us-east-1b	running	2/2 checks ...	None		34.201.149.213	-
GaneshWebserver1	i-069edf8122712ca8d	t2.micro	us-east-1b	running	2/2 checks ...	None		-	-

Instance: **i-069edf8122712ca8d** (GaneshWebserver1) Private IP: 10.0.1.26

**Description** Status Checks Monitoring Tags

Instance ID: i-069edf8122712ca8d  
Instance state: running  
Instance type: t2.micro  
Finding: Opt-in to AWS Compute Optimizer for recommendations.  
Private DNS: ip-10-0-1-26.ec2.internal  
Private IPs: **10.0.1.26**  
Secondary private IPs: -  
VPC ID: vpc-05c7c0e9eae126aba (Ganesh\_VPC)  
Platform: Amazon Linux

Public DNS (IPv4): -  
IPv4 Public IP: -  
IPv6 IPs: -  
Elastic IPs: -  
Availability zone: us-east-1b  
Security groups: Ganesh\_Webserver\_SG. view inbound rules. view outbound rules  
Scheduled events: No scheduled events  
AMI ID: amzn2-ami-hvm-2.0.20200917.0-x86\_64-gp2 (ami-0947d2ba12ee1ff75)  
Subnet ID: subnet-0c7211948a3aeec99 (Ganesh\_Private\_subnet)

Webserver1 Name: - **"GaneshWebserver1"**

Private IP: - **10.0.1.26**

Public IP: - -----

The screenshot shows the AWS Management Console interface for the EC2 service. The top navigation bar includes the AWS logo, 'Services', and user information. The main content area displays a list of instances. The instance 'Ganeshwebserver2' is highlighted with a purple box. Below the list, the details for this instance are shown, including its ID, type, state, and various network-related information. A purple box highlights the 'Private IP' field, which shows the value '10.0.1.63'.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Publ	IPv4 Public IP	IPv6 IPs
Ganeshwebserver2	i-07d2028b3a782297c	t2.micro	us-east-1b	running	2/2 checks ...	None		-	-

Instance: **i-07d2028b3a782297c** (Ganeshwebserver2) Private IP: 10.0.1.63

**Description** Status Checks Monitoring Tags

Instance ID: i-07d2028b3a782297c  
Instance state: running  
Instance type: t2.micro  
Finding: Opt-in to AWS Compute Optimizer for recommendations.  
Private DNS: ip-10-0-1-63.ec2.internal  
Private IPs: **10.0.1.63**  
Secondary private IPs: -  
VPC ID: vpc-05c7c0e9eae126aba (Ganesh\_VPC)  
Platform: Amazon Linux  
Platform details: Linux/UNIX

Public DNS (IPv4): -  
IPv4 Public IP: -  
IPv6 IPs: -  
Elastic IPs: -  
Availability zone: us-east-1b  
Security groups: Ganesh\_Webserver\_SG. view inbound rules. view outbound rules  
Scheduled events: No scheduled events  
AMI ID: amzn2-ami-hvm-2.0.20200917.0-x86\_64-gp2 (ami-0947d2ba12ee1ff75)  
Subnet ID: subnet-0c7211948a3aeec99 (Ganesh\_Private\_subnet)  
Network interfaces: eth0

Webserver2 Name: - **"Ganeshwebserver2"**

Private IP: - **10.0.1.63**

Public IP: - -----



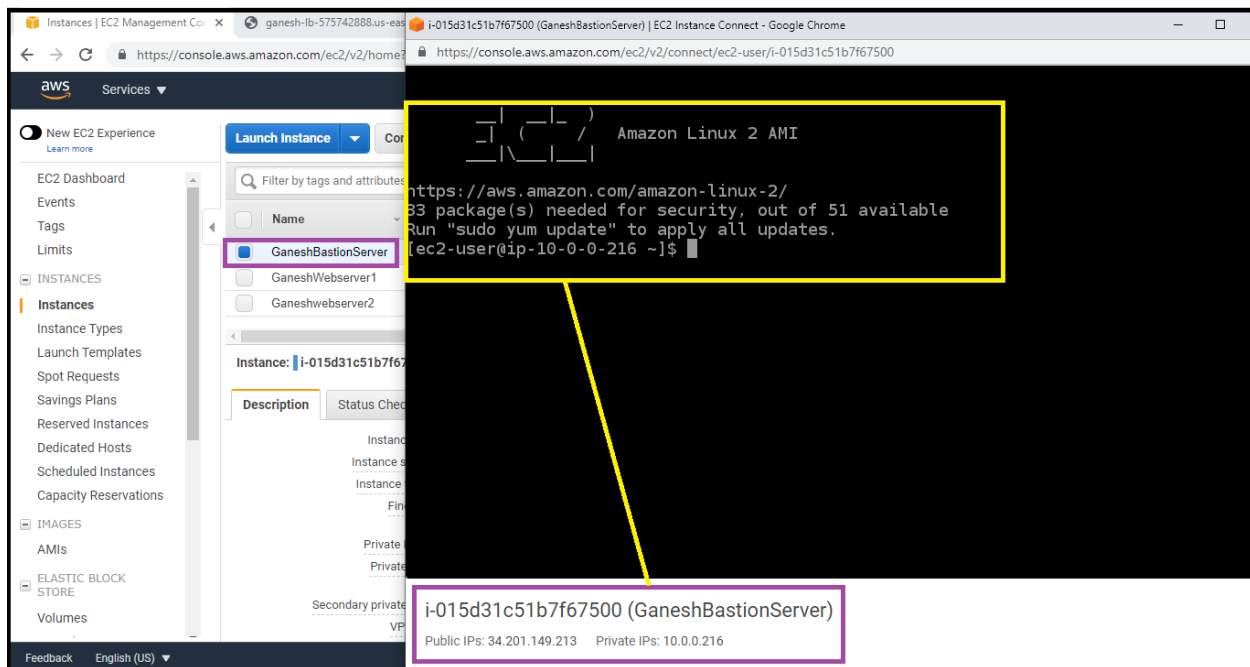
## ➔ Create Load Balancer: -

The screenshot shows the AWS Management Console 'Load Balancers' page. At the top, there's a 'Create Load Balancer' button. Below it is a table of load balancers. The first entry is 'Ganesh-LB' with a DNS name of 'Ganesh-LB-575742888.us-east-1.elb.amazonaws.com', state 'active', VPC ID 'vpc-05c7c0e9eae126aba', and availability zones 'us-east-1b, us-east-1d, us-east-1a, us-east-1c'. Below the table, the 'Basic Configuration' section for 'Ganesh-LB' is shown, including its ARN, DNS name, state, type, scheme, IP address type, VPC, and availability zones. Red boxes highlight the load balancer entry in the table and the 'Basic Configuration' section.

Load Balancer Name: – **“Ganesh-LB”**

## ➔ Launch bastion Server: -

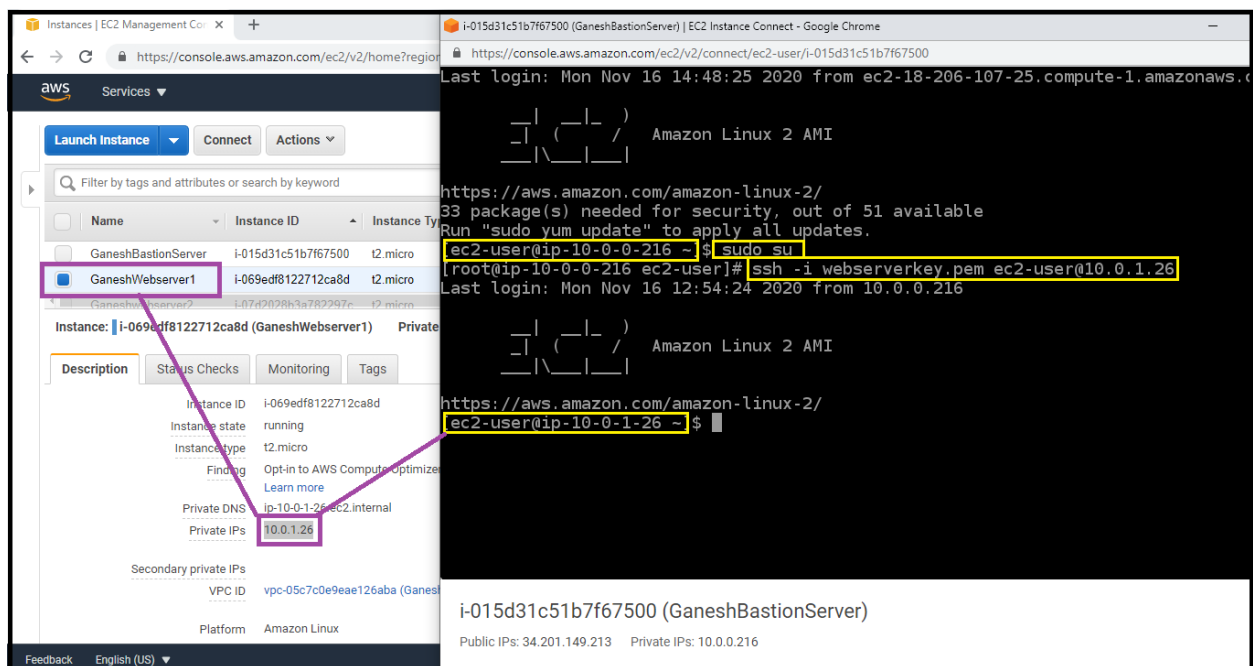
The screenshot shows the AWS Management Console 'Instances' page. A modal window titled 'Connect to your instance' is open. It shows three connection methods: 'A standalone SSH client', 'Session Manager', and 'EC2 Instance Connect (browser-based SSH connection)'. The 'EC2 Instance Connect' option is selected. Below the methods, there's a 'User name' field with the value 'ec2-user'. At the bottom right of the modal, there are 'Close' and 'Connect' buttons. Red boxes highlight the 'GaneshBastionServer' instance in the background, the 'EC2 Instance Connect' option, and the 'Connect' button.



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**

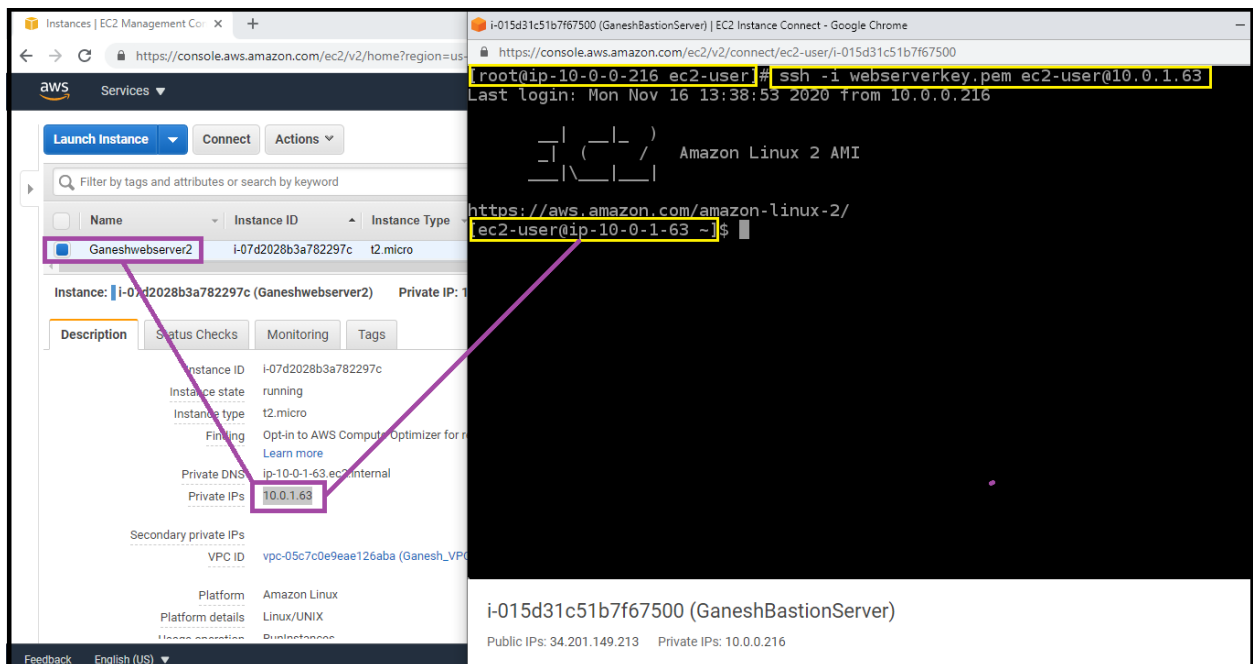
The screenshot shows a terminal window titled "i-015d31c51b7f67500 (GaneshBastionServer) | EC2 Instance Connect - Google Chrome". The terminal output is as follows:

```
[root@ip-10-0-0-1-26 ec2-user]# yum install httpd -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Package httpd-2.4.46-1.amzn2.x86_64 already installed and latest version
Nothing to do
[root@ip-10-0-0-1-26 ec2-user]# systemctl start httpd
[root@ip-10-0-0-1-26 ec2-user]# systemctl enable httpd
[root@ip-10-0-0-1-26 ec2-user]# ls
[root@ip-10-0-0-1-26 ec2-user]# echo "REQUEST HANDLING BY SERVER1" > index.html
[root@ip-10-0-0-1-26 ec2-user]# ls
index.html
[root@ip-10-0-0-1-26 ec2-user]# cat index.html
REQUEST HANDLING BY SERVER1
[root@ip-10-0-0-1-26 ec2-user]# exit
exit
[ec2-user@ip-10-0-0-1-26 ~]$ exit
logout
Connection to 10.0.1.26 closed.
[root@ip-10-0-0-216 ec2-user]#
```

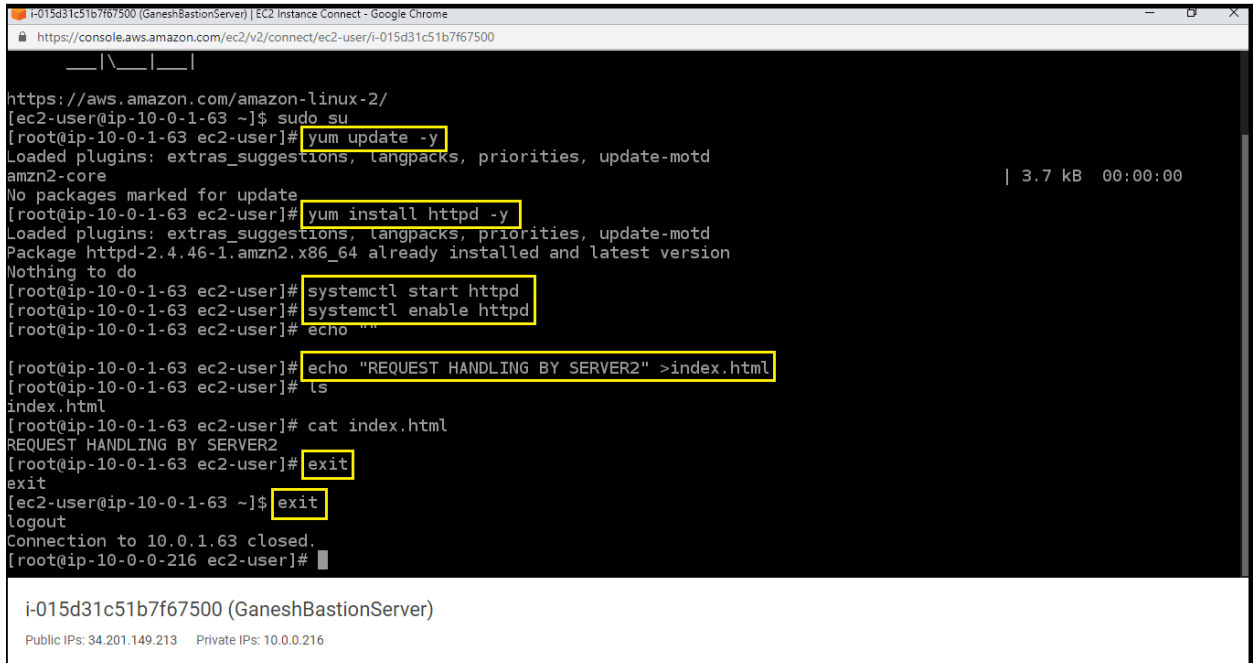
Below the terminal output, the instance details are shown:

i-015d31c51b7f67500 (GaneshBastionServer)  
Public IPs: 34.201.149.213 Private IPs: 10.0.0.216

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



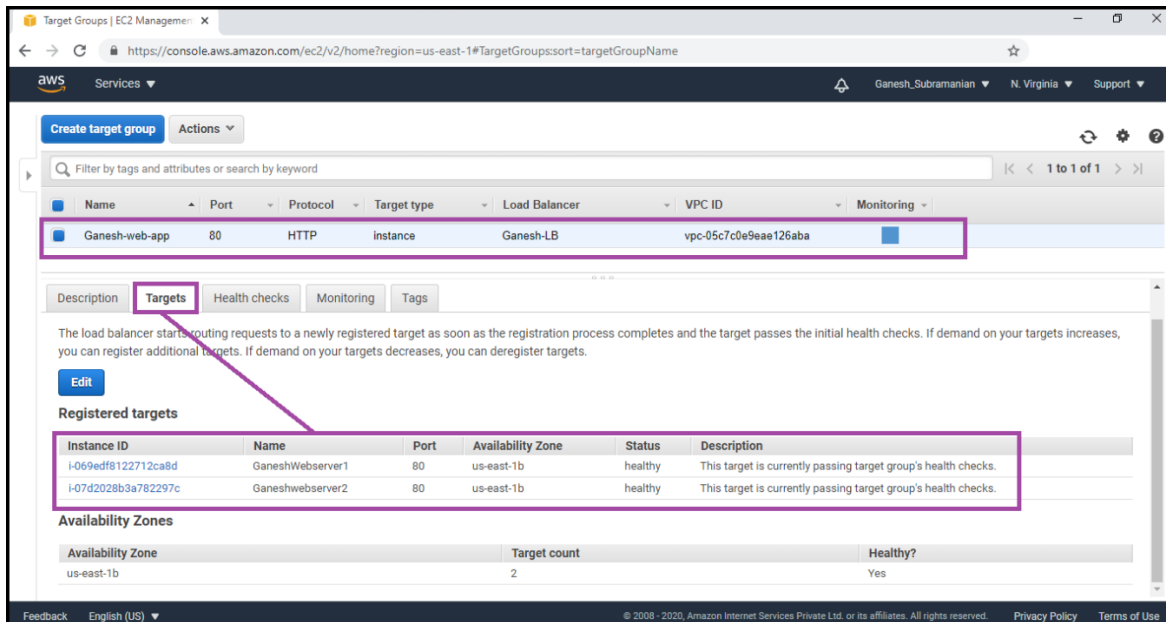
9. Repeat steps 7 & 8 for web server 2 with its respective private IP (10.0.1.63), making sure to change the content of index.html to "REQUEST HANDLING BY SERVER 2"



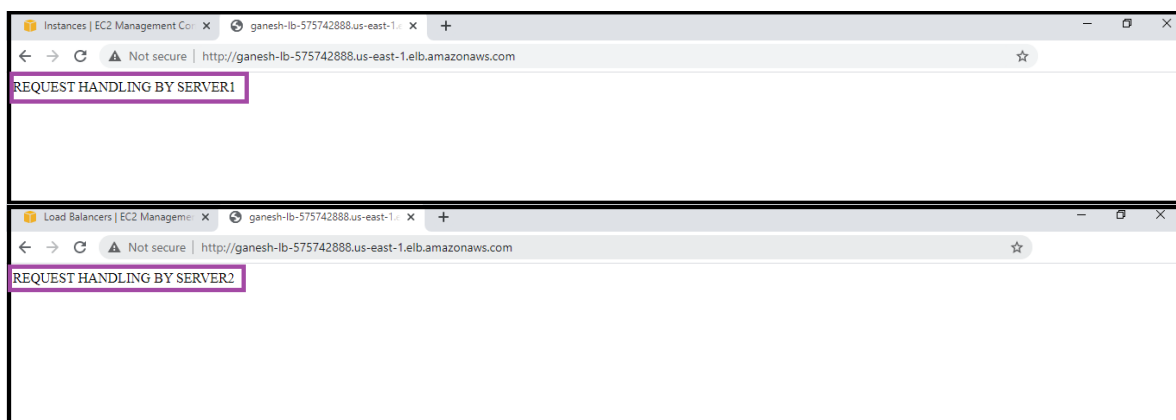
10. To come out of 1st instance, type **exit** command for coming out of root user, and **exit** command again for coming out of the instance.

# 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.
  - o 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.

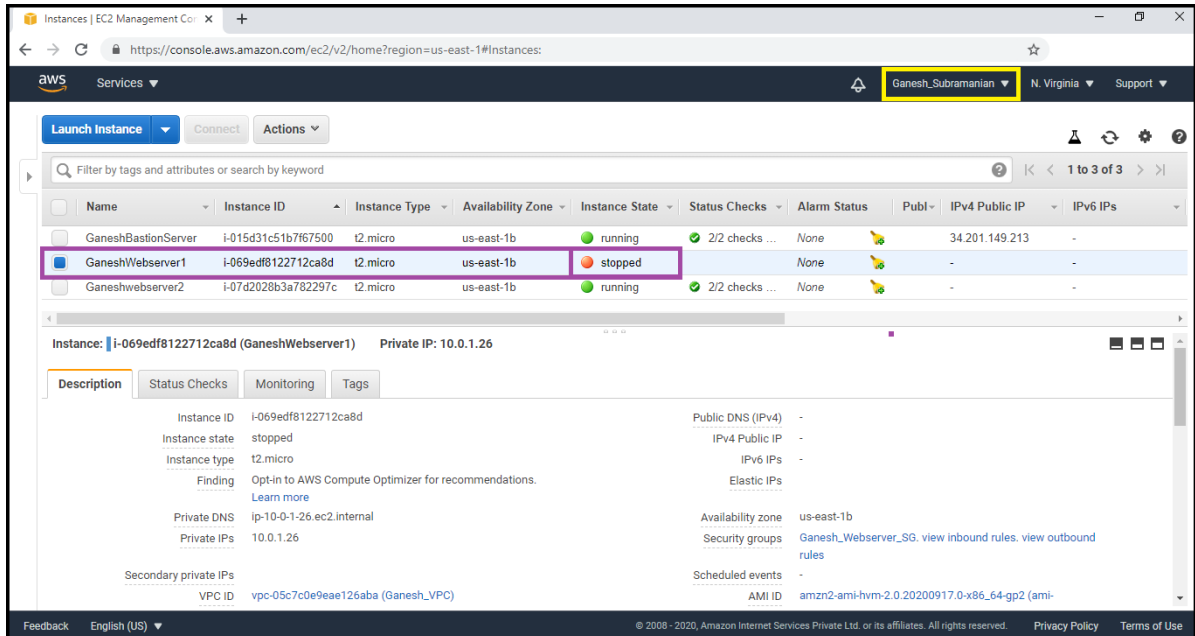


6. Now we have successfully created a bastion server, two web servers and an Application Load balancer, registered the targets to the load balancer and tested the working of Load Balancer.

# Test case for High Availability

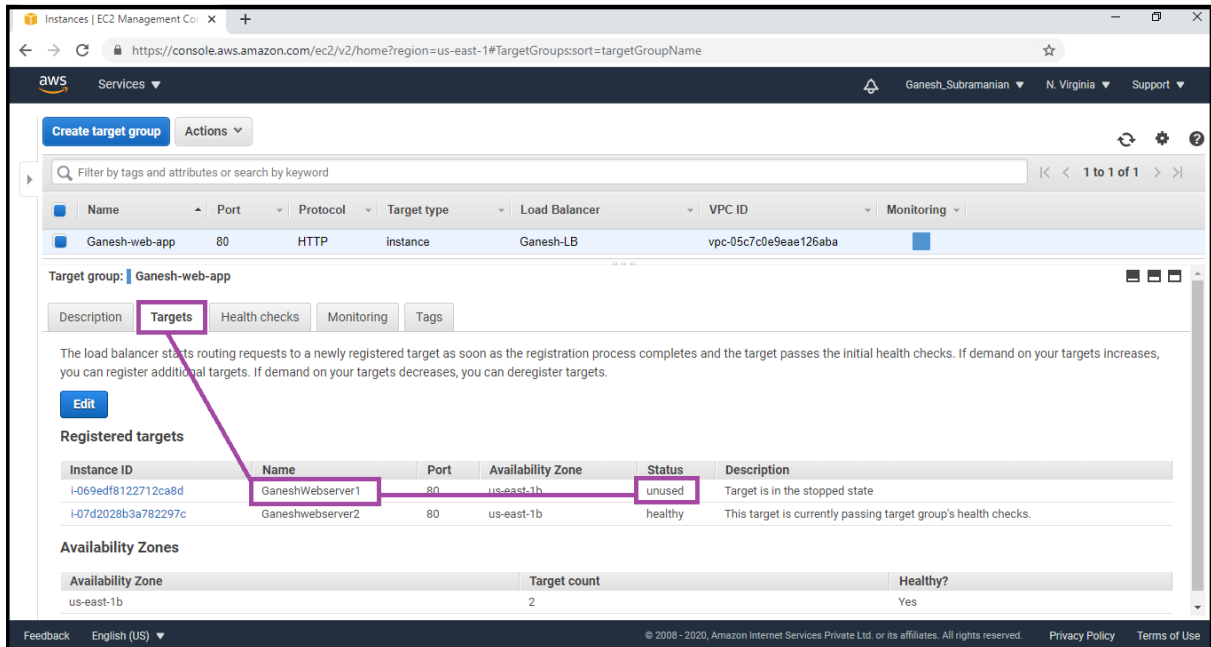
To check for high availability, we will make one of the instances unhealthy and test whether we get response from the other server.

1. Navigate to the EC2 dashboard and select “**GaneshWebserver1**”.

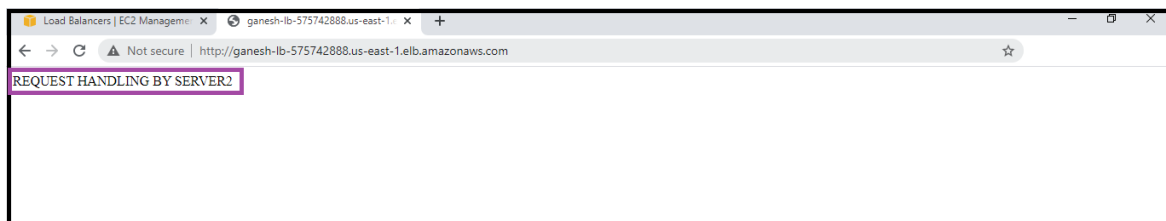


Select Action / Instance State / **Stop**.

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).



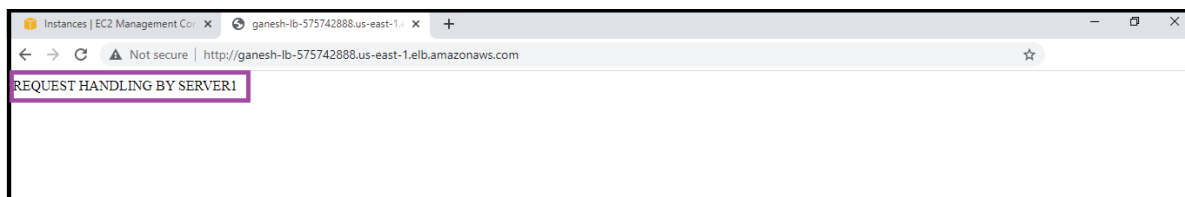
5. Navigate to **Load balancers-->Description-->DNS name**. Copy the DNS name and paste it into your browser. You should see the response "REQUEST HANDLING BY SERVER 2" from **GaneshWebserver2**.



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