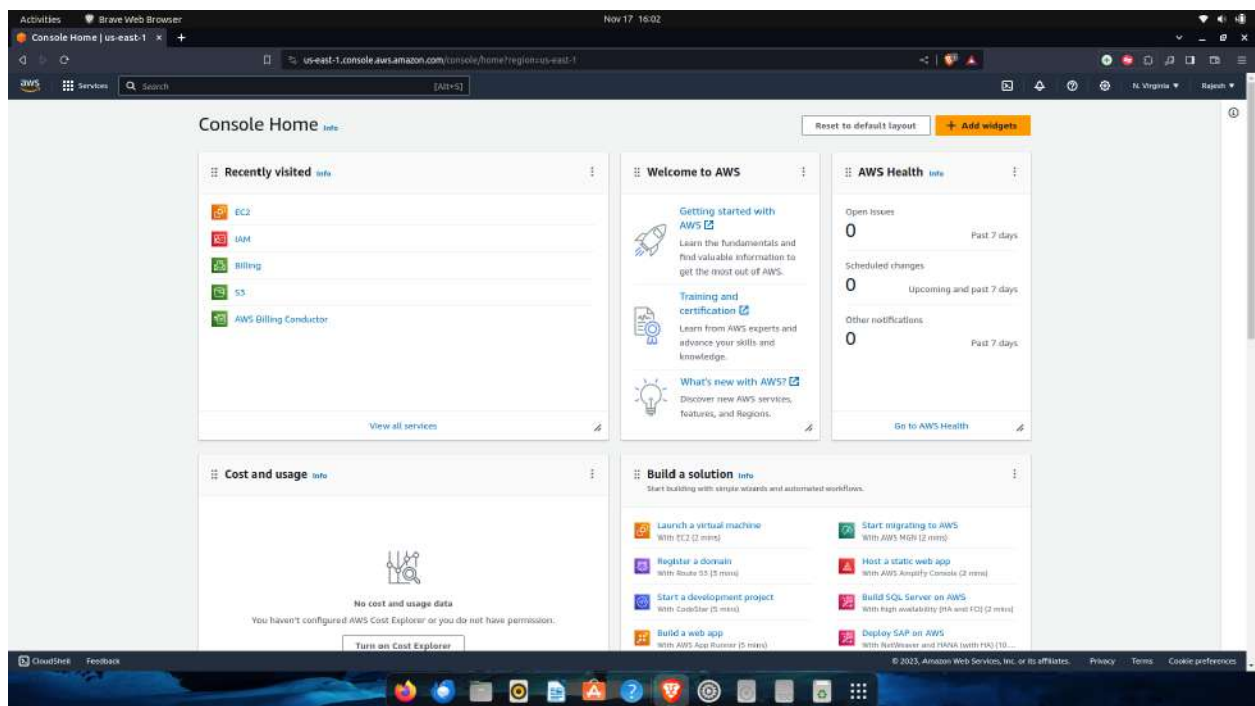


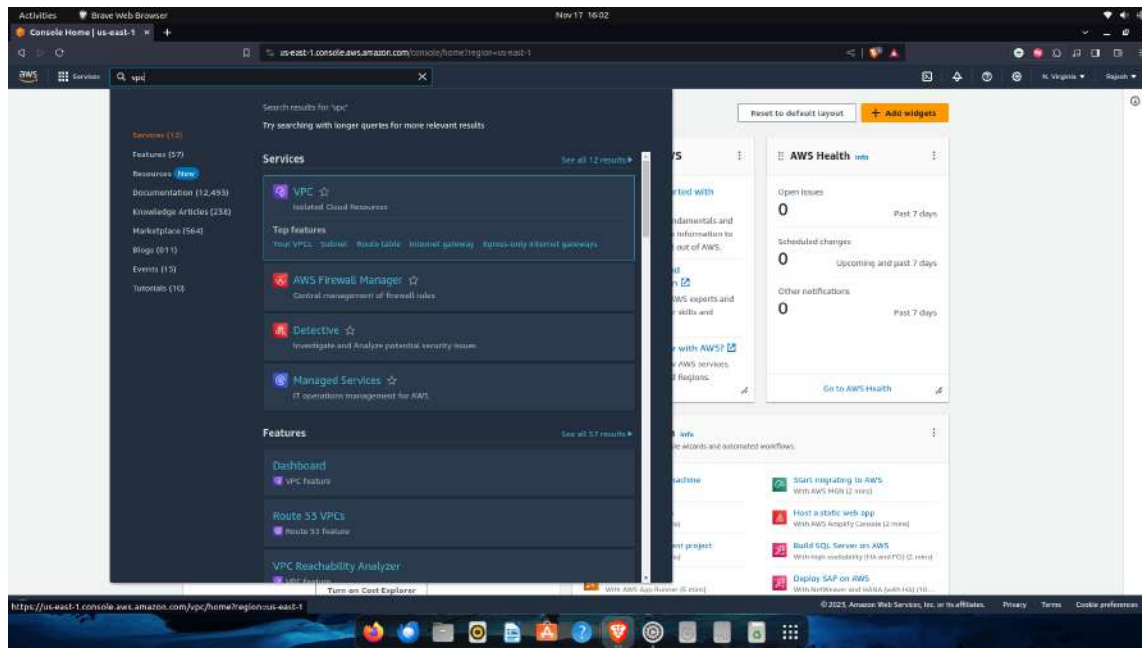
Running a Python Server in a Secured VPC

Follow these steps to implement a python server in a secured vpc along that you will gain a good knowledge about working with NACLs and Security Groups.

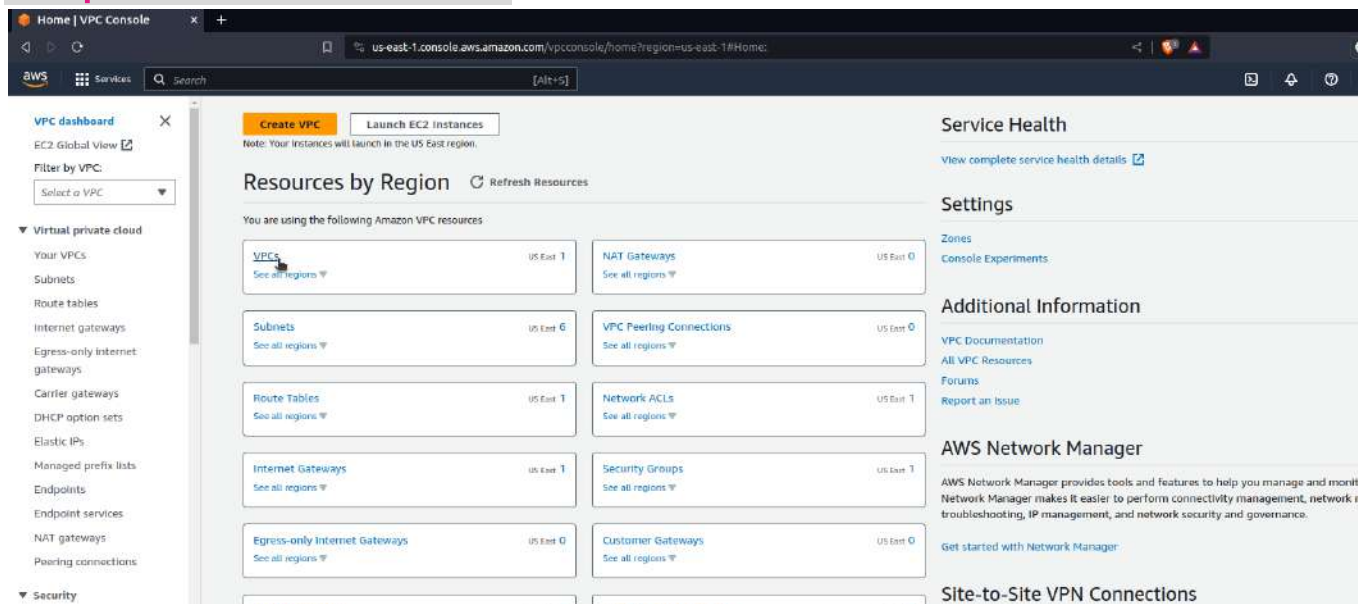
Step 1: Open your AWS Account



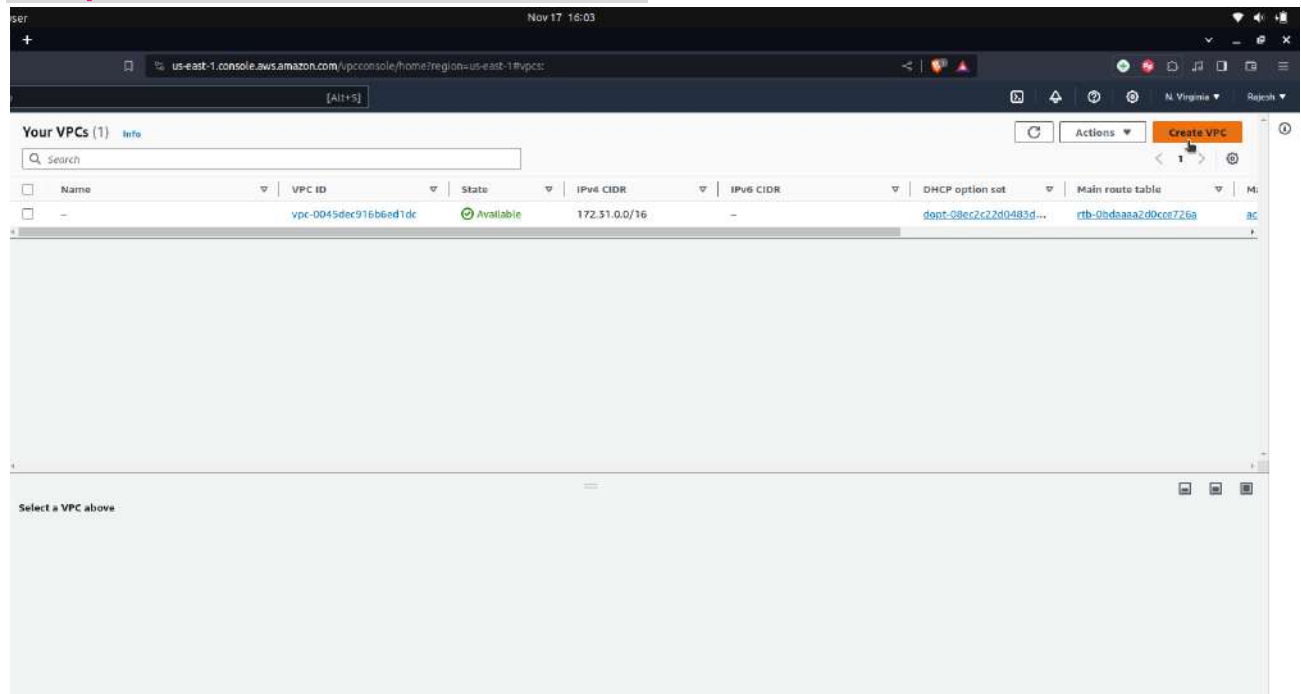
Step 2: Search for VPC in console and click on vpc



Step 3: Click on VPCs

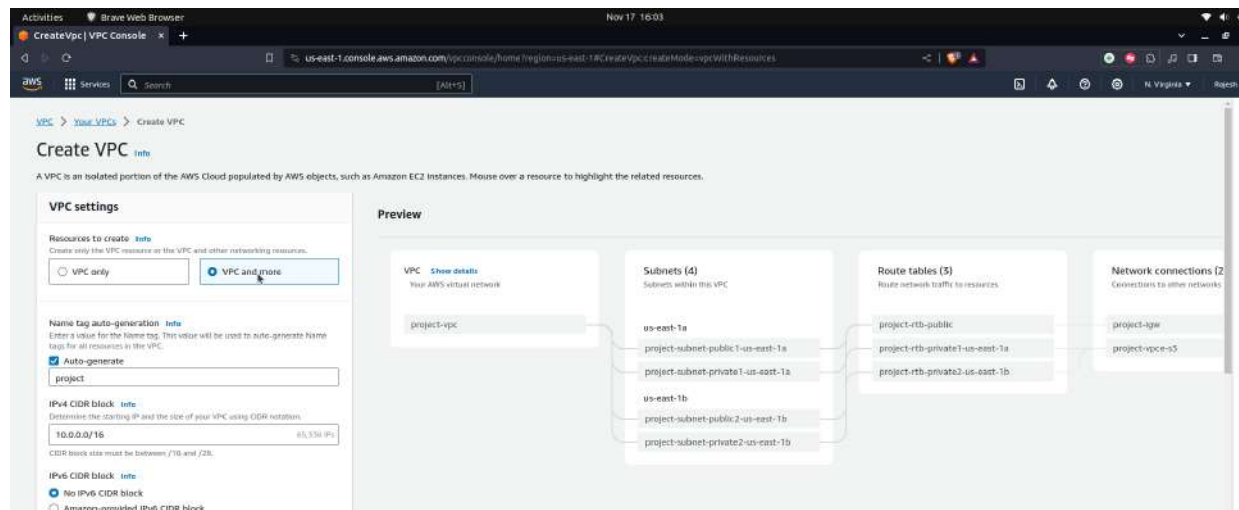


Step 4: Click on create VPC

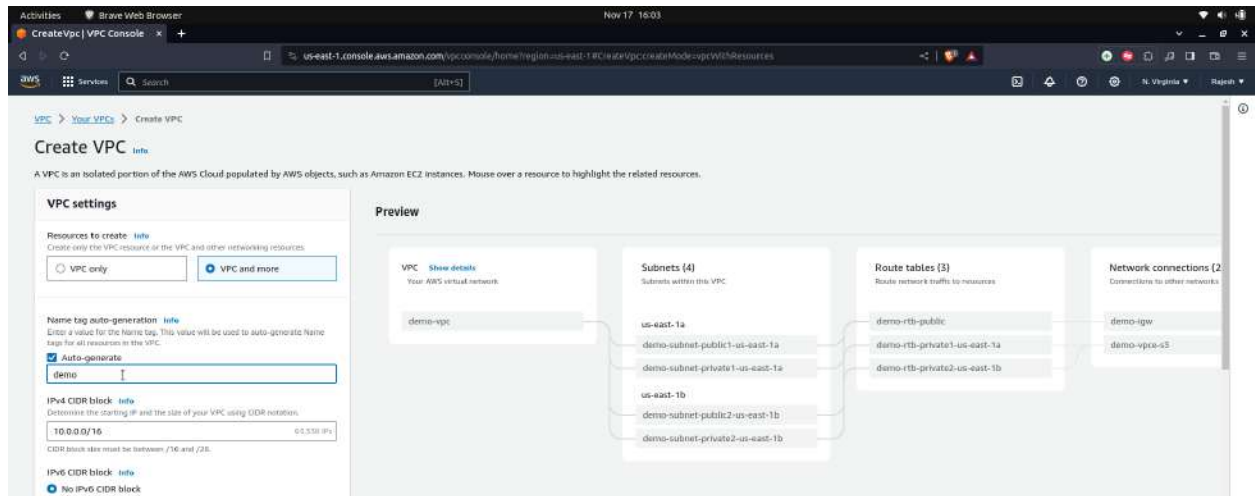


Step 5: Choose VPC and more

When we choose vpc and more ,AWS will create subnets and Route Table for us.please observe the preview to gain more understanding.

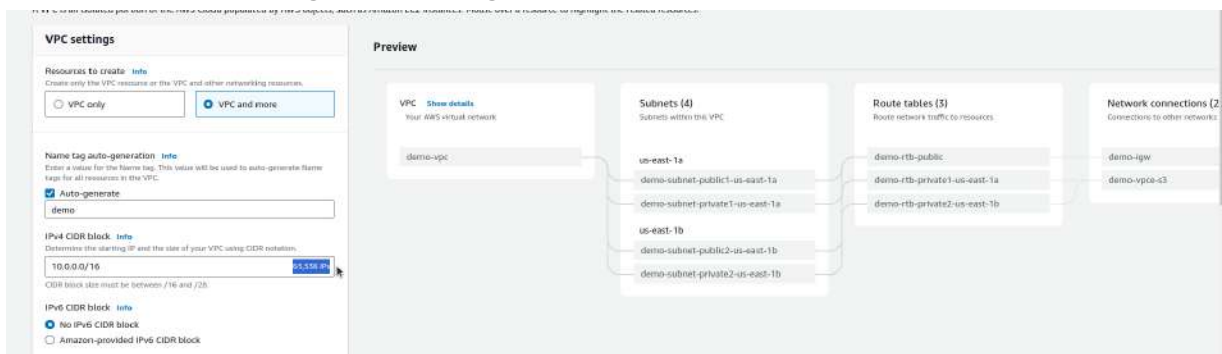


Step 6: Enter you vpc name,for now i will go with demo



Step 7: you can choose a range of IP addresses for your VPC

For now I am leaving default range



Step 8: Selecting AZ,public private subnets,NAT Gateway and VPC endpoints

You can select the number of availability zones you

Want to have your vpc.

You can set the count of subnets you want in public

As well as private

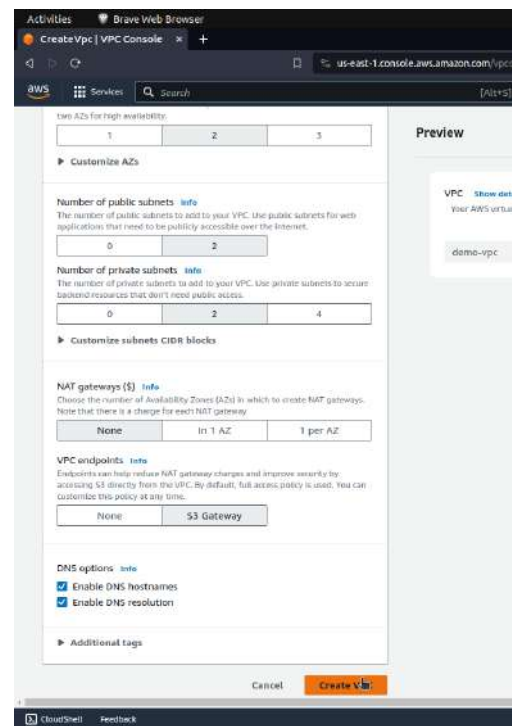
In this project I don't want to interact with the NAT gateway.

I am leaving it default.

In VPC endpoints you can choose any one either None

Or S3 Gateway.

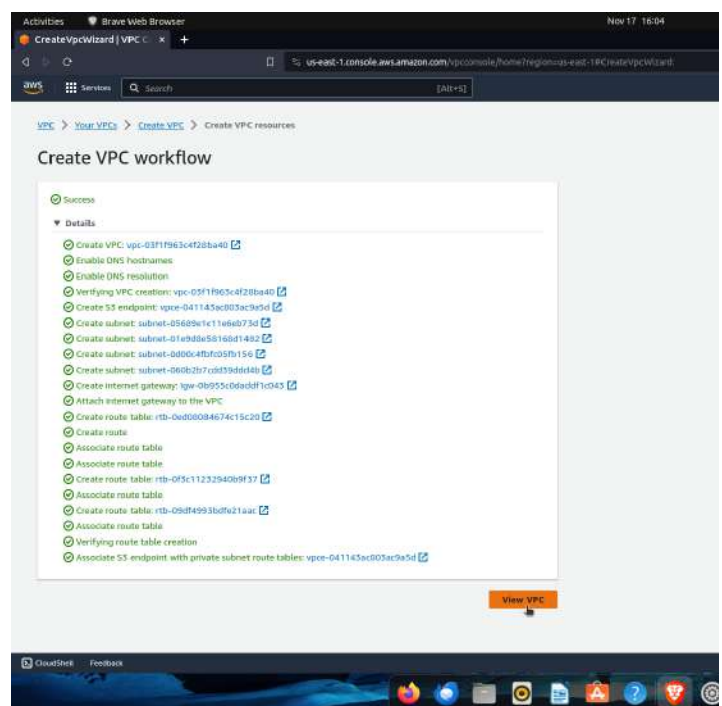
After all click on **createVPC**



Step 9: creating vpc

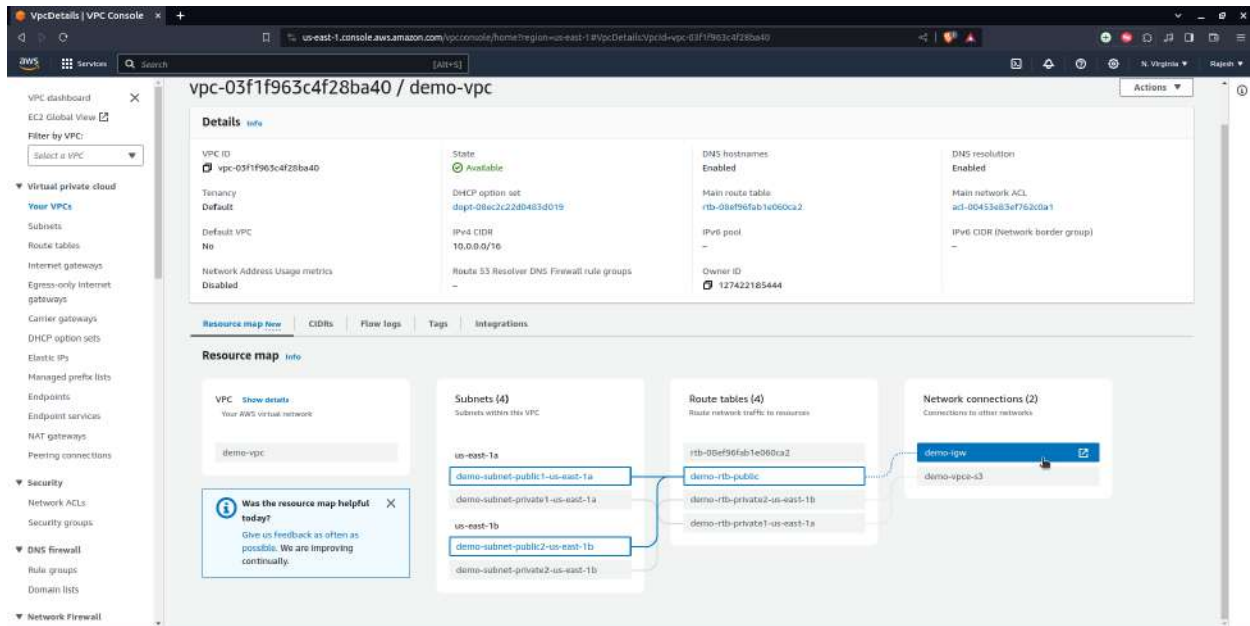
Observe how VPC is creating.

After all is done. Click on **view VPC**

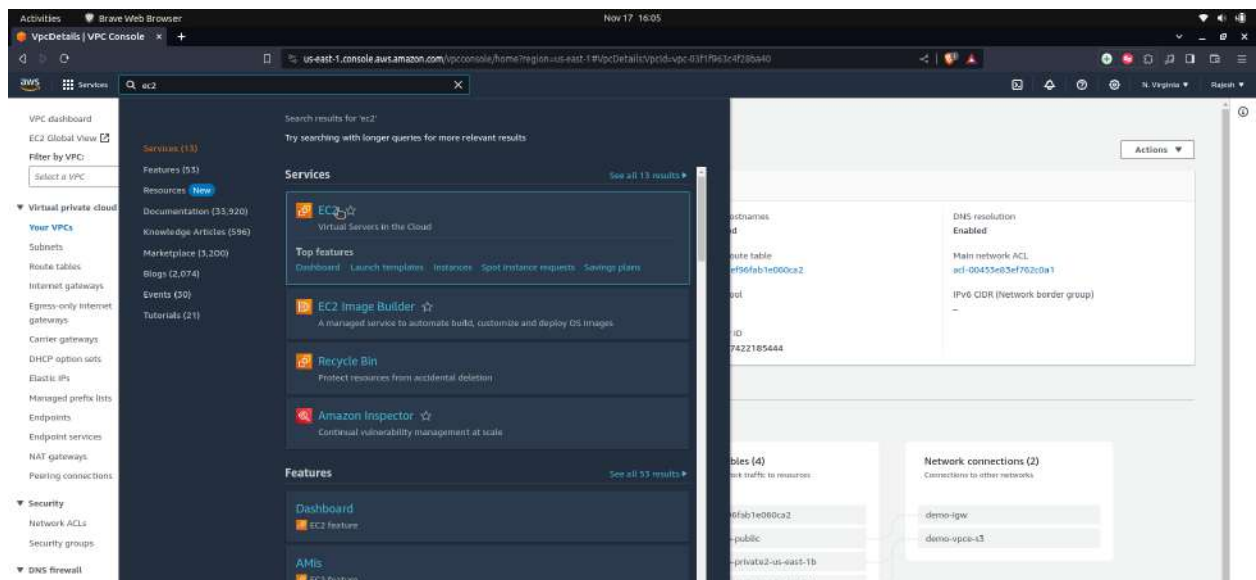


Step 10: Observe the Resource map diagram

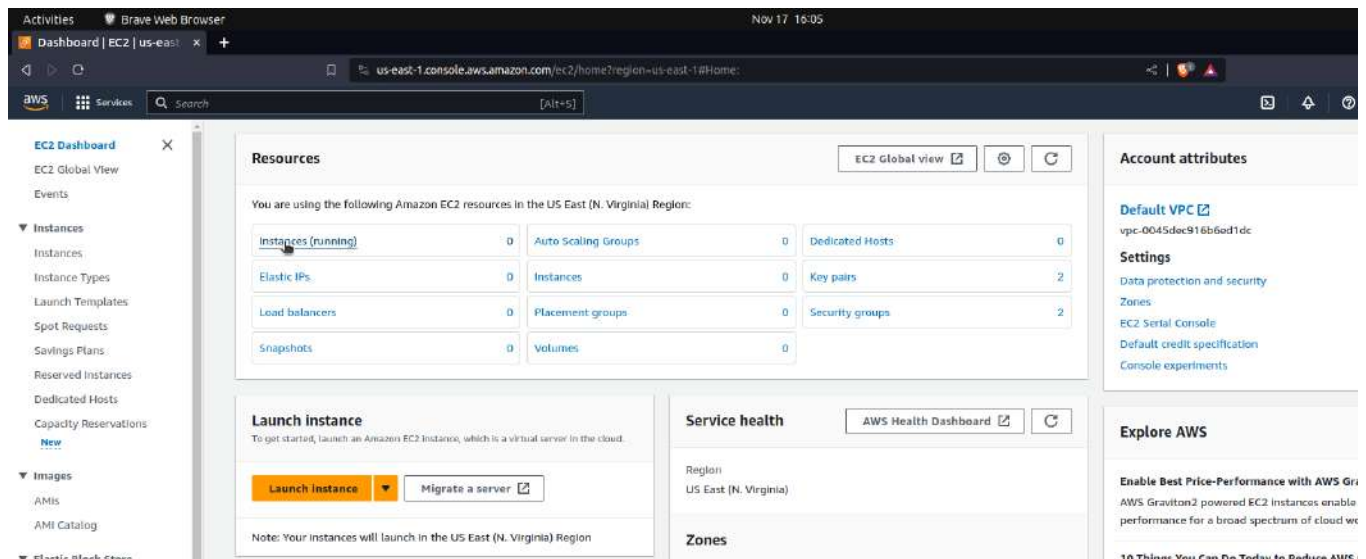
Looking into the diagram you can identify how logically it is connecting.



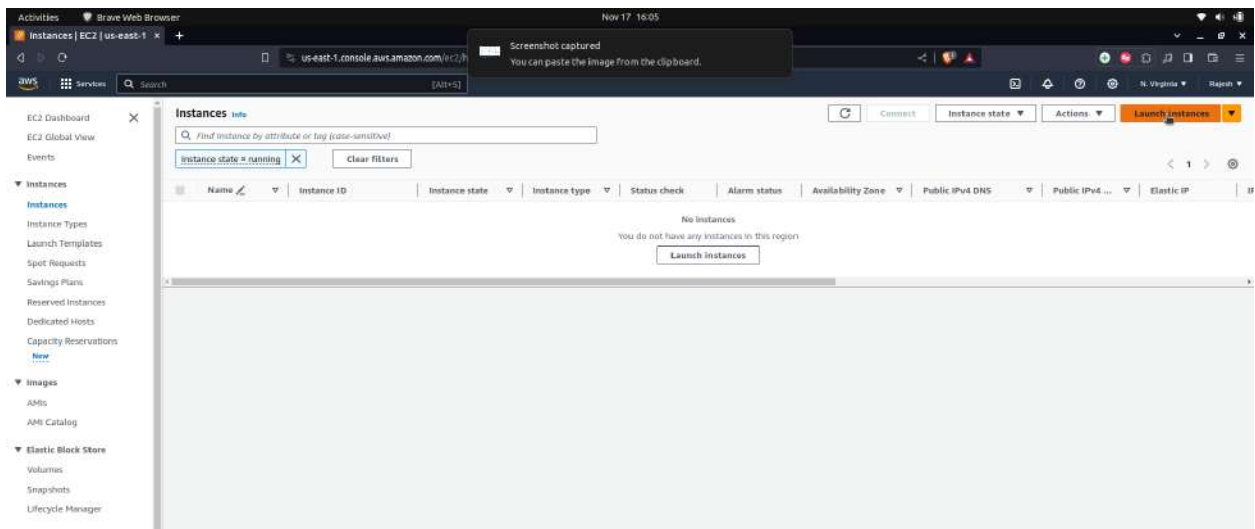
Step 11: Open Ec2



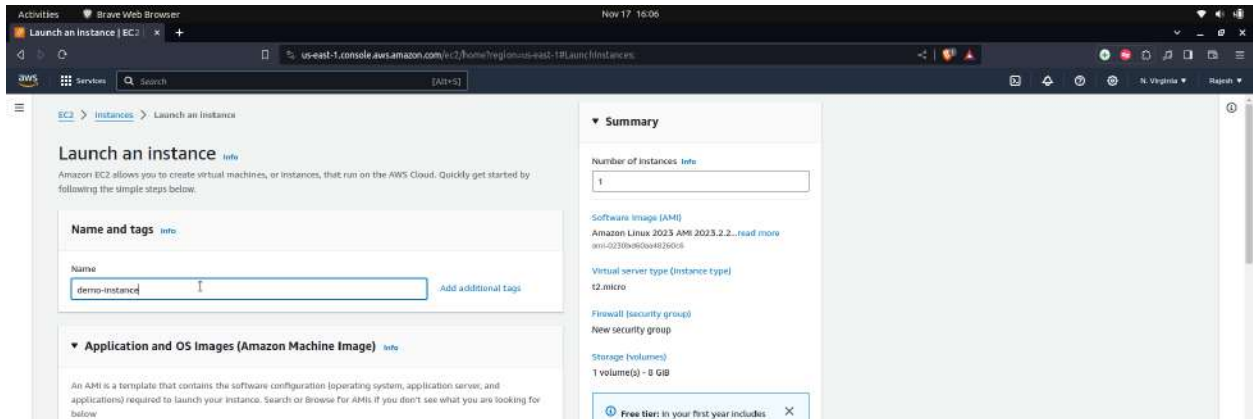
Step 12: Click on Instances Running



Step 13: Click on Launch Instance

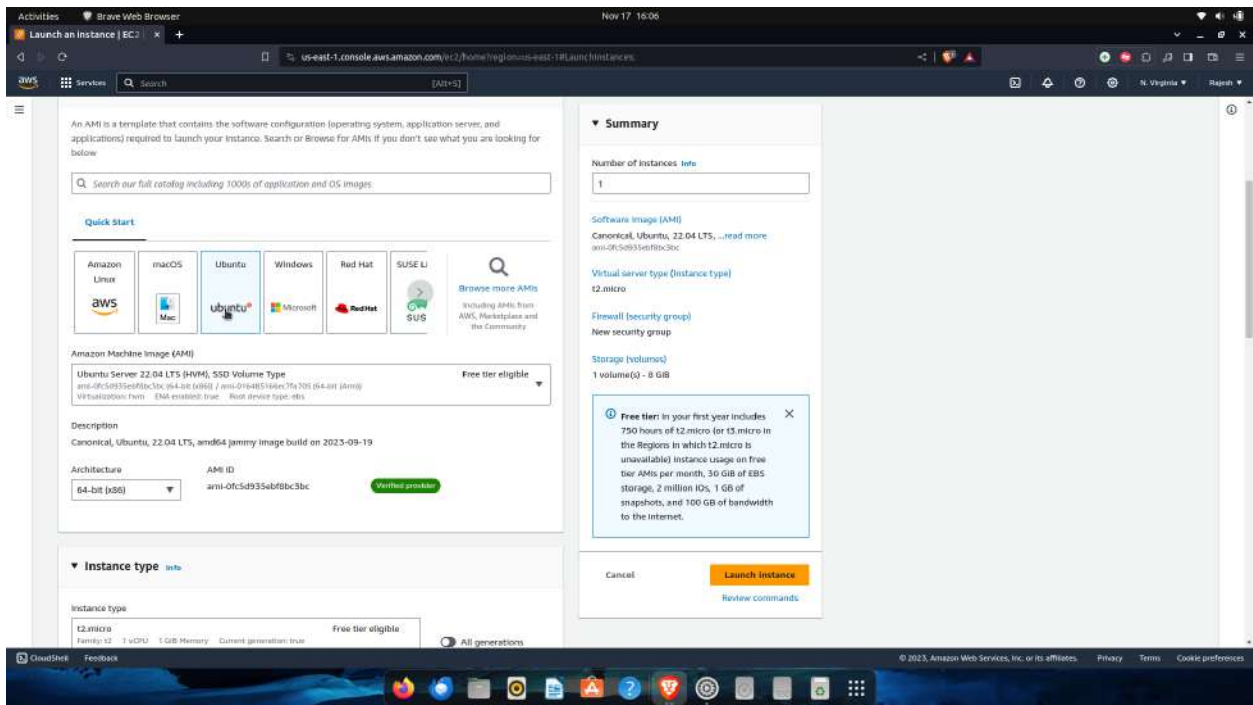


Step 14: Provide Ec2 Instance Name(demo-Instance)

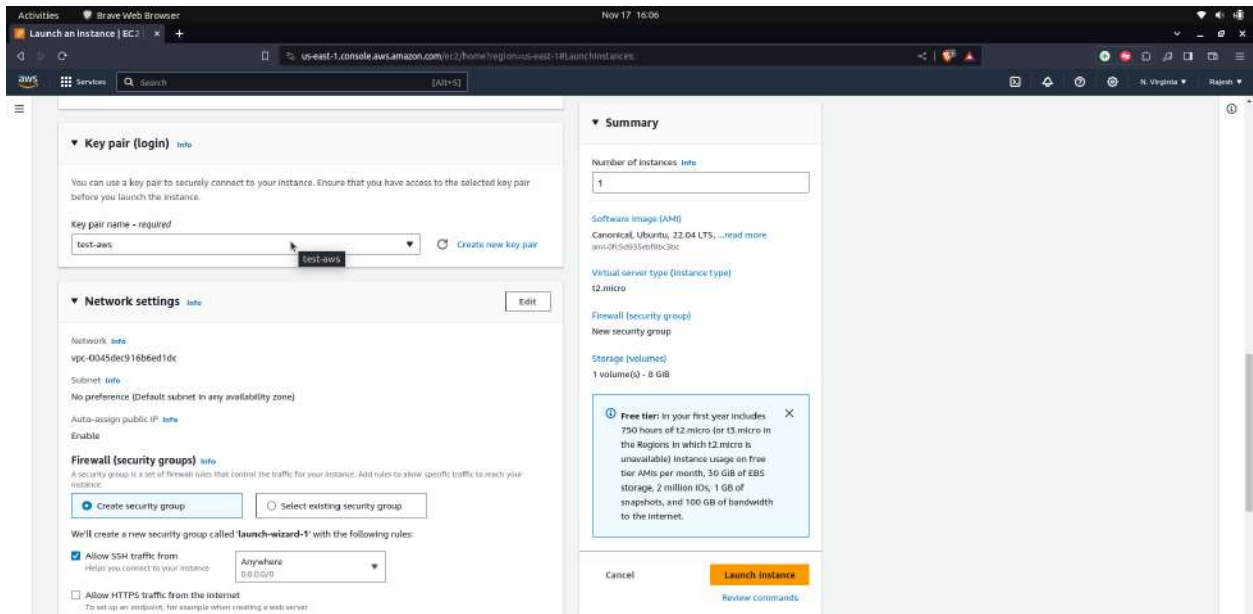


Step 15: choose ubuntu os

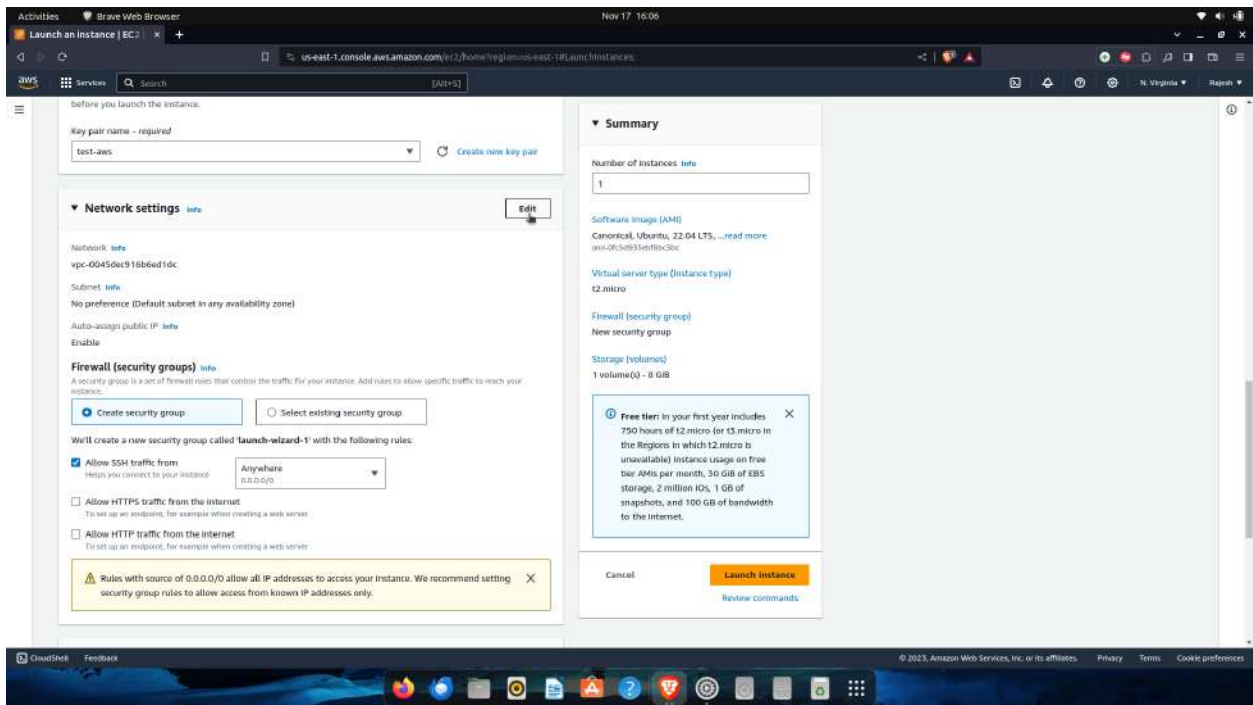
But you can choose your familiar os



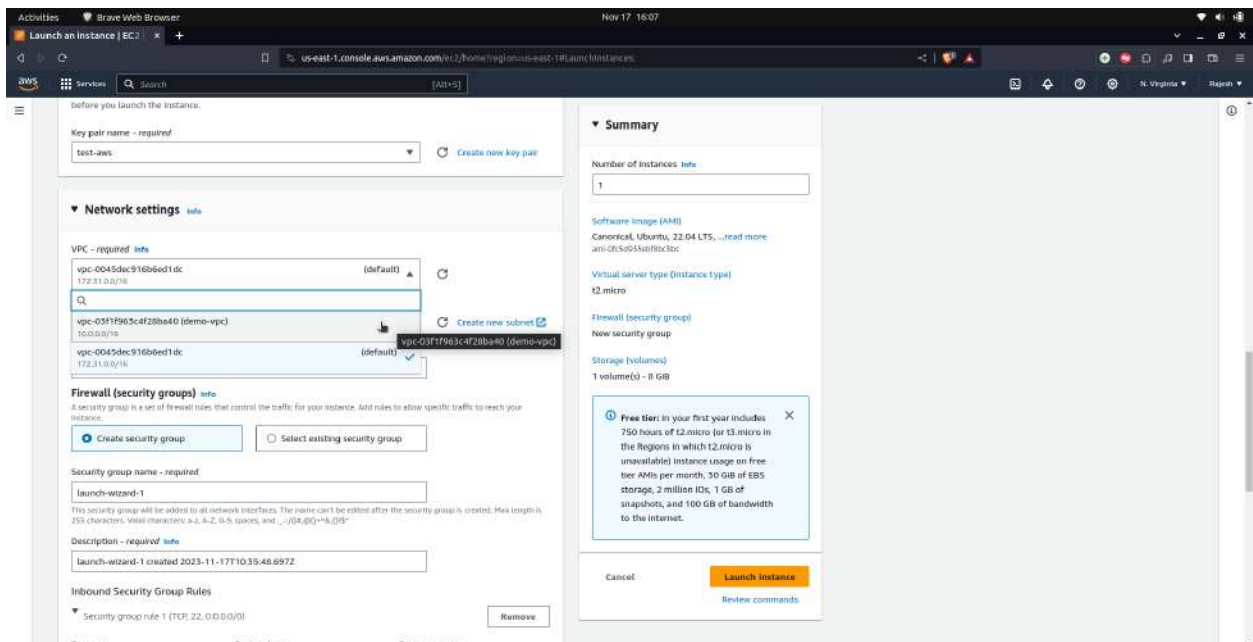
Step 16: Select key-pair



Step 17: Click Edit in Networking settings

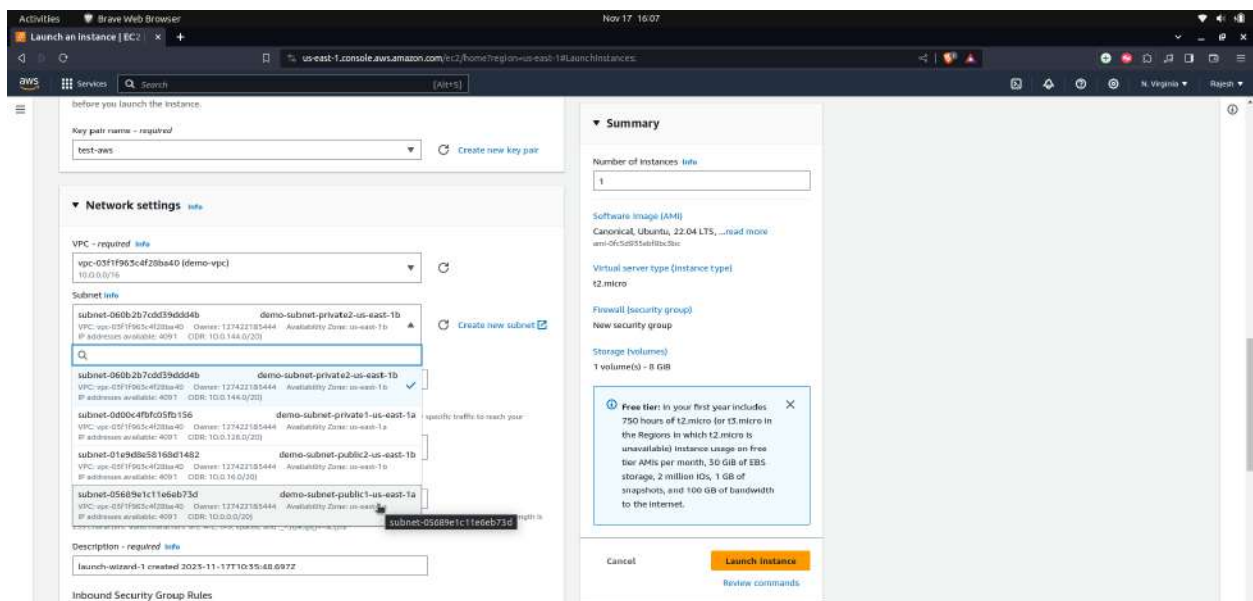


Step 18: Select our created demo vpc



Step 19: Select Subnet

Choose demo-subnet-public1-us-east-1a. In this project I don't want to deal with load balancers and other topics. Go with the public subnet in your availability zone.



Step 20: Enable Auto-assign public IP

The screenshot shows the 'Launch Instance' wizard in the AWS Management Console, specifically the 'Network settings' step. The 'Auto-assign public IP' option is selected and set to 'Enable'. The 'VPC' is set to 'vpc-031f963c4f28ba40 (demo-vpc)' and the 'Subnet' is 'subnet-05689e1c1e1eb75d (demo-subnet-public-us-east-1a)'. The 'Security group' is set to 'launch-wizard-1'. The 'Summary' panel on the right shows the instance configuration: 1 instance, Canonical Ubuntu 22.04 LTS, t2.micro instance type, and 1 volume (8 GB). A 'Free tier' notification is visible, stating that the first year includes 750 hours of t2.micro usage.

Key configuration details from the screenshot:

- Auto-assign public IP:** Enable
- VPC:** vpc-031f963c4f28ba40 (demo-vpc)
- Subnet:** subnet-05689e1c1e1eb75d (demo-subnet-public-us-east-1a)
- Security group:** launch-wizard-1
- Summary:** 1 instance, Canonical Ubuntu 22.04 LTS, t2.micro, 1 volume (8 GB)

Step 21: Select create security group

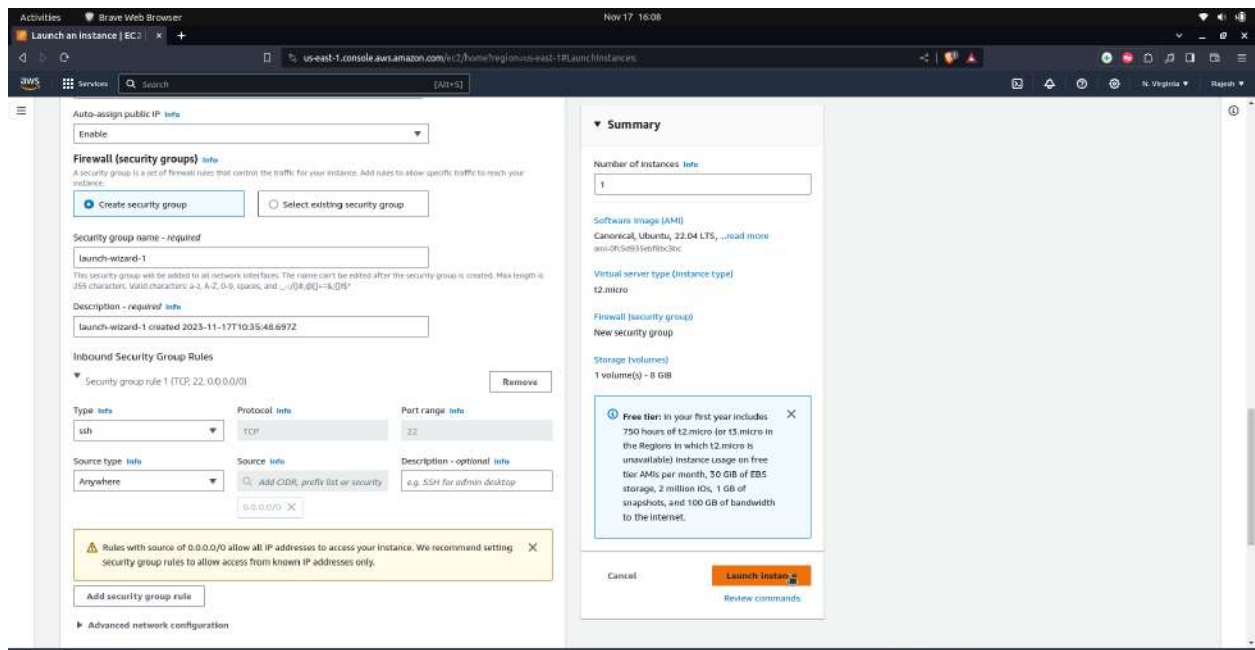
AWS will automatically create the security group for you then you can manage it

The screenshot shows the 'Launch Instance' wizard in the AWS Management Console, specifically the 'Firewall (security groups)' step. The 'Create security group' button is highlighted. The 'Security group name' is 'launch-wizard-1'. The 'Inbound Security Group Rules' section shows a rule for 'ssh' on 'TCP' port '22' from 'Anywhere' (0.0.0.0/0). A warning message states: 'Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.' The 'Summary' panel on the right shows the instance configuration: 1 instance, Canonical Ubuntu 22.04 LTS, t2.micro, and 1 volume (8 GB). A 'Free tier' notification is visible, stating that the first year includes 750 hours of t2.micro usage.

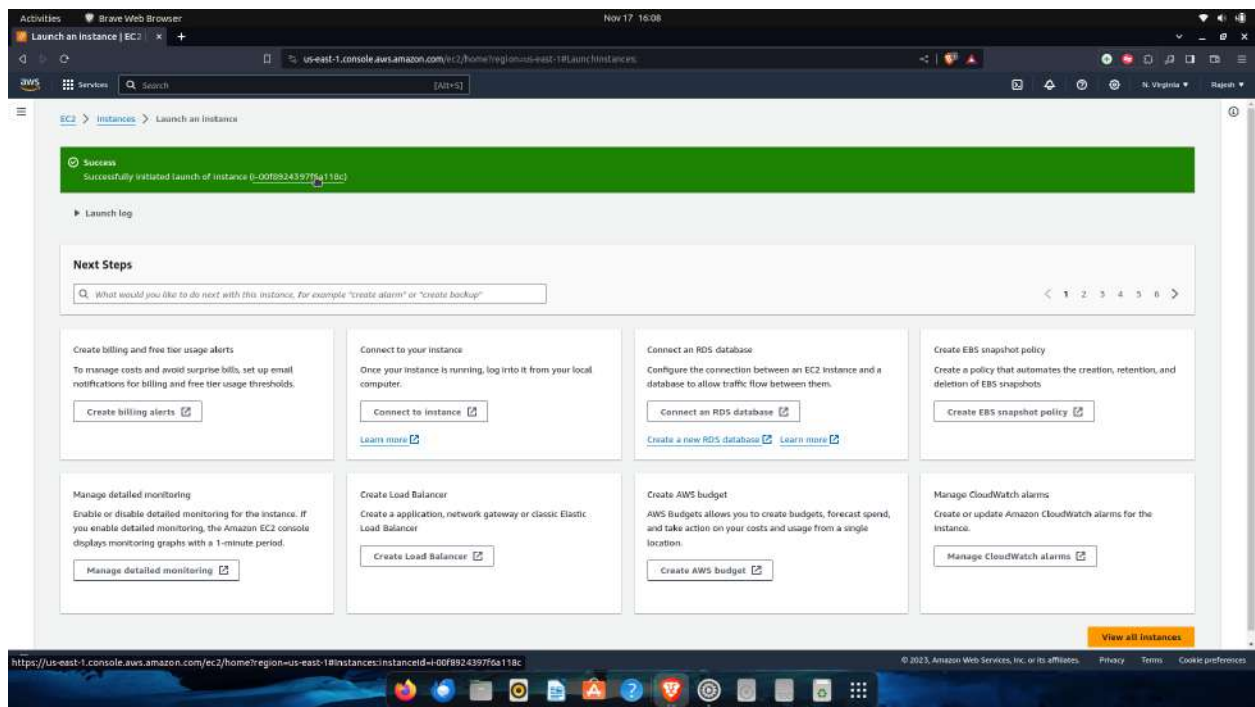
Key configuration details from the screenshot:

- Auto-assign public IP:** Enable
- Firewall (security groups):** Create security group
- Security group name:** launch-wizard-1
- Inbound Security Group Rules:** ssh, TCP, 22, Anywhere (0.0.0.0/0)
- Summary:** 1 instance, Canonical Ubuntu 22.04 LTS, t2.micro, 1 volume (8 GB)

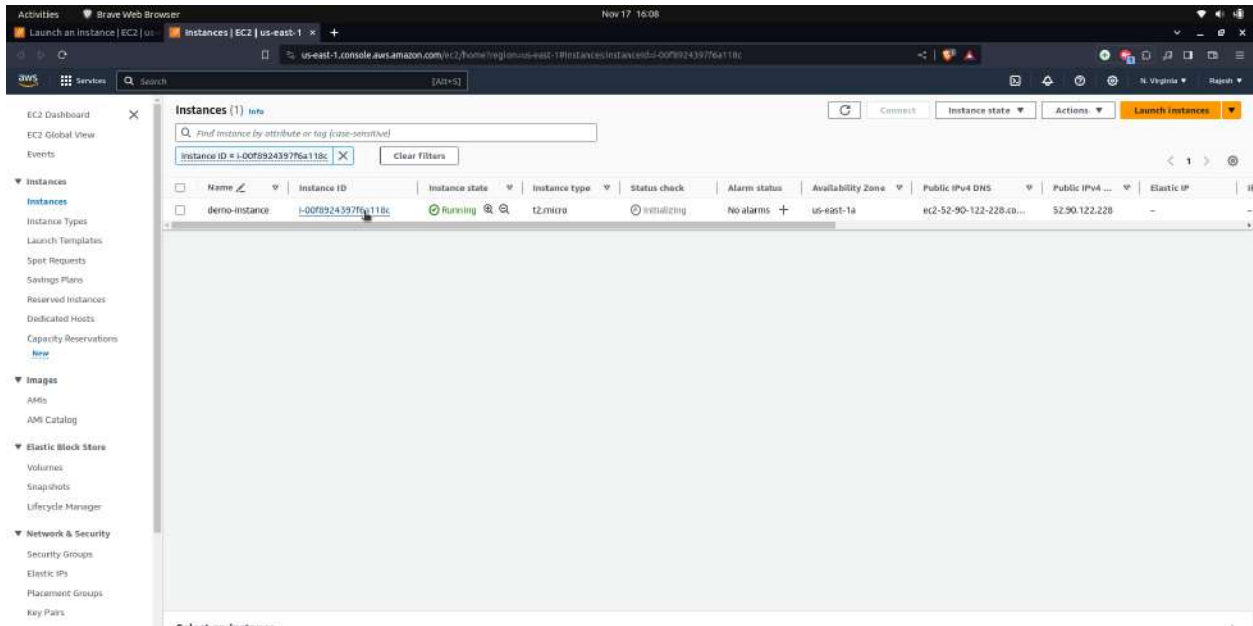
Step 22: Click on Launch Instance



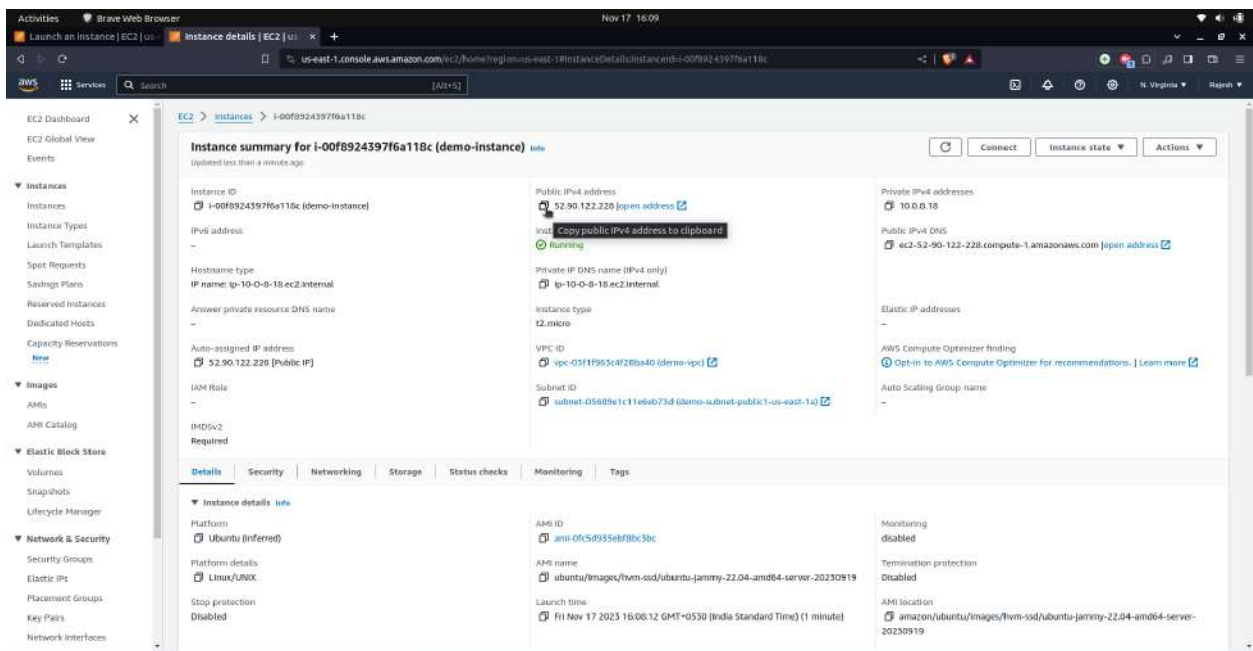
Step 23: After creating open your Instance



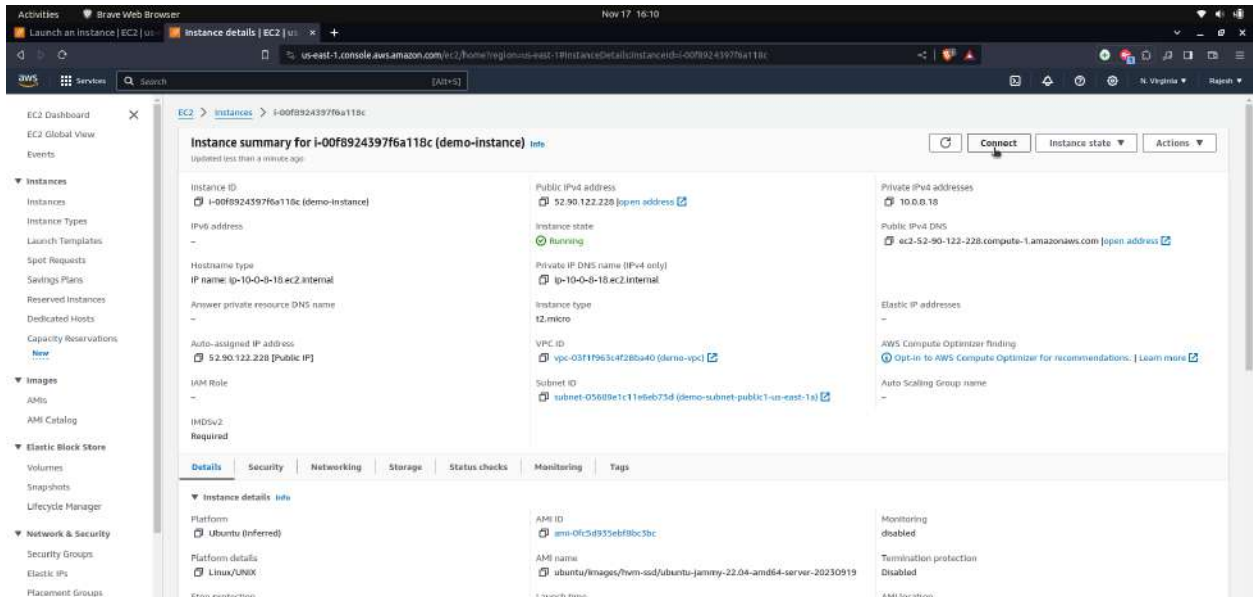
Step 24: Click on your Instance ID



Step 25: Copy your Instance public IP address

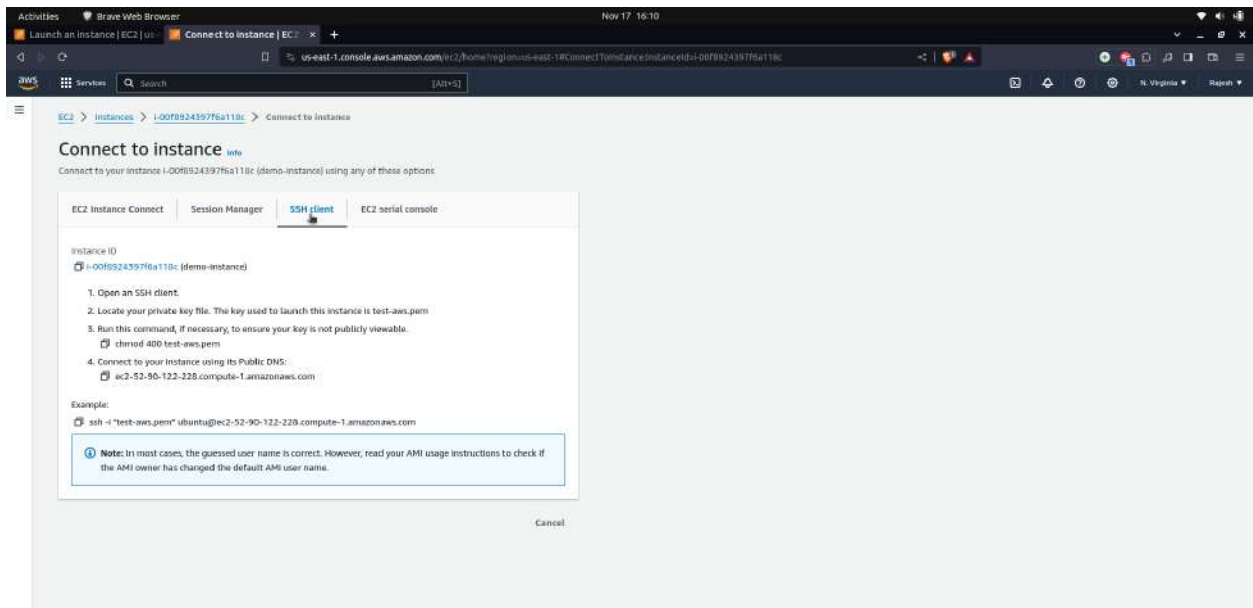


Step 26: Click on Connect

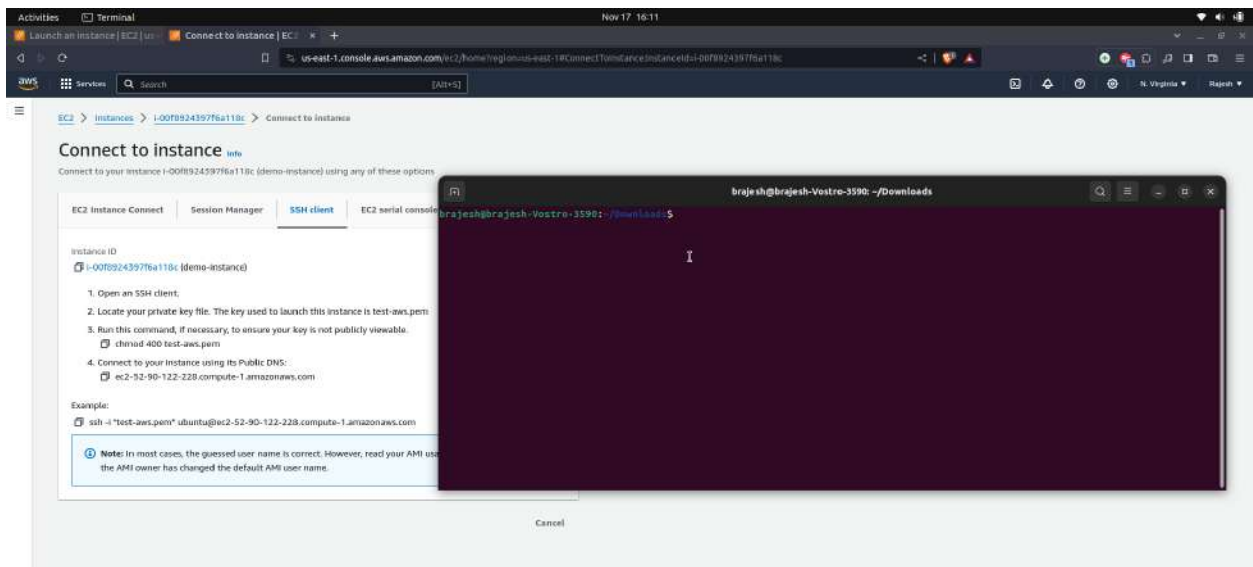


Step 27: Click on SSH client

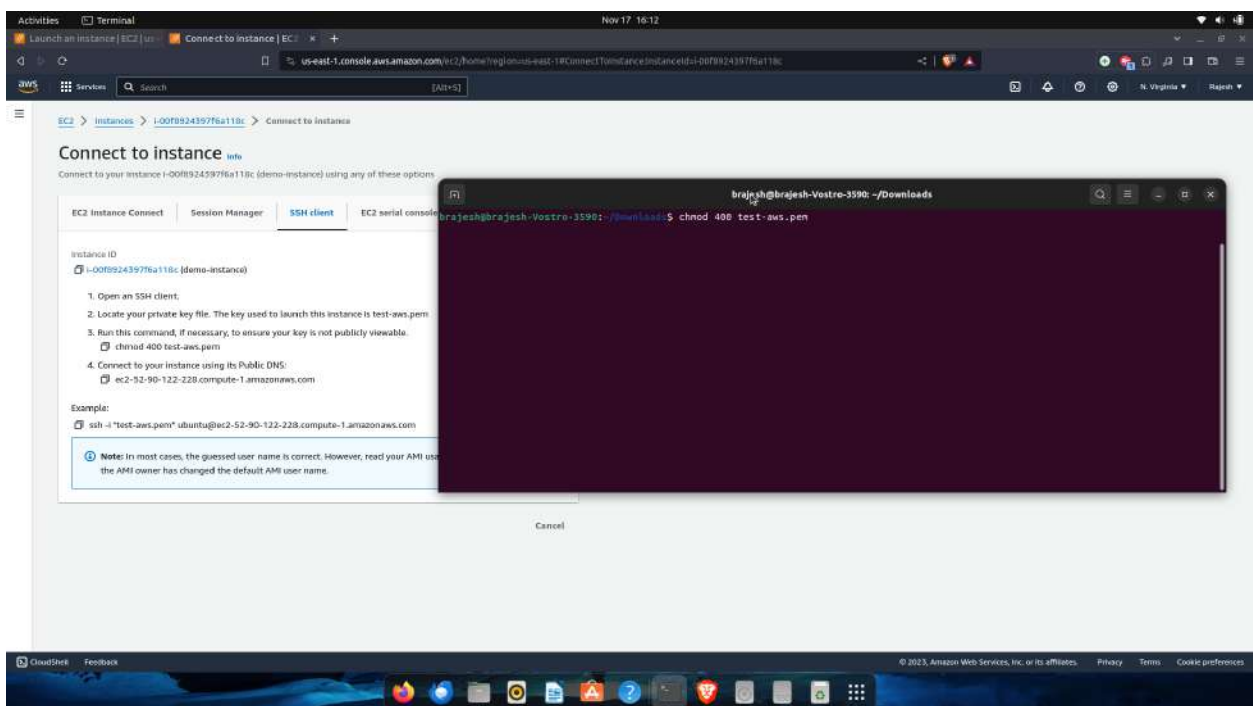
Follow the given steps to connect your ec2 instance through terminal.let's go.



Step 28: Open your Terminal



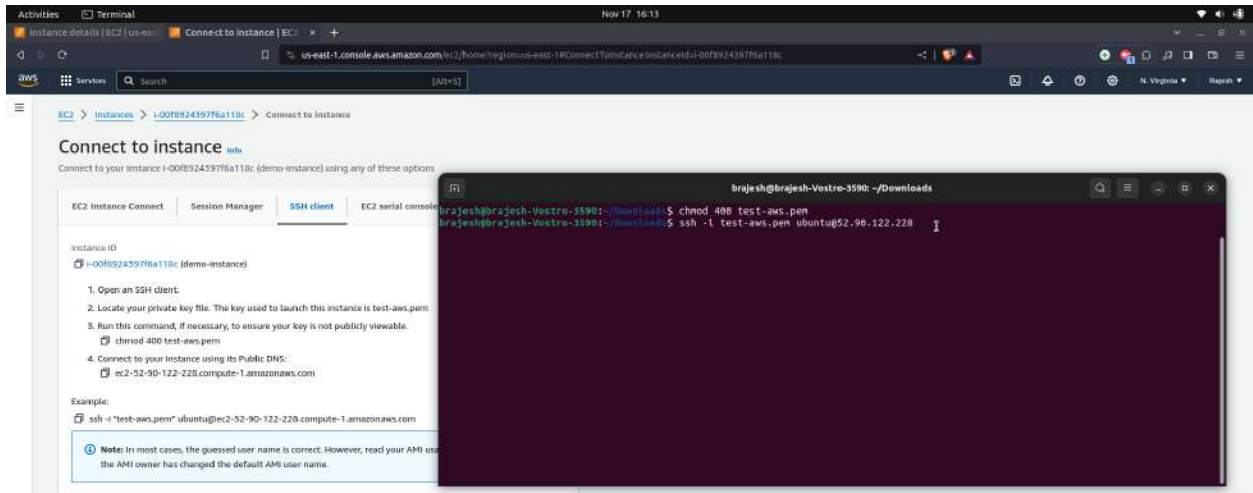
Step 29: Provide 400 permission to your .pem file



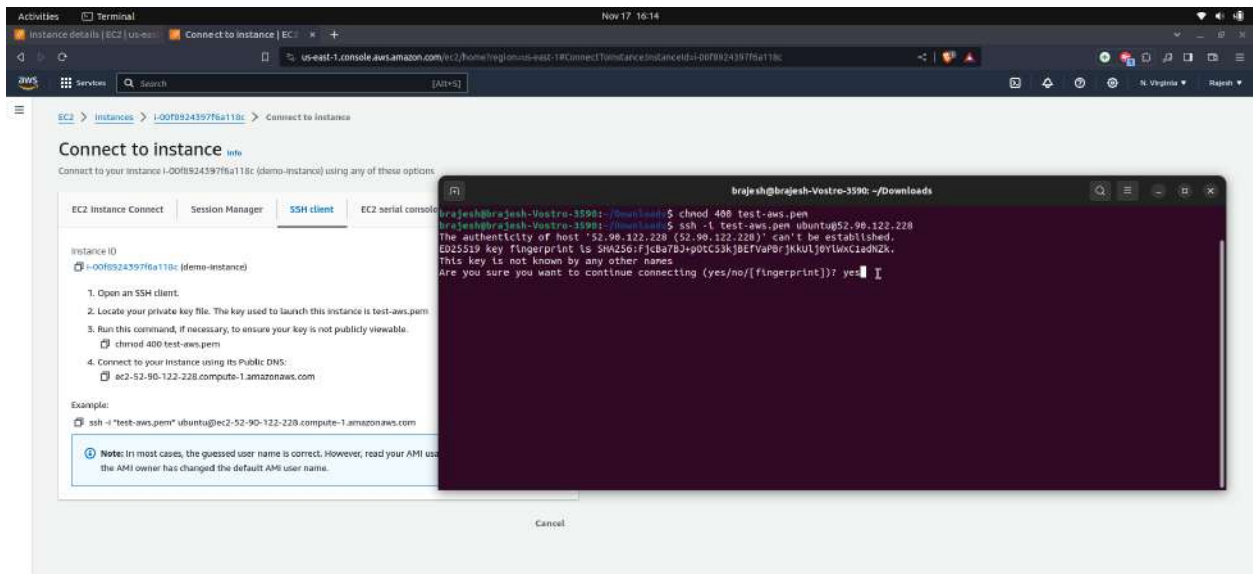
Step 30: run ssh command to connect

Run this command : **`ssh -i test-aws.pem ubuntu@ipaddress`**

Replace ip address with your instance public ip address.



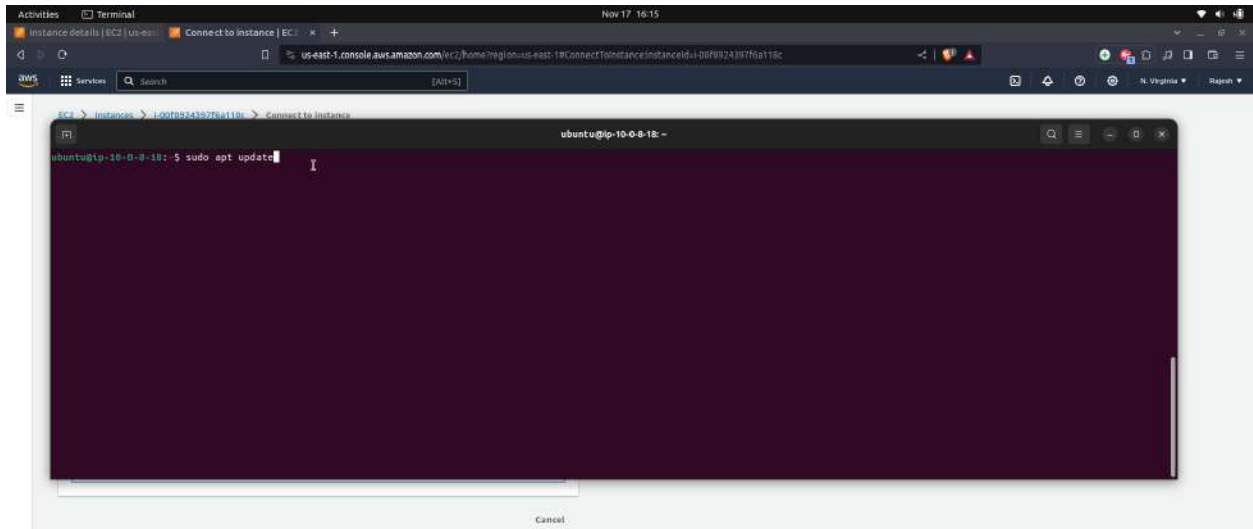
Step 31: Type yes



Step 32: Best practice to update your instance

After successfully connecting to your instance.

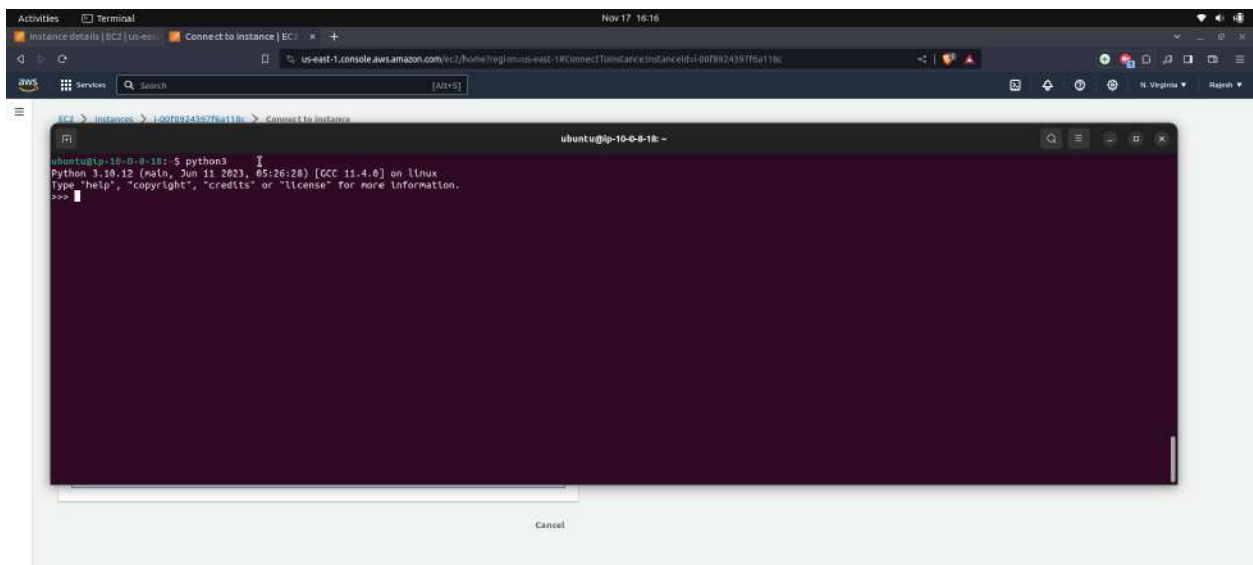
Run this command : **sudo apt update**



Step 33: Check python3 is available or not ?

Run this command : **python3**

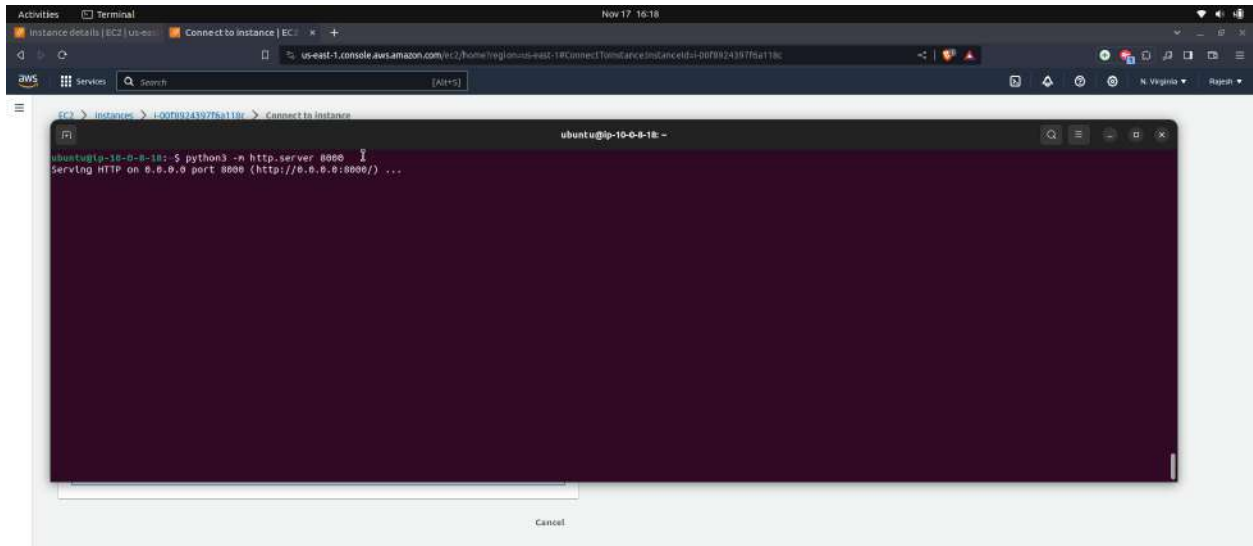
If you don't have it ,then install it.



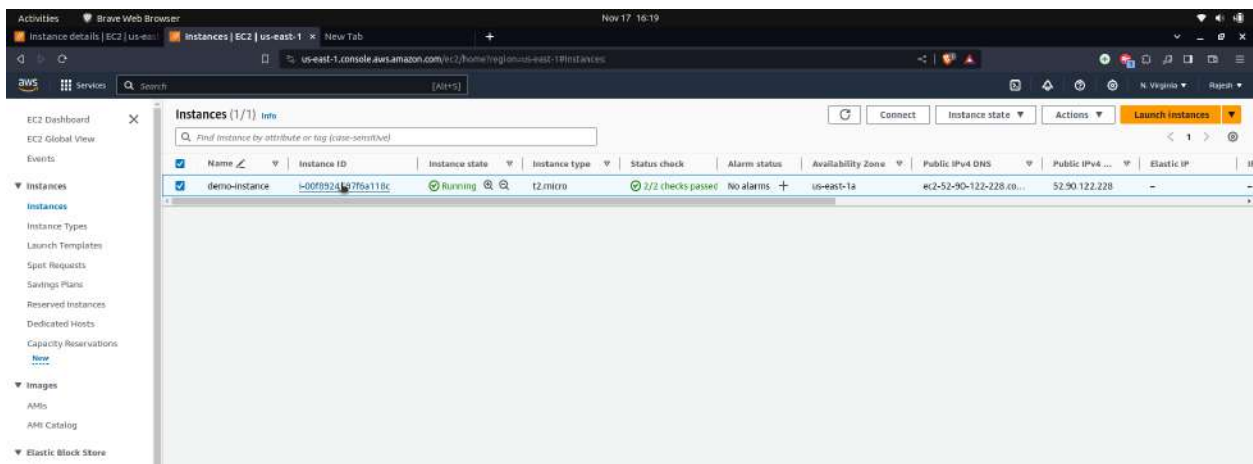
Step 34: Now run the basic python server

Run this command: **python3 -m http.server 8000**

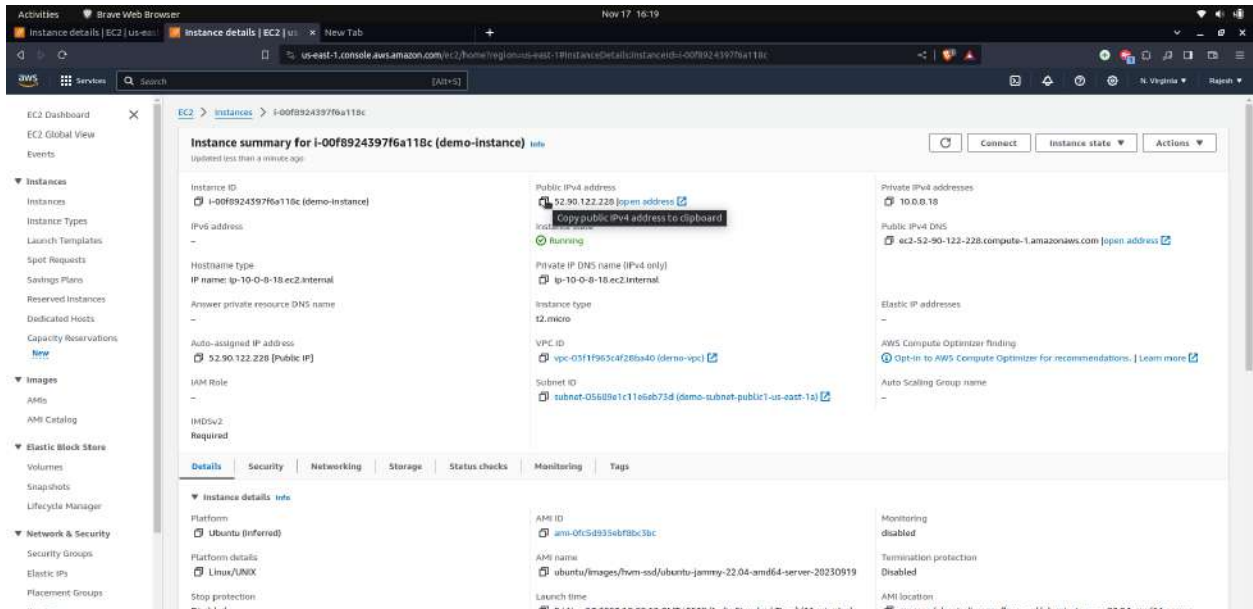
It Will start a basic http server in our Instance.you can access it from port 8000.



Step 35: Go to your Instance



Step 36: copy your Instance public IP address

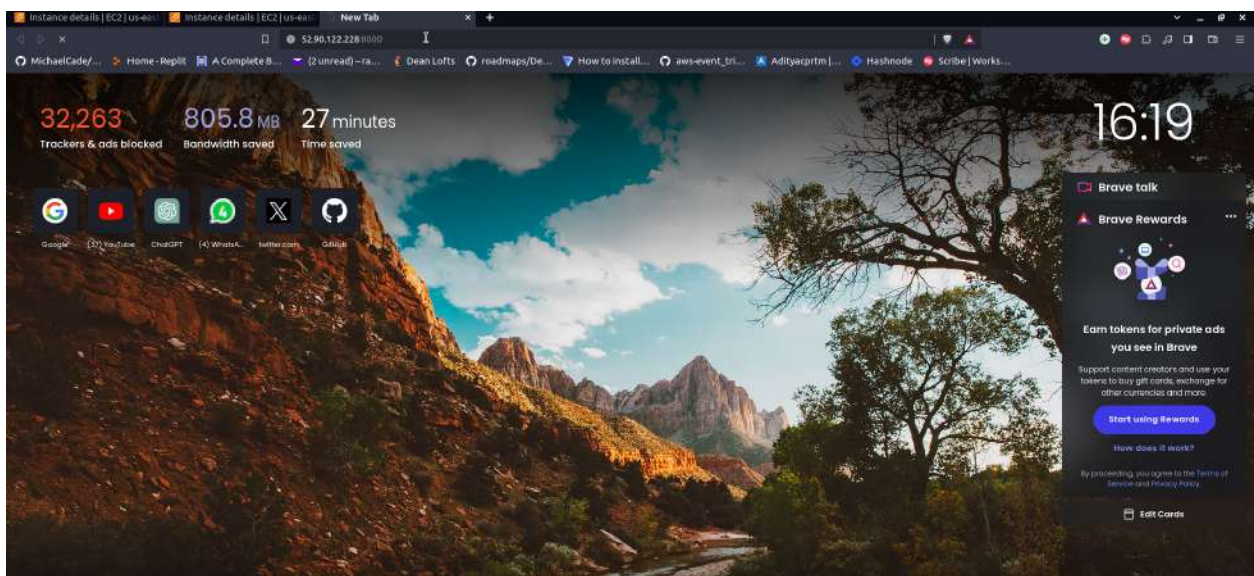


Step 37: Try to access our server

Open a new tab in your browser and type like this in your address bar:

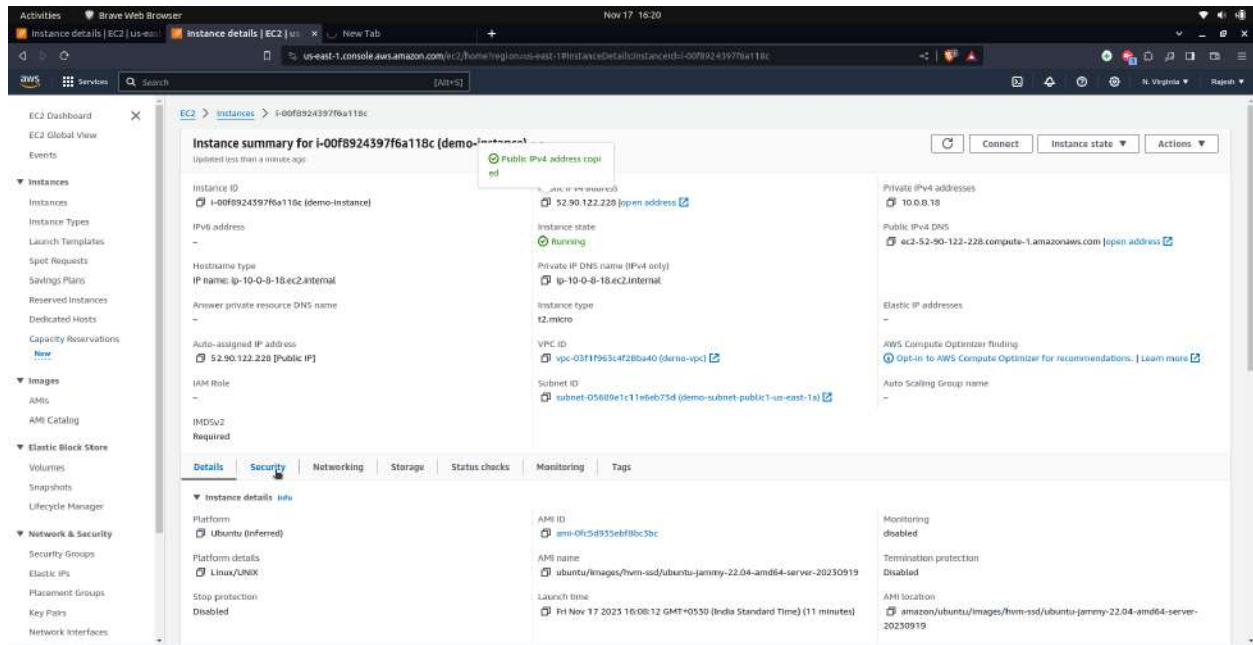
“<http://copiedipaddress:8000>”. Replace copiedipaddress with your instance public ip address.

You can't open it. By default the security groups won't allow traffic inbound.

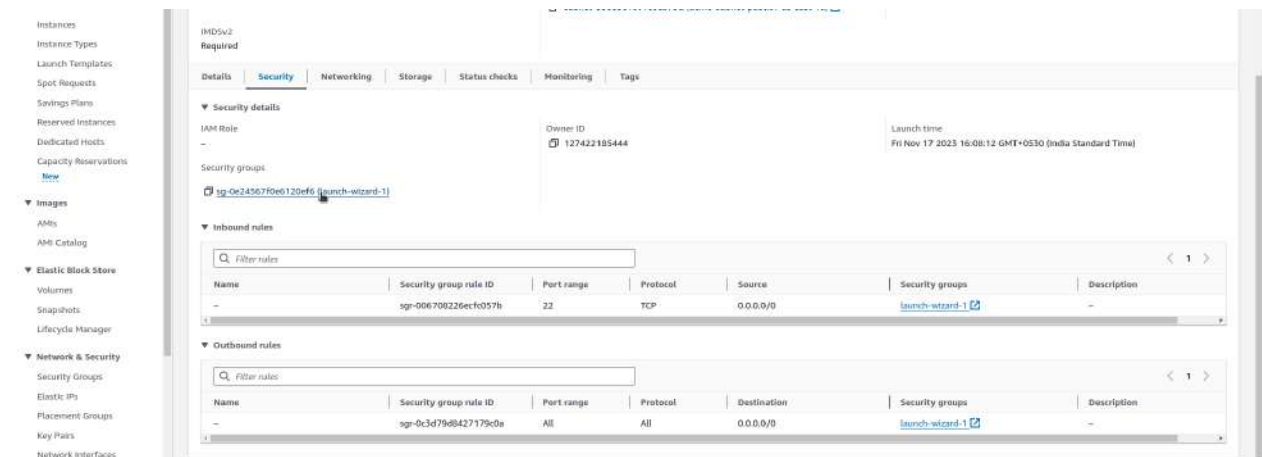


Step 38: Go back to your instance settings (security)

Open your instance tab and click on security tab



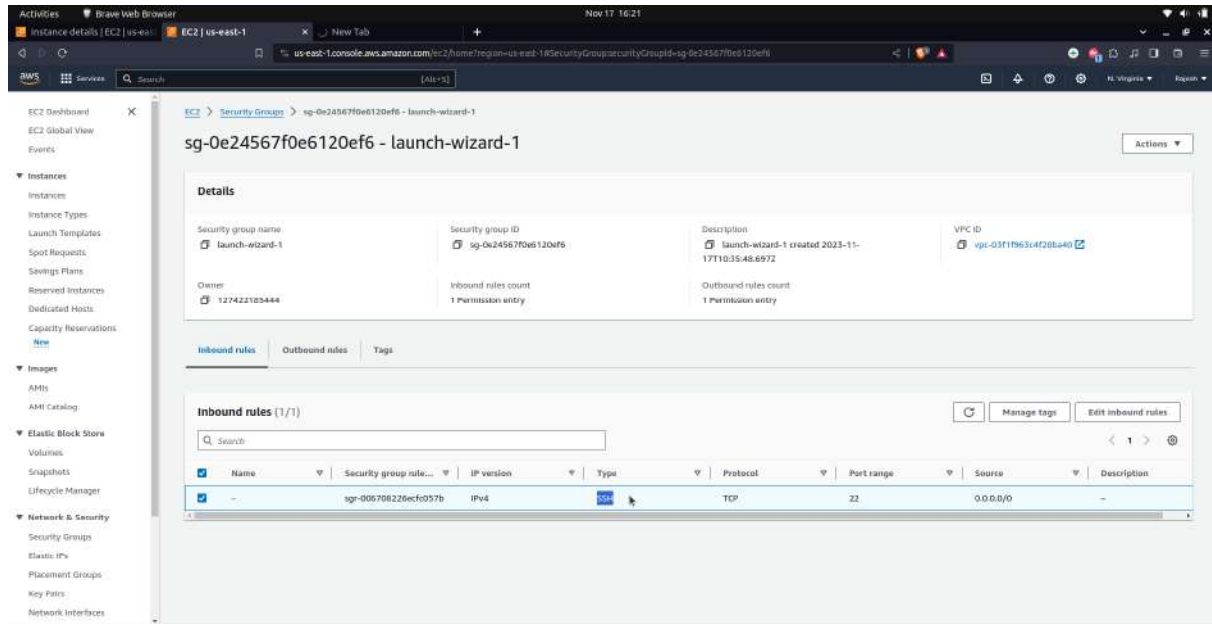
Step 39: Click on security groups



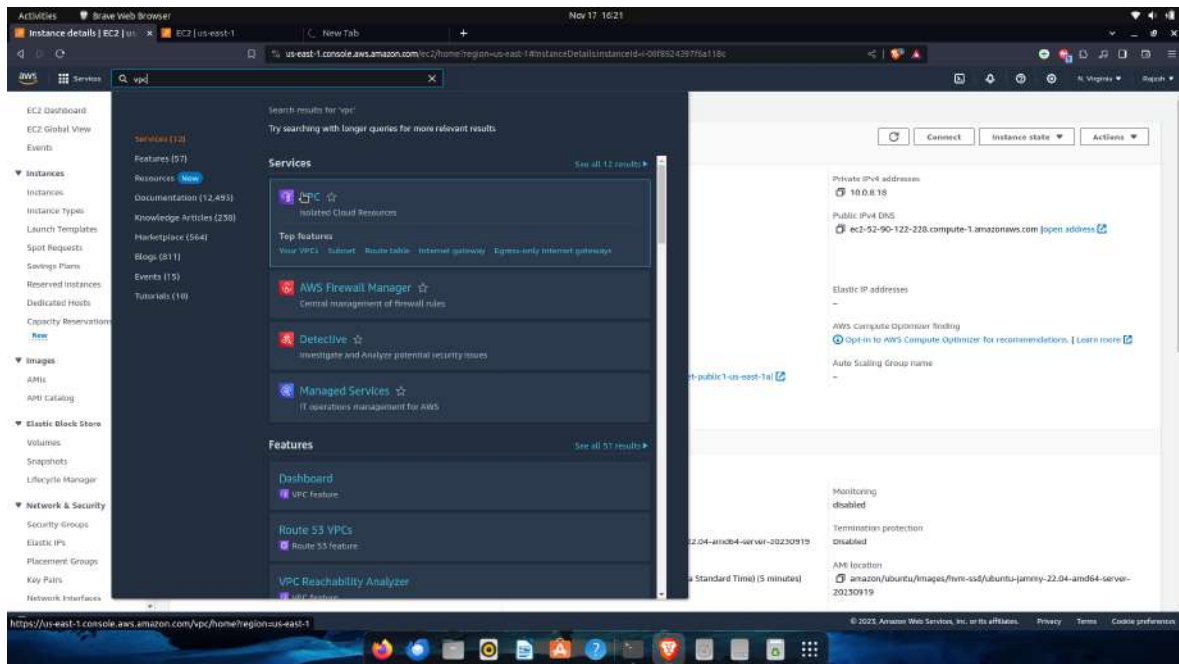
Step 40: Observe a bit

Initially the security groups are allowing port 22 only for ssh. Now we need to add a new rule.

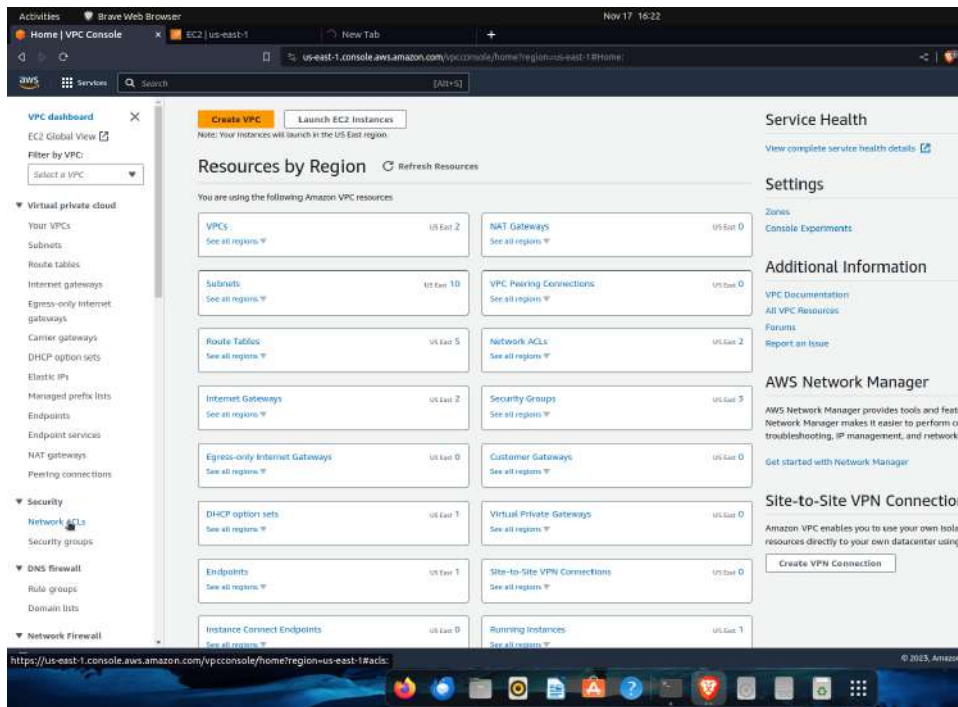
Before that we will check NACLs



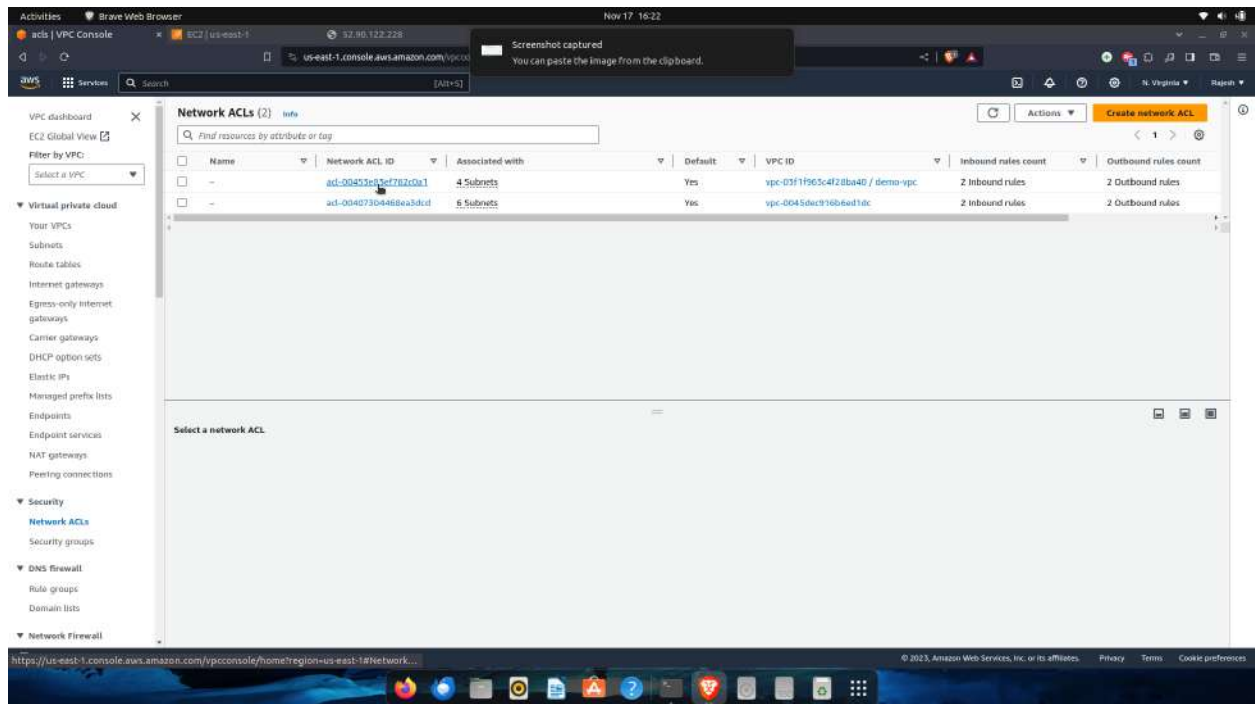
Step 41: Open VPC



Step 42: Click “network NACLs” in under “security”



Step 43: Click on your demo vpc



Step 44: Observe the inbound rules

NACLs allowing all traffic. Here we got an idea that there is no problem in VPC NACLs

The screenshot shows the AWS Management Console for a Network ACL (acl-00453e83ef762c0a1) in the us-east-1 region. The console displays the details of the Network ACL, including its ID, associated subnets (4), and the VPC ID (vpc-03f1f963c4f28ba40). The 'Inbound rules' tab is selected, showing a table with one rule (Rule number 100) that allows all traffic from 0.0.0.0/0 on all ports and protocols. The rule is associated with the VPC ID vpc-03f1f963c4f28ba40.

Rule number	Type	Protocol	Port range	Source	Allow/Deny
100	All traffic	All	All	0.0.0.0/0	Allow

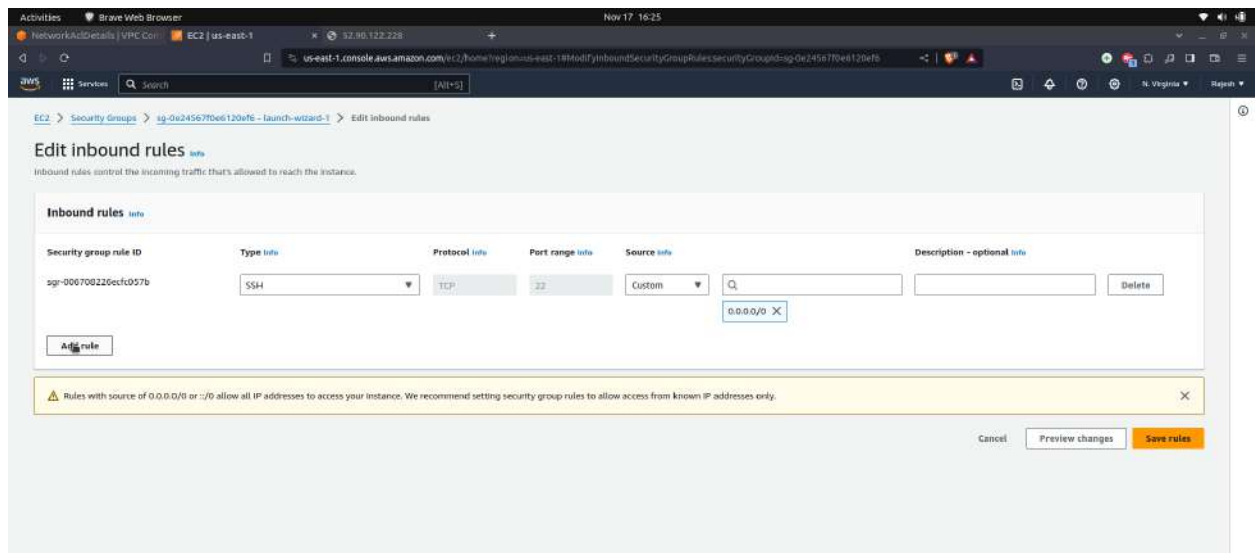
Step 45: Come back to ec2 instance security groups

Click on **Edit Inbound Rules**

The screenshot shows the AWS Management Console for a Security Group (sg-0e24567f0e6120ef6) in the us-east-1 region. The console displays the details of the Security Group, including its name (launch-wizard-1), ID (sg-0e24567f0e6120ef6), and VPC ID (vpc-03f1f963c4f28ba40). The 'Inbound rules' tab is selected, showing a table with one rule (Rule name sg-006708226ecf057b) that allows SSH traffic from 0.0.0.0/0 on port 22. The rule is associated with the VPC ID vpc-03f1f963c4f28ba40.

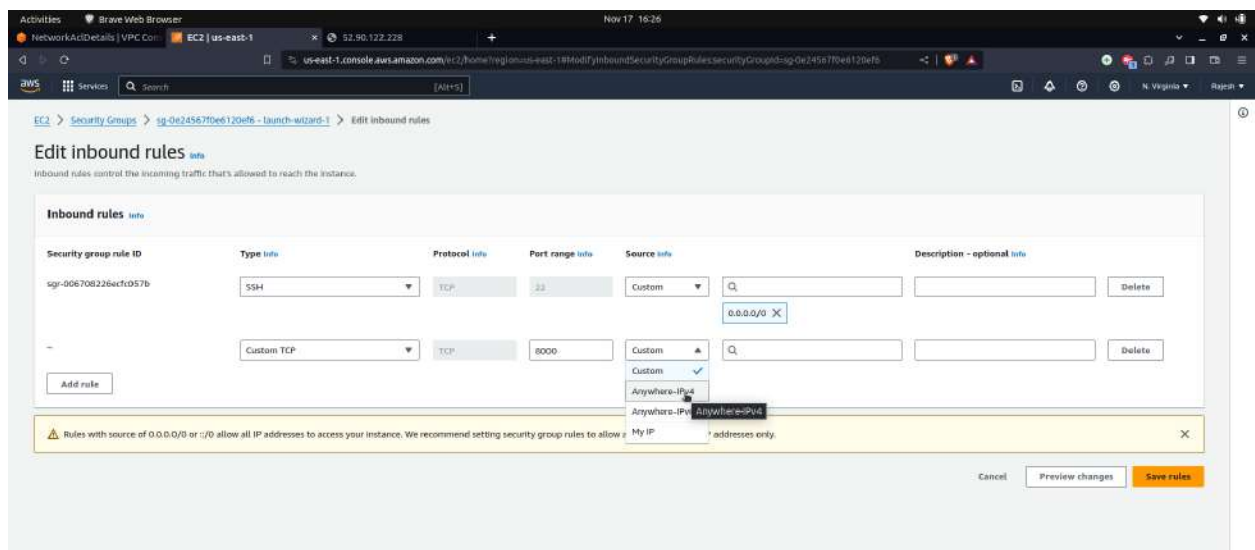
Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
sg-006708226ecf057b	SSH	IPv4	SSH	TCP	22	0.0.0.0/0	-

Step 46: Click on Add Rule

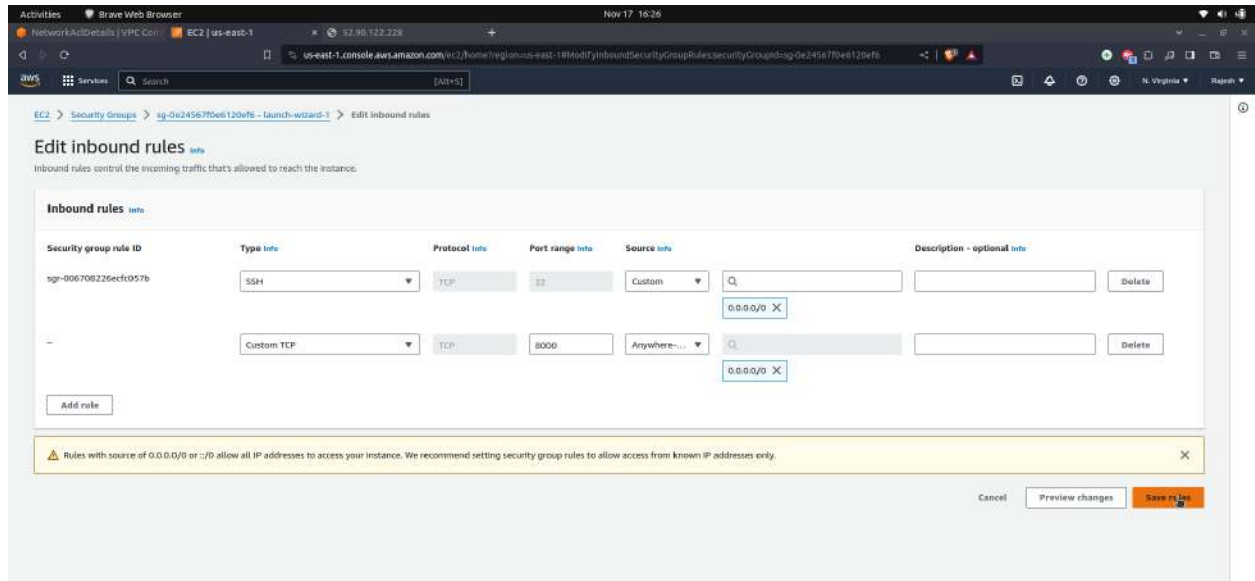


Step 47: Apply this changes

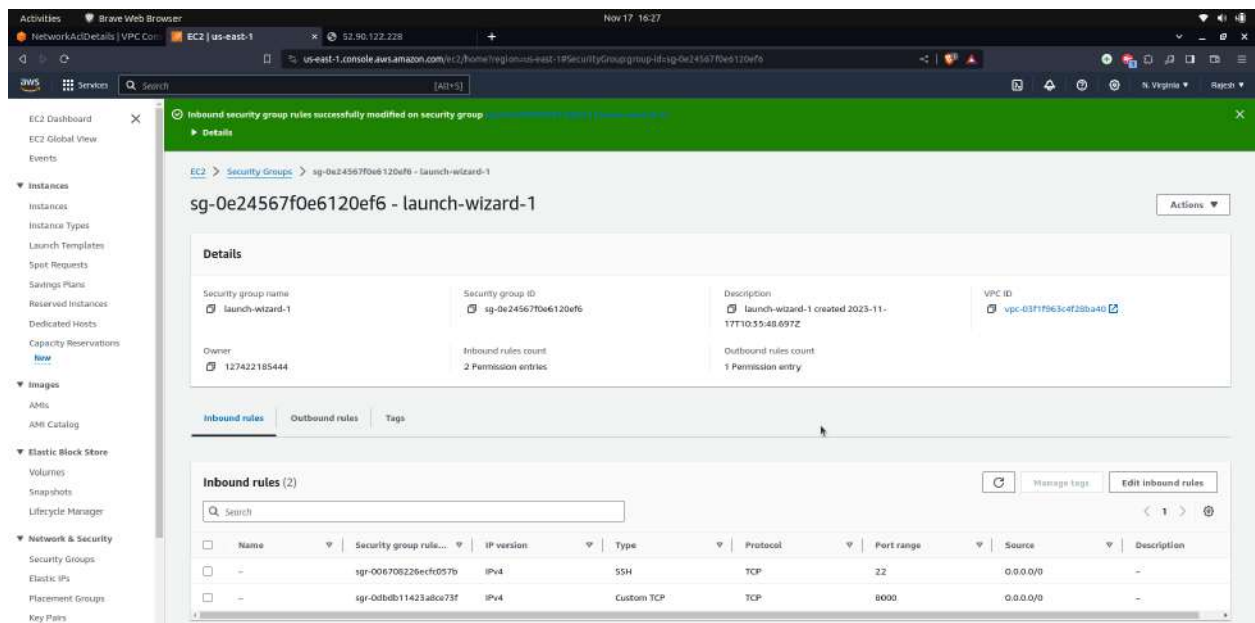
Choose **CustomTcp** as type mention **8000** in port range .In source select **anywhere ipv4**



Step 48: Click on save

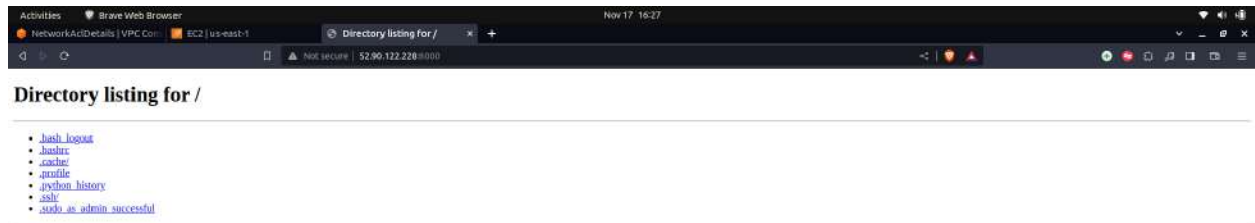


Step 49: successfully edited the inbound rules

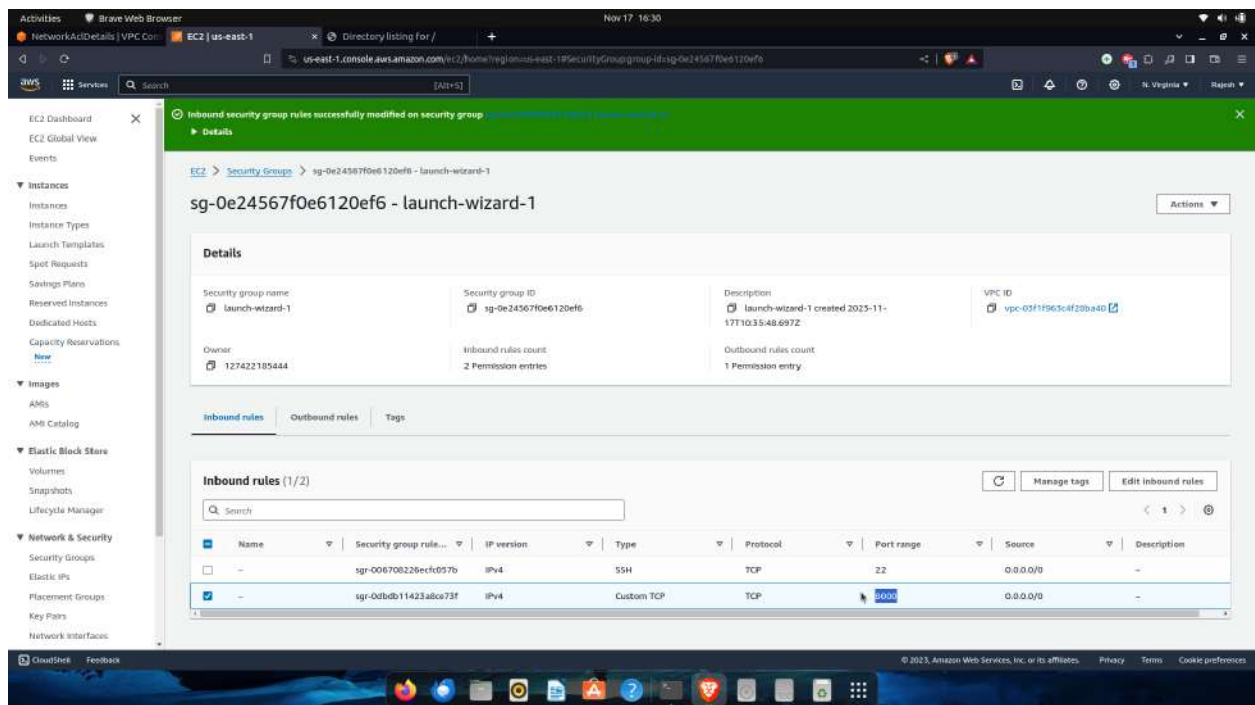


Step 50: Open your previous tab and refresh it

Now you may access your instance 

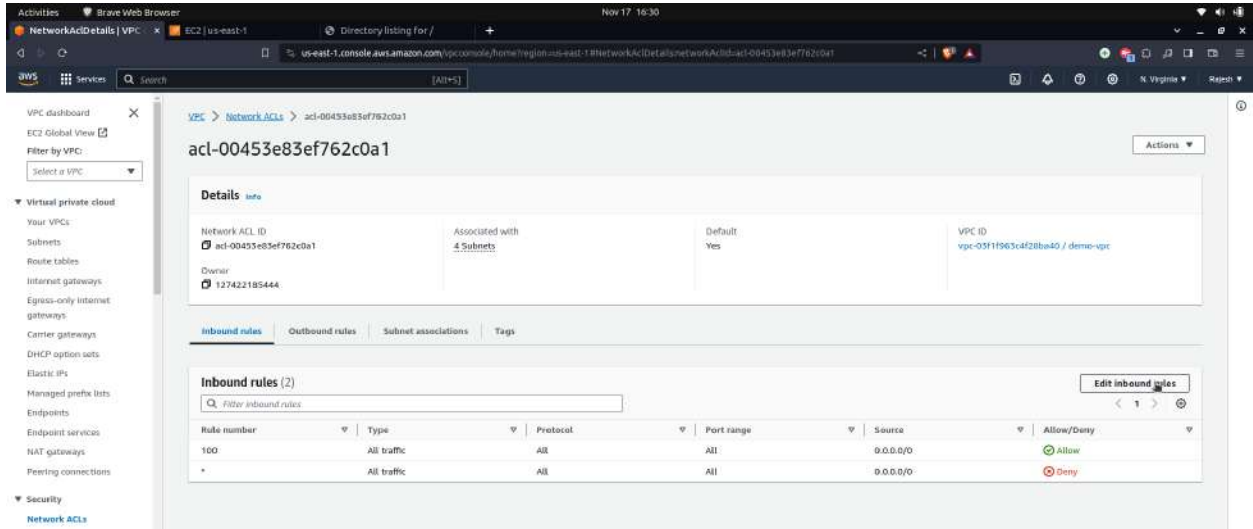


Step 51: Security group allowing port 8000



Step 52: Go to NACLs setting of your vpc

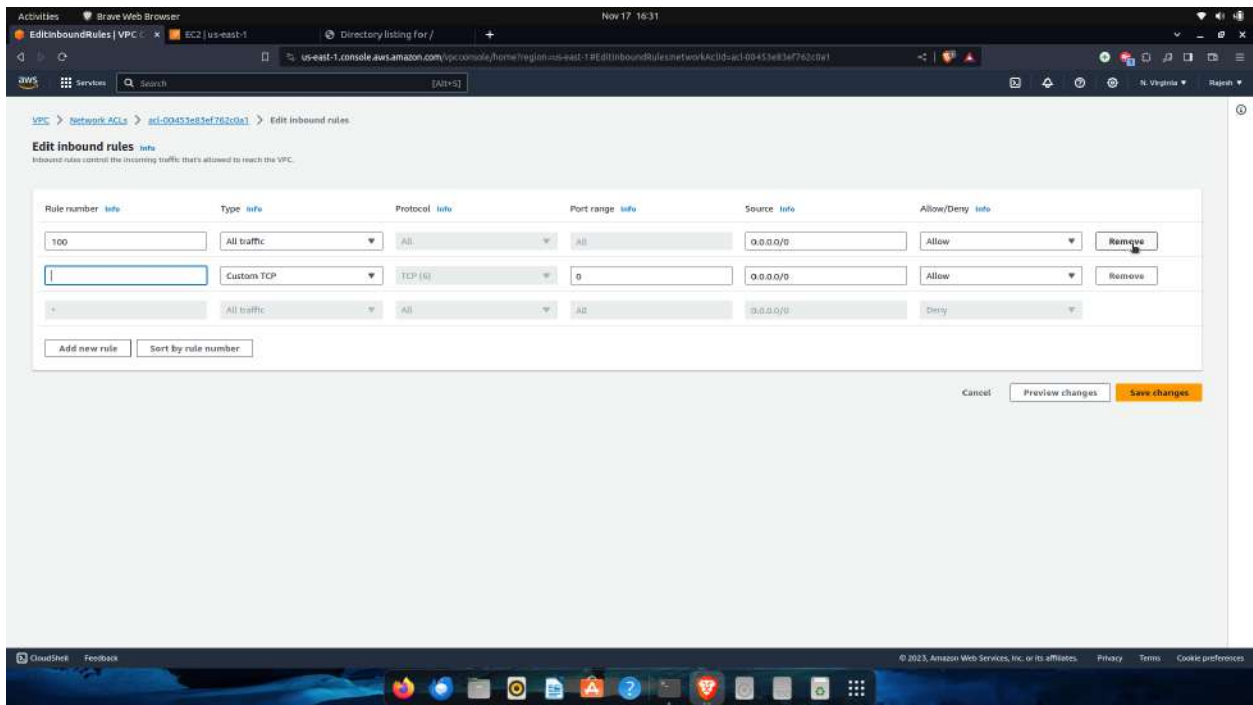
Click on **Edit inbound rules**



The screenshot shows the AWS Management Console interface for a Network ACL. The breadcrumb navigation is **VPC > Network ACLs > acl-00453e83ef762c0a1**. The page title is **acl-00453e83ef762c0a1**. The **Details** section shows the Network ACL ID, associated subnets (4), default status (Yes), and VPC ID (vpc-0911963ca428be40 / demo-vpc). The **Inbound rules** tab is active, displaying a table with 2 rules.

Rule number	Type	Protocol	Port range	Source	Allow/Deny
100	All traffic	All	All	0.0.0.0/0	Allow
*	All traffic	All	All	0.0.0.0/0	Deny

Step 53: Remove the 100 rule



The screenshot shows the **Edit inbound rules** page for the same Network ACL. The breadcrumb navigation is **VPC > Network ACLs > acl-00453e83ef762c0a1 > Edit inbound rules**. The page title is **Edit inbound rules**. The instructions state: "Inbound rules control the incoming traffic that's allowed to reach the VPC." The table shows three rules, with the first rule (number 100) highlighted. The **Remove** button is visible next to the first rule.

Rule number	Type	Protocol	Port range	Source	Allow/Deny
100	All traffic	All	All	0.0.0.0/0	Allow
	Custom TCP	TCP (80)	80	0.0.0.0/0	Allow
*	All traffic	All	All	0.0.0.0/0	Deny

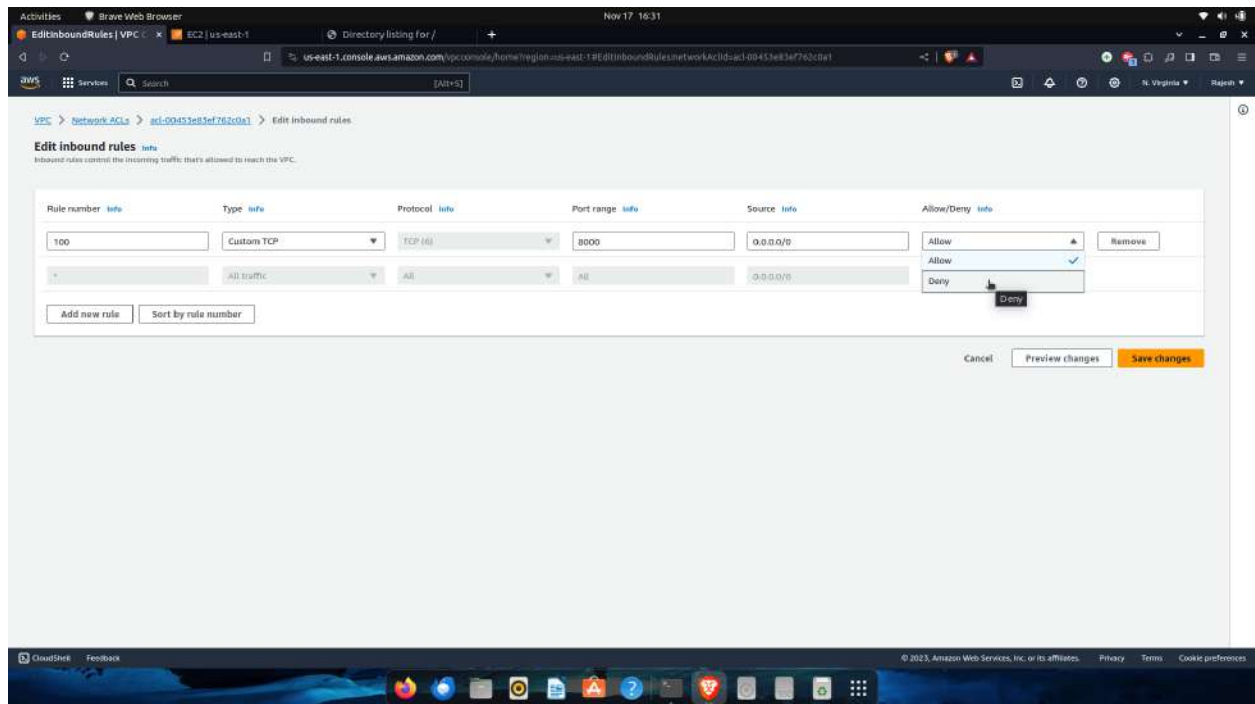
Step 54: Add new 100 rule

Rule number: 100,

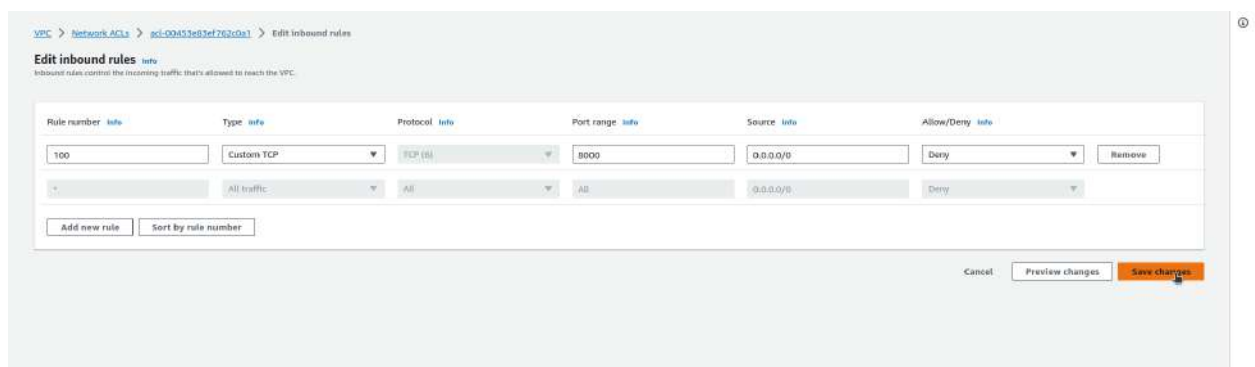
Type: Custom TCP

Port range: 8000

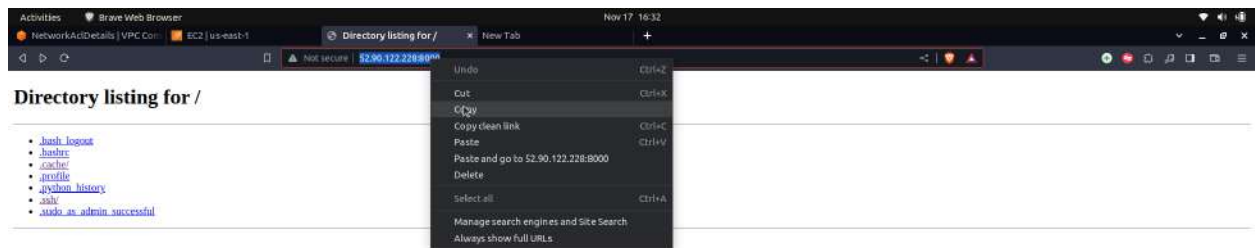
Choose : Deny



Step 55: Save Changes

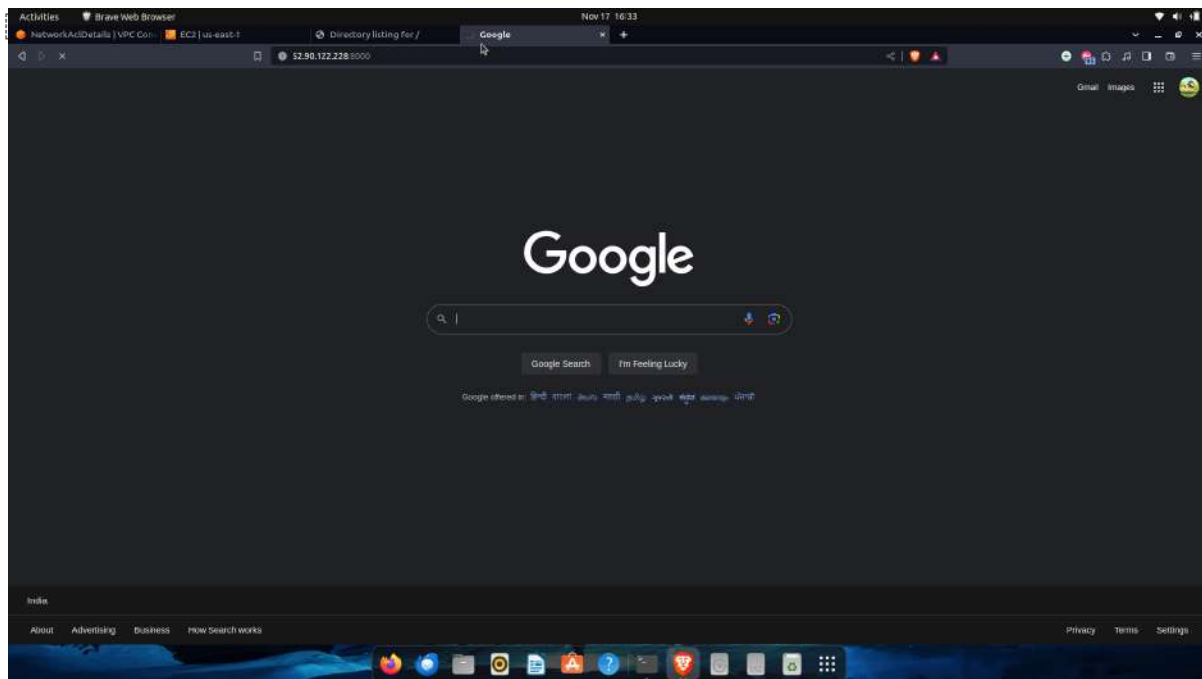


Step 56: Now copy the url of our instance



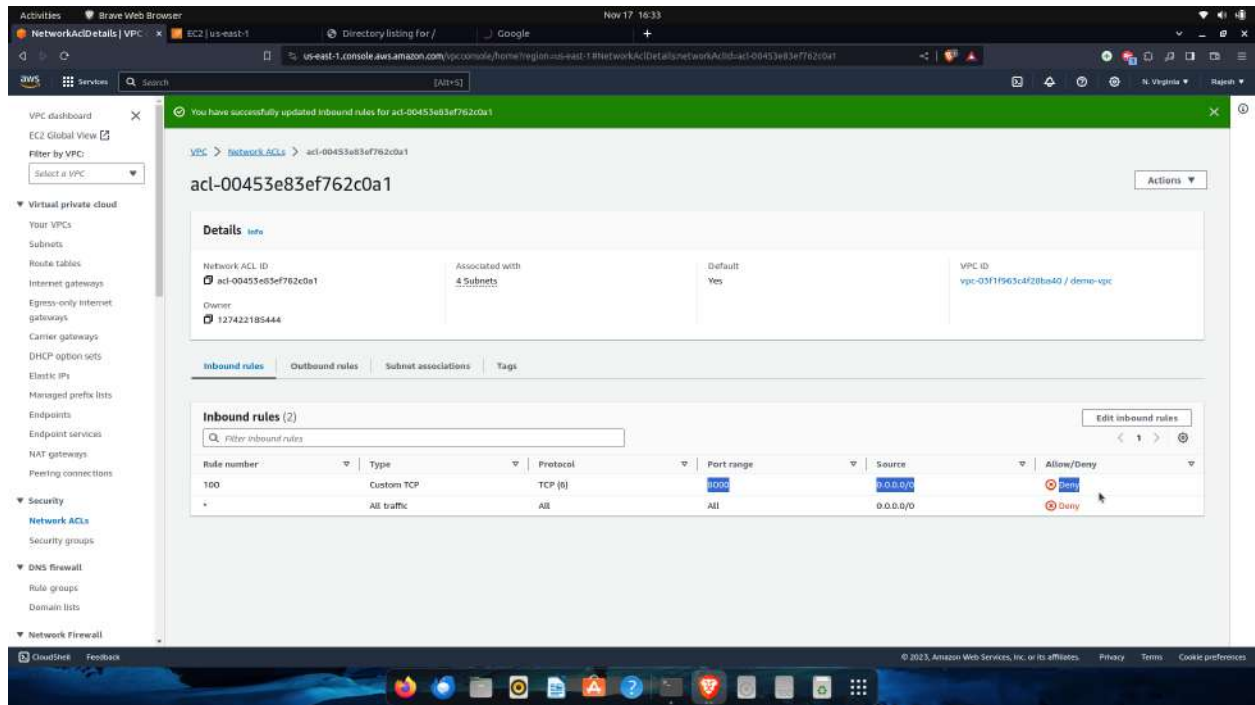
Step 57: paste it a new tab or browser

You don't get any response from our instance.

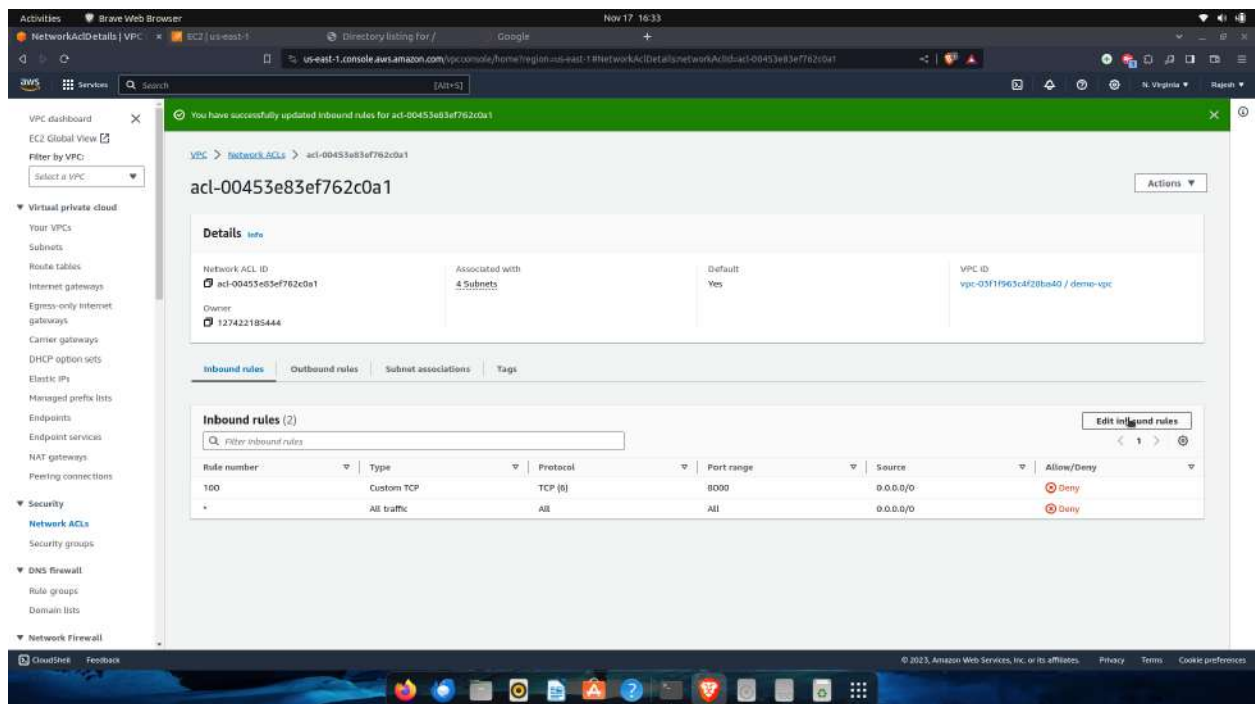


Step 58: Reason

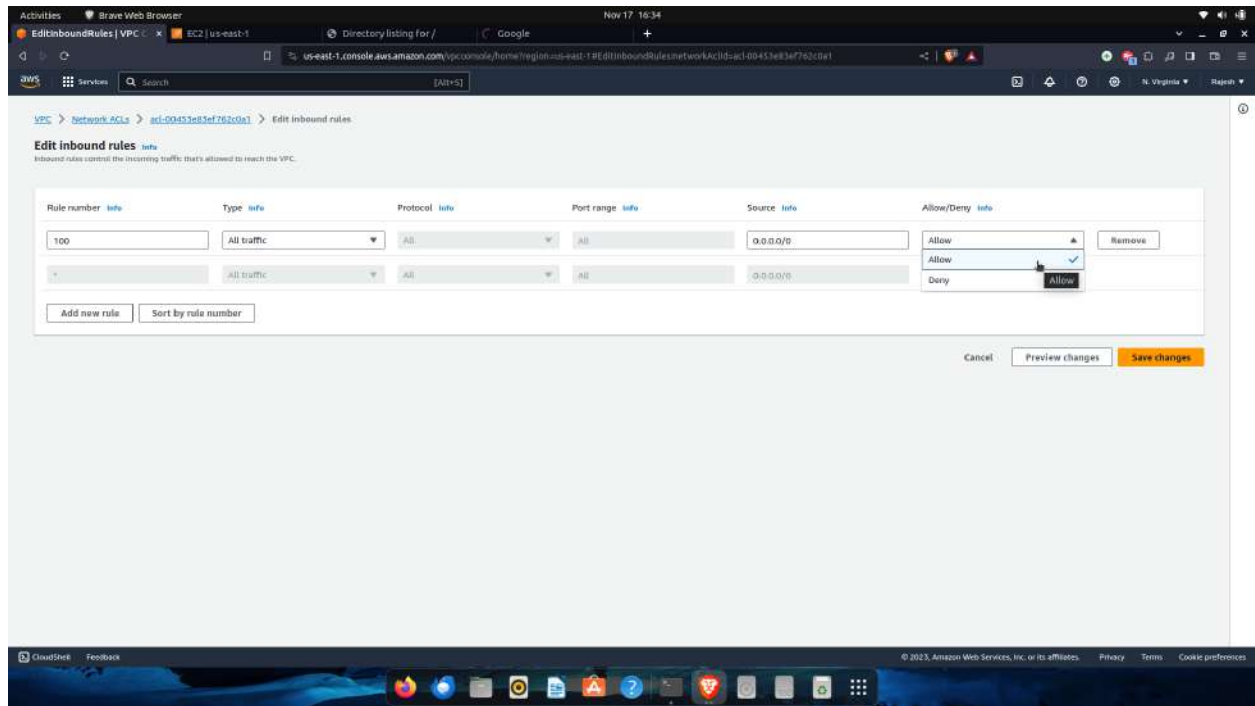
because ,we deny the traffic from port 8000 at NACLs level.



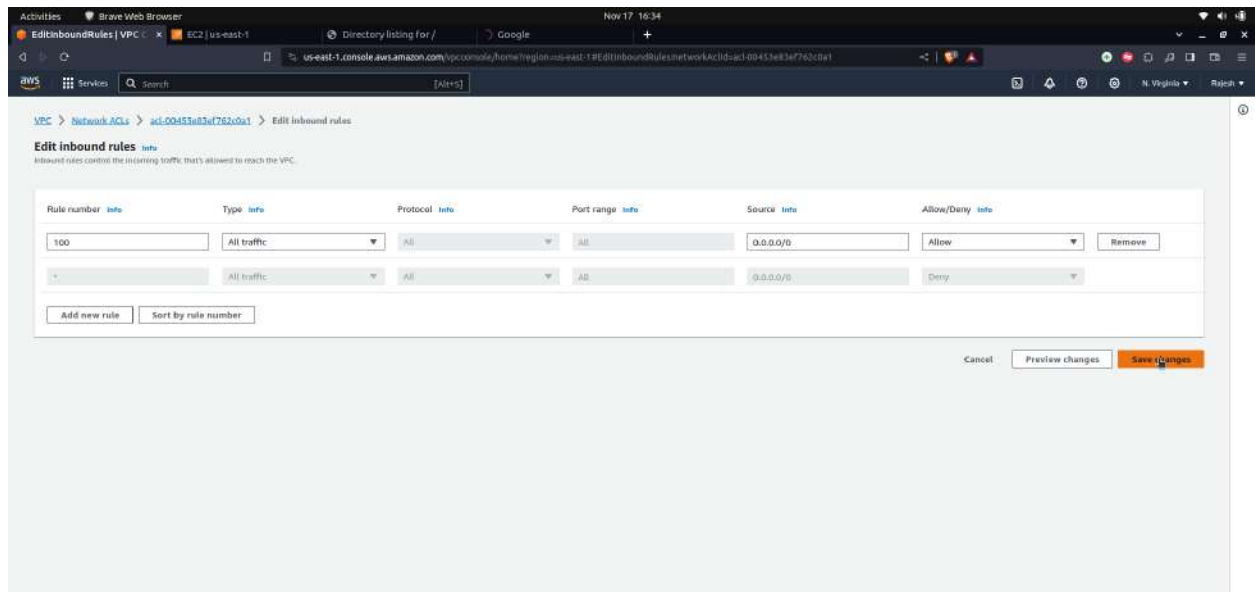
Step 59: click on edit inbound rules



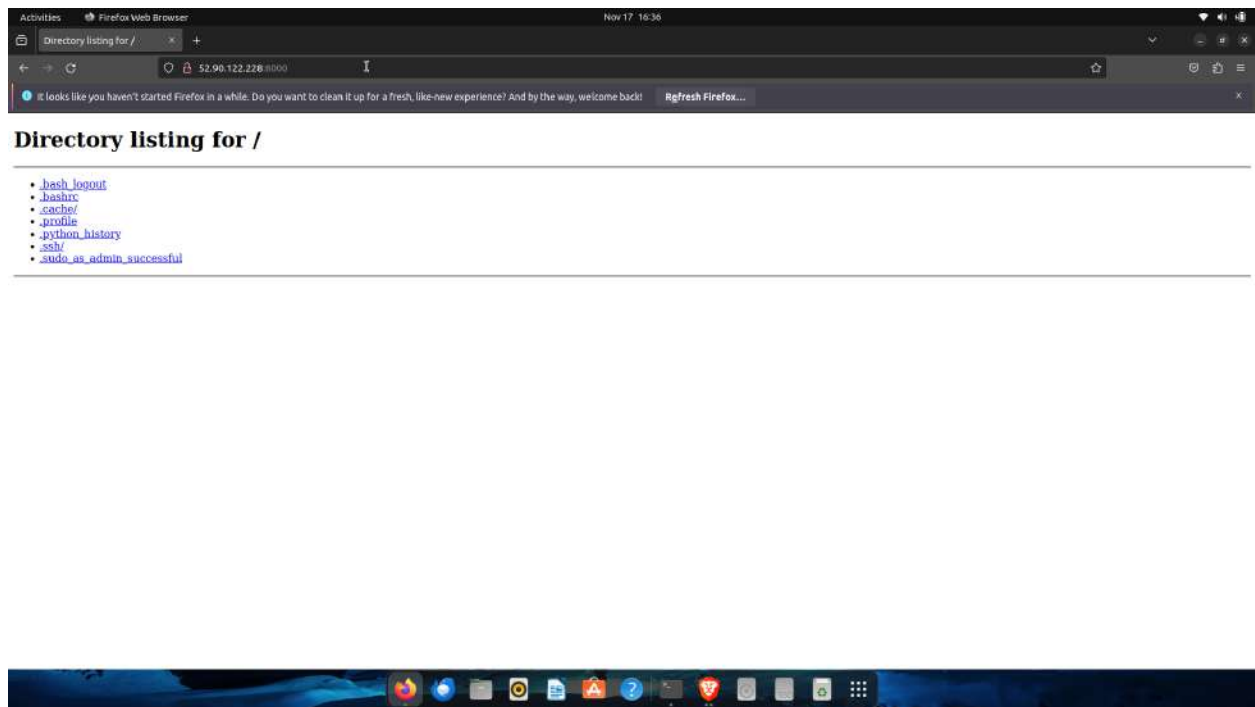
Step 60: Now allow all traffic



Step 61: Save changes



Step 62: Now server is getting response to port 8000



This is all about a small practice project on Security Groups and NACLs .I hope you like this project 🙏.Thank you

by

[Rajesh's blog](#)