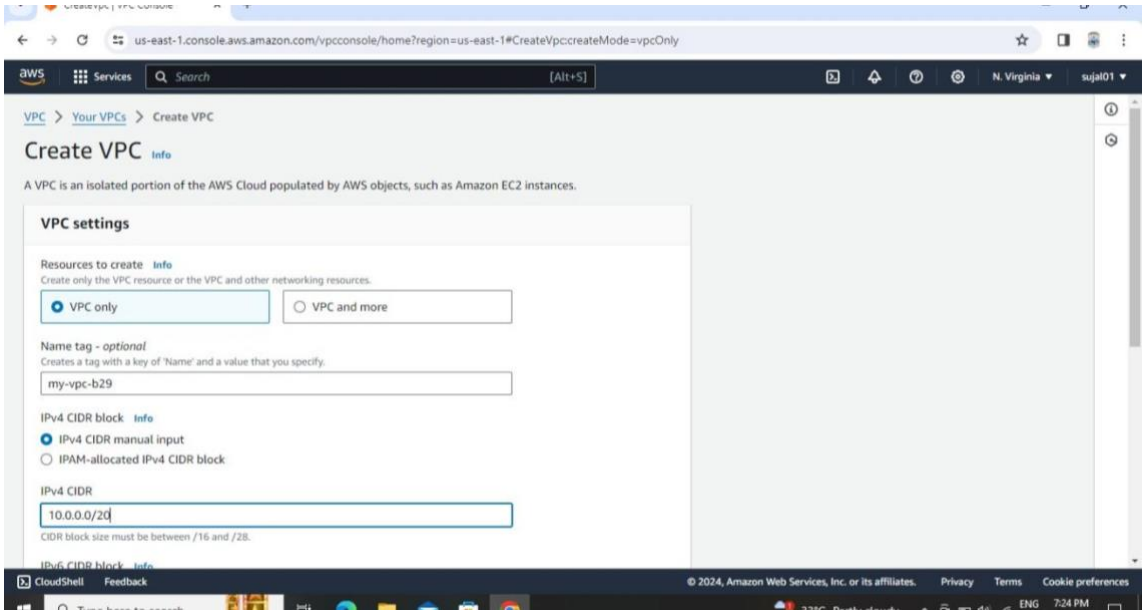
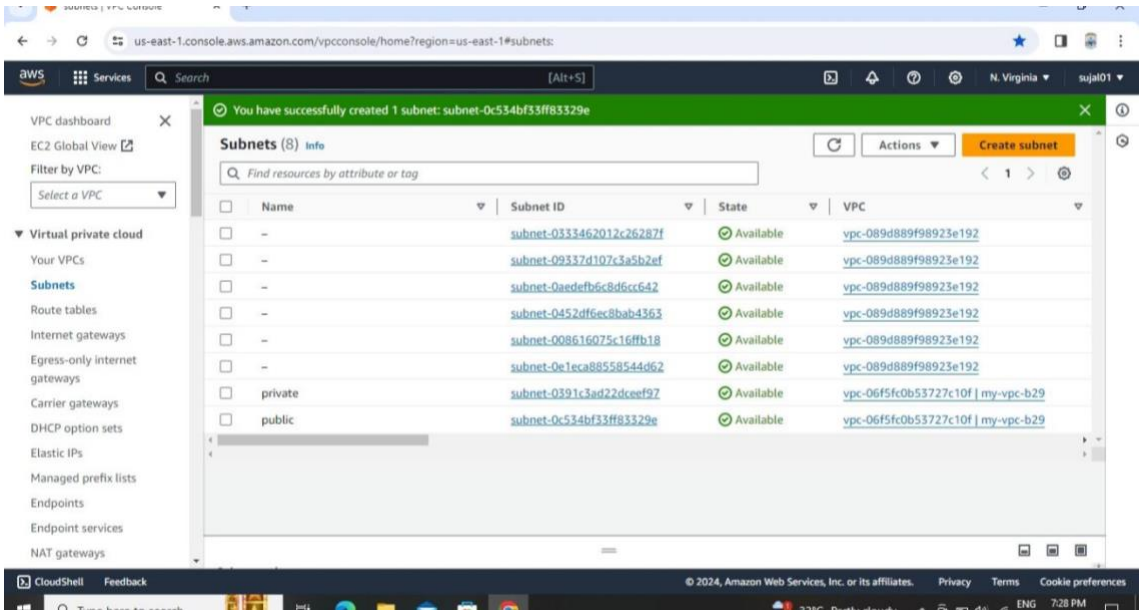
AWS Project

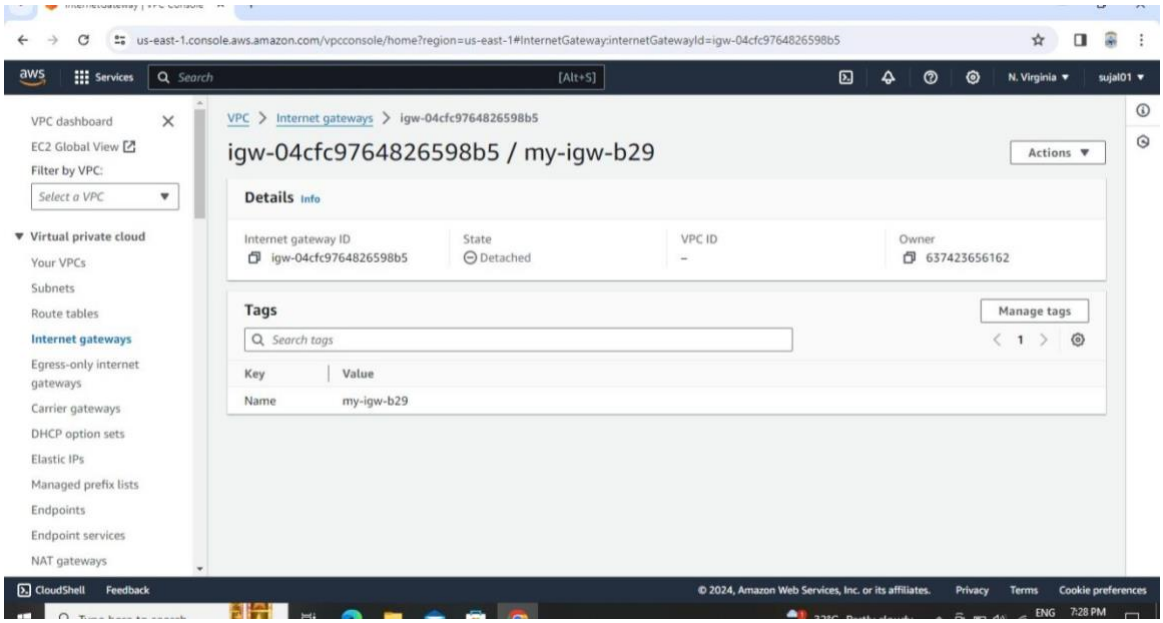
1) Login to aws using registered user credentials. Go to the VPC dashboard and create a new VPC. We have used ’10.0.0.0/20’ as IPv4 CIDR, while creating VPC.



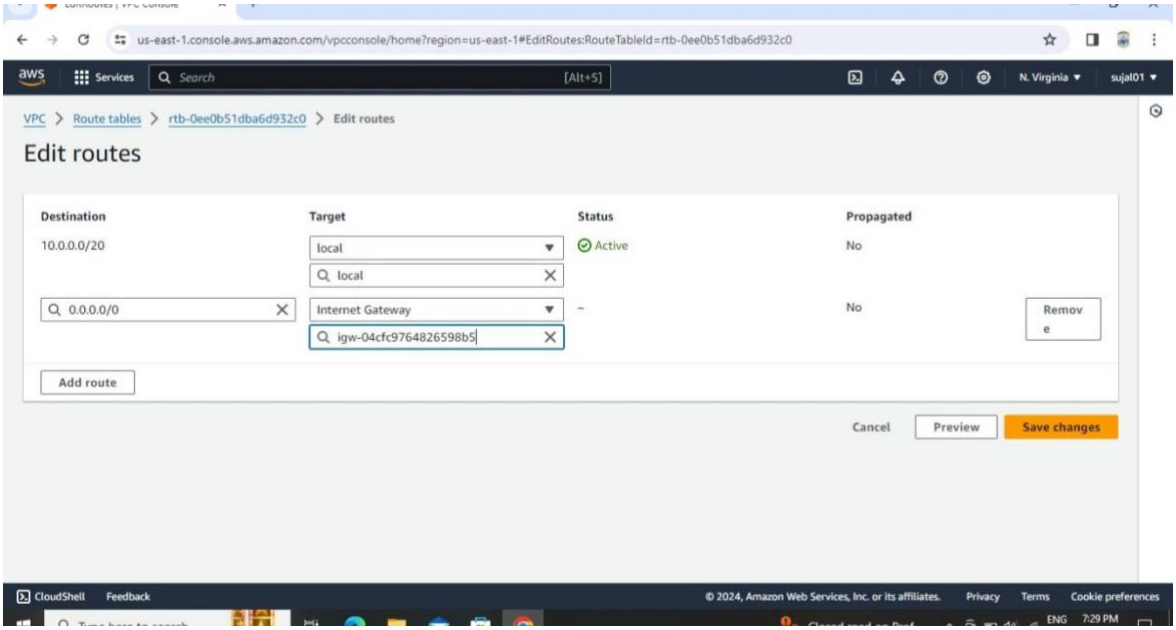
2) Go to subnets dashboard and create 2 new subnets of the new VPC. Private subnet has IPv4 CIDR block ‘10.0.1.0/25’ configuration and public subnet has IPv4 CIDR block ‘10.0.2.0/25’ configuration. Create both subnets in different availability zones.



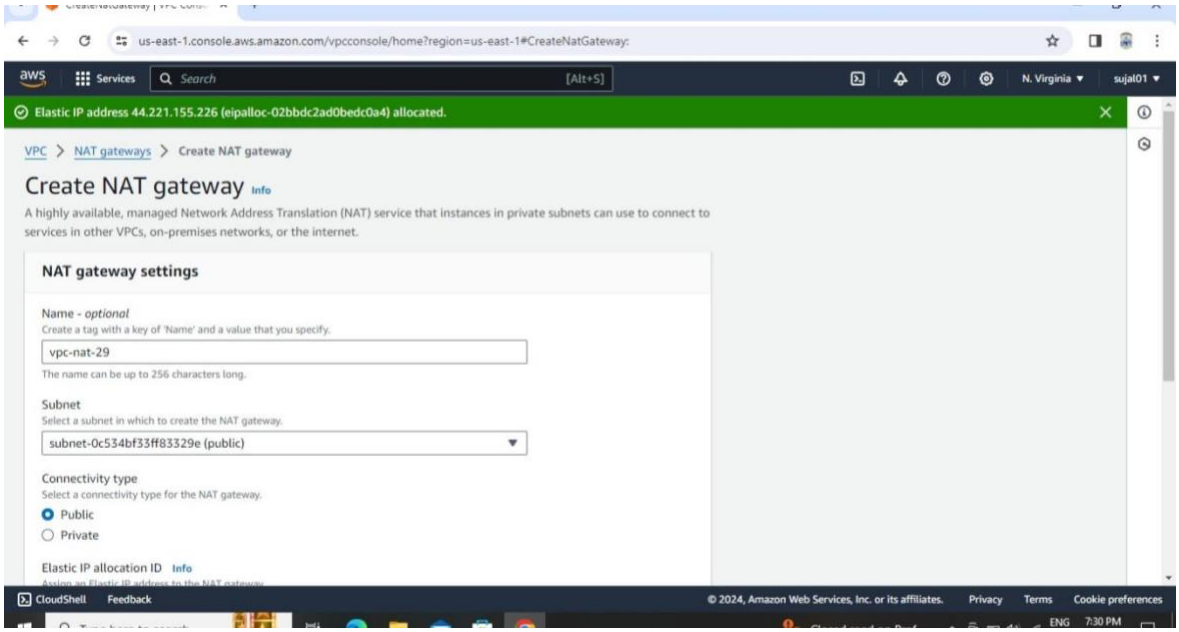
3) Go to internet gateways and create a new internet gateway. Click on actions and attach this internet gateway to our new VPC.



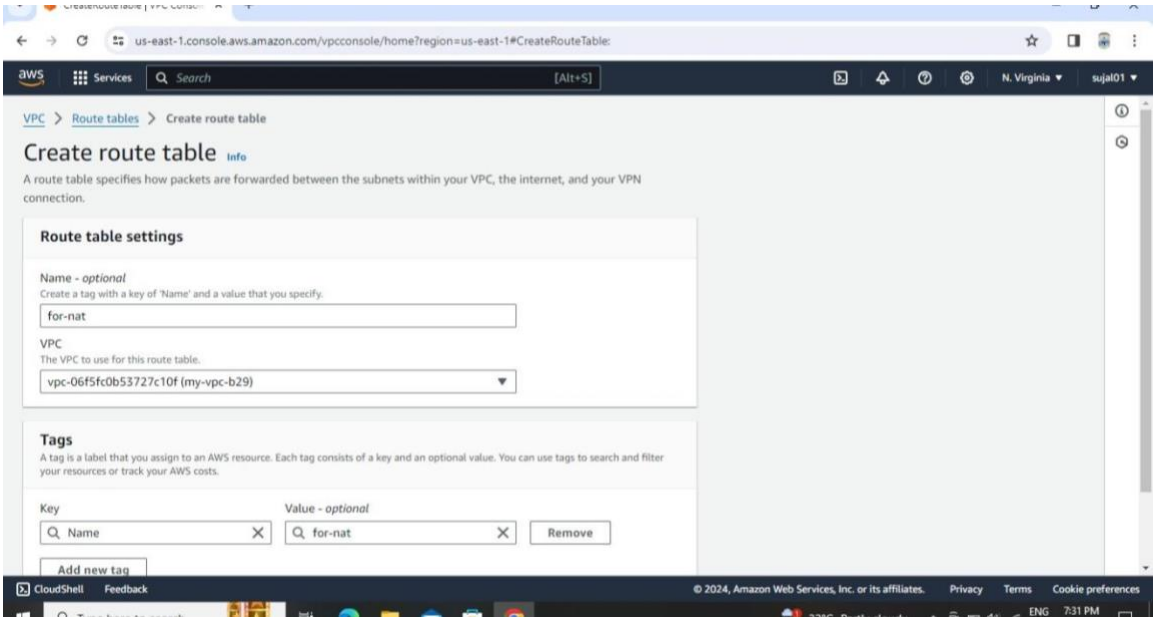
4) Go to routing tables dashboard and open the new routing table that was created by default when new VPC was created. Click on edit routes and add the new internet gateway to the route table.



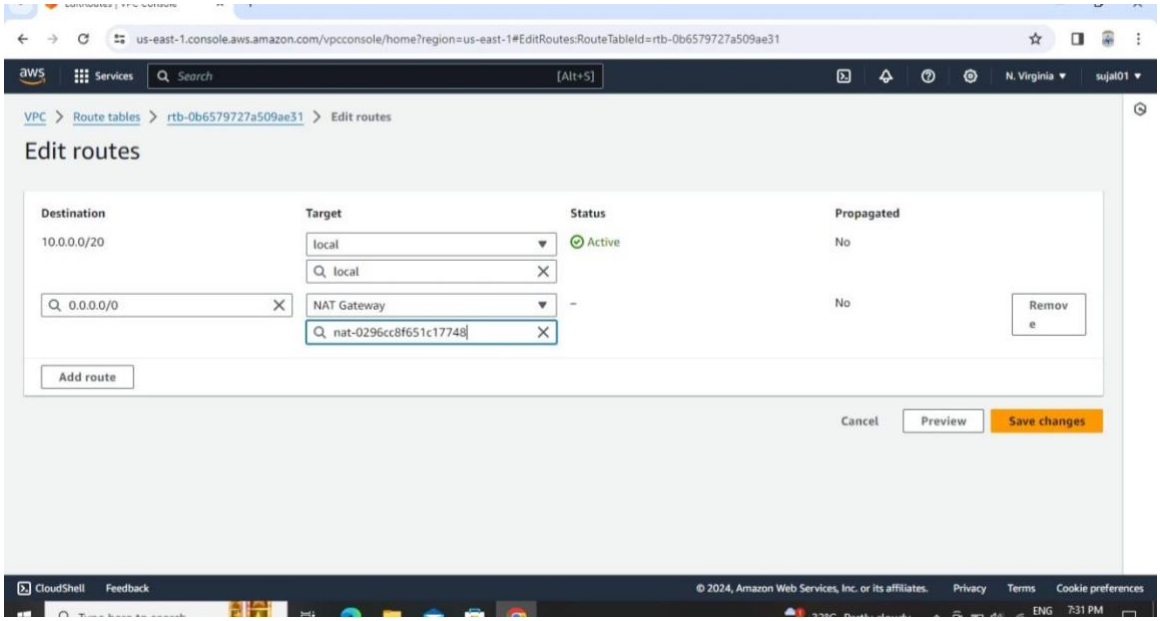
5) Go to NAT gateway dashboard and create a NAT gateway. While creating the gateway choose public subnet from subnet drop-down.



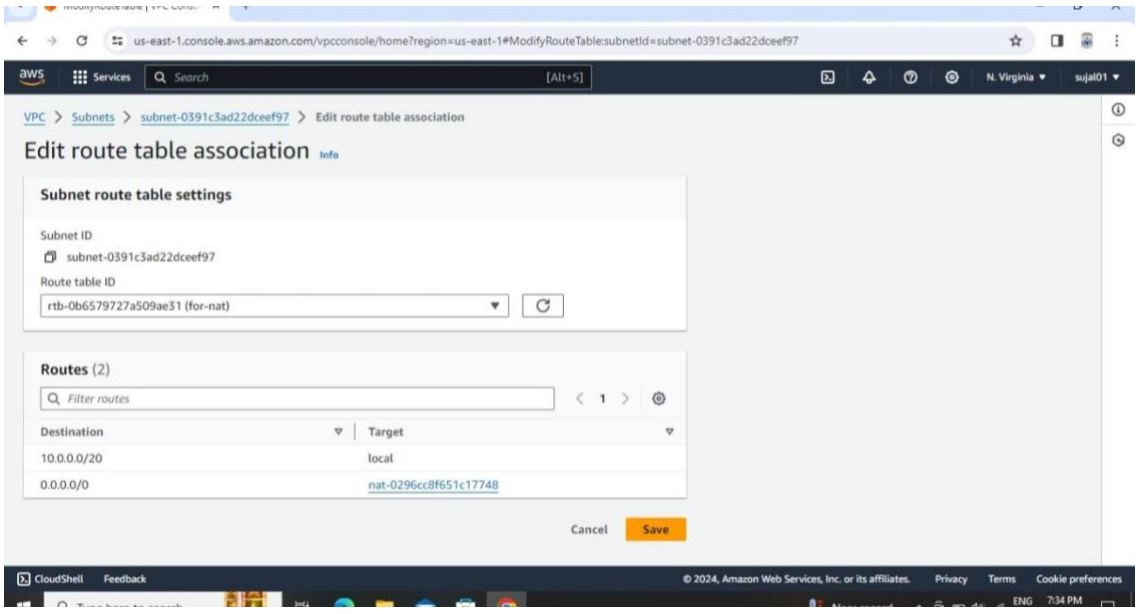
6) Go to route tables dashboard and create a new route table (for-nat). Click on edit routes.



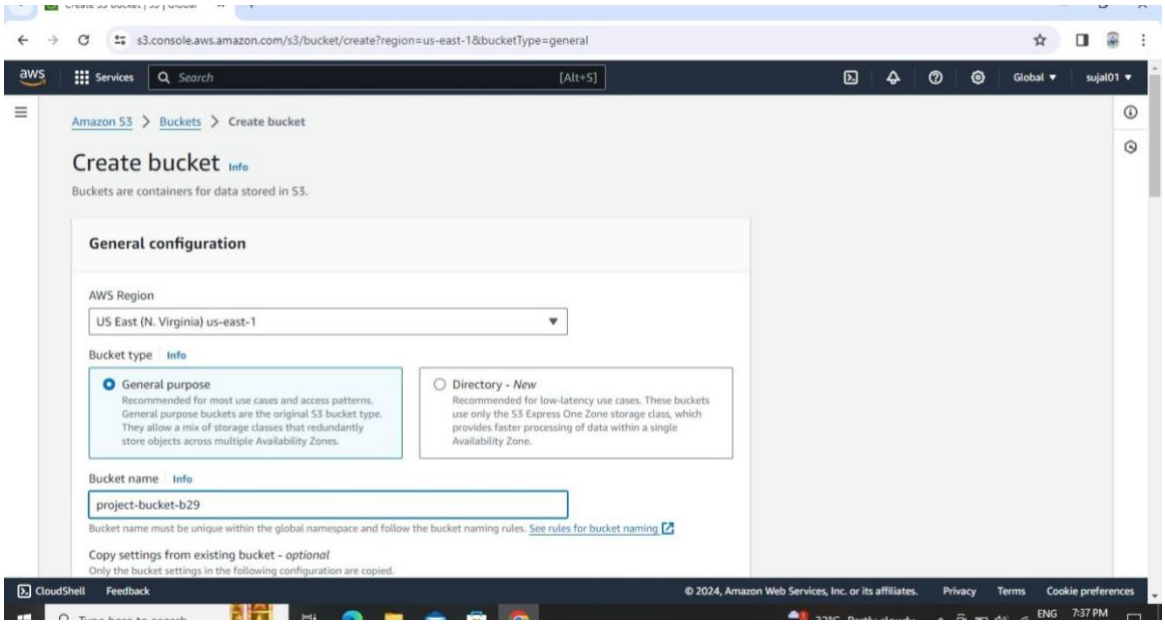
7) Add new NAT Gateway route and click save changes.



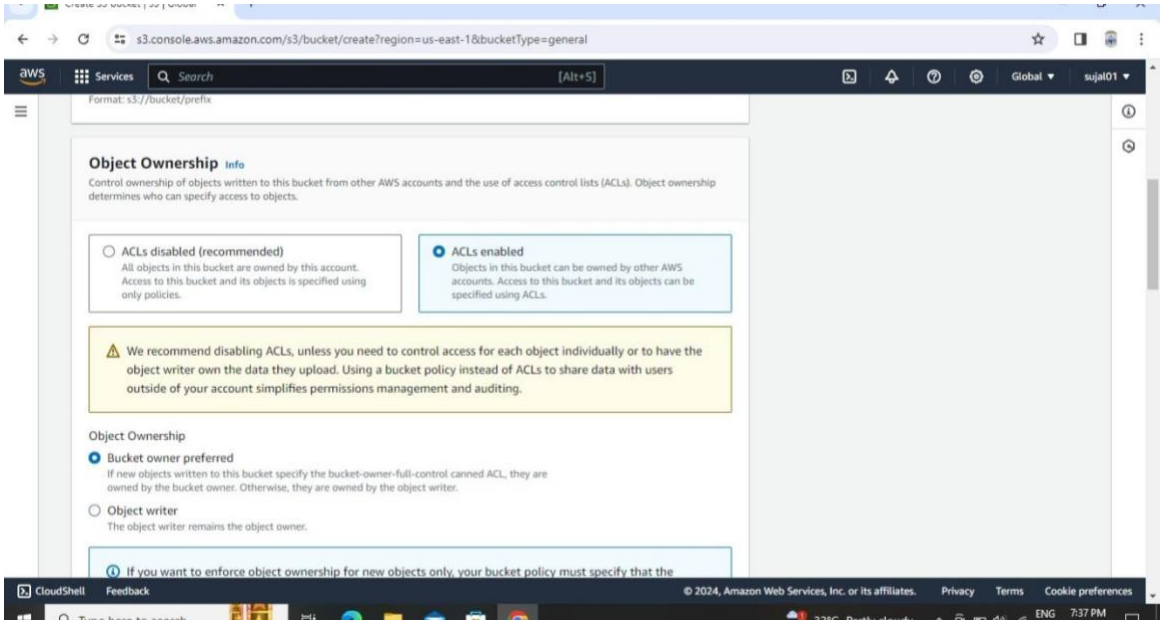
8) Go to subnets dashboard and open private subnets. Under route tables click on edit subnet association and change the route table to for-nat. Click on save changes.



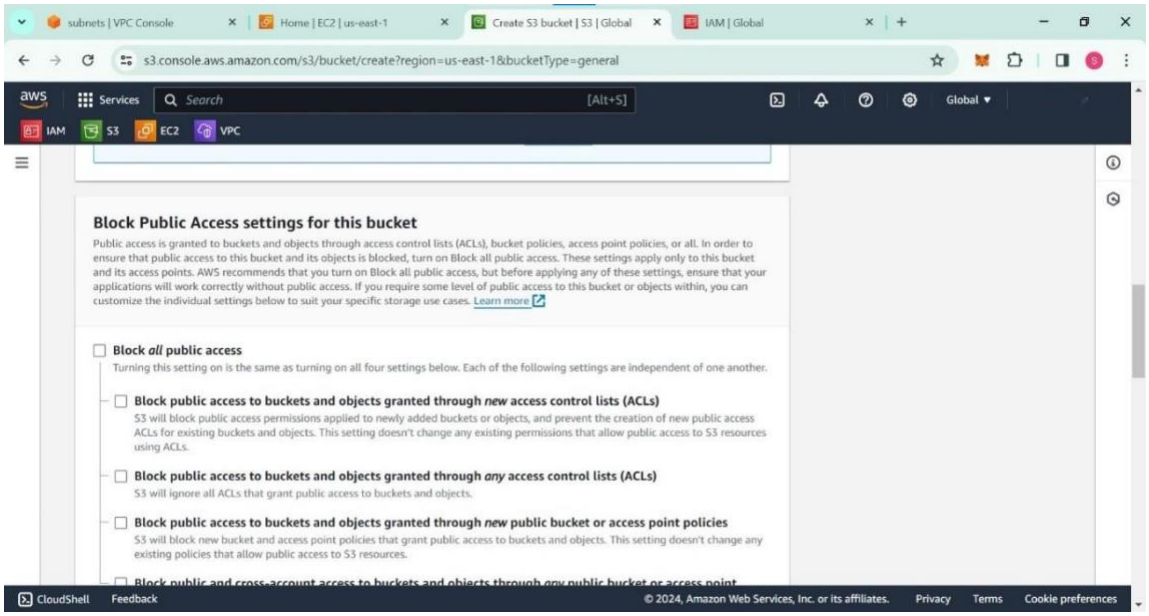
9) Go to S3 service dashboard and click on create bucket. Enter a unique bucknet name.



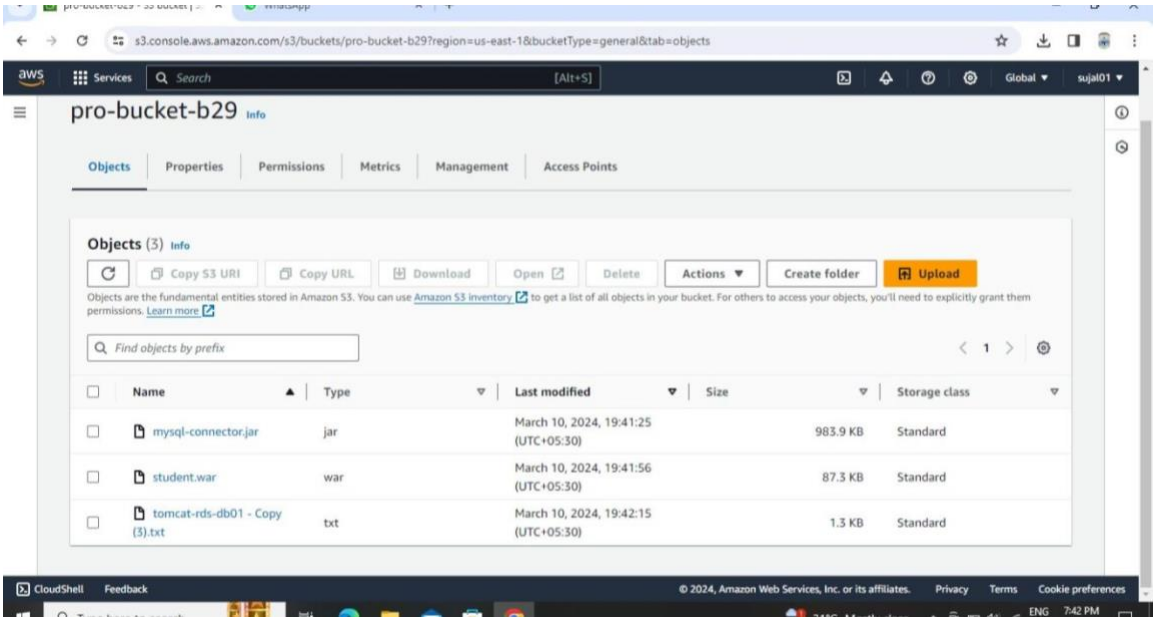
10) Choose ACLs enabled option.



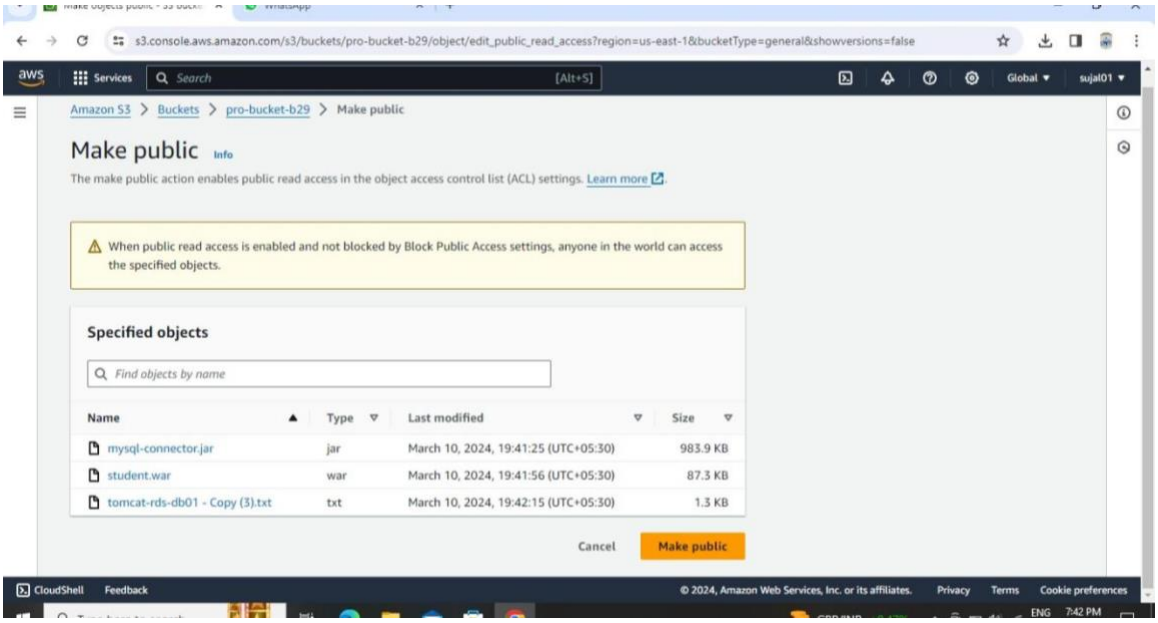
11) Untick the block all public access box. Click on create bucket.



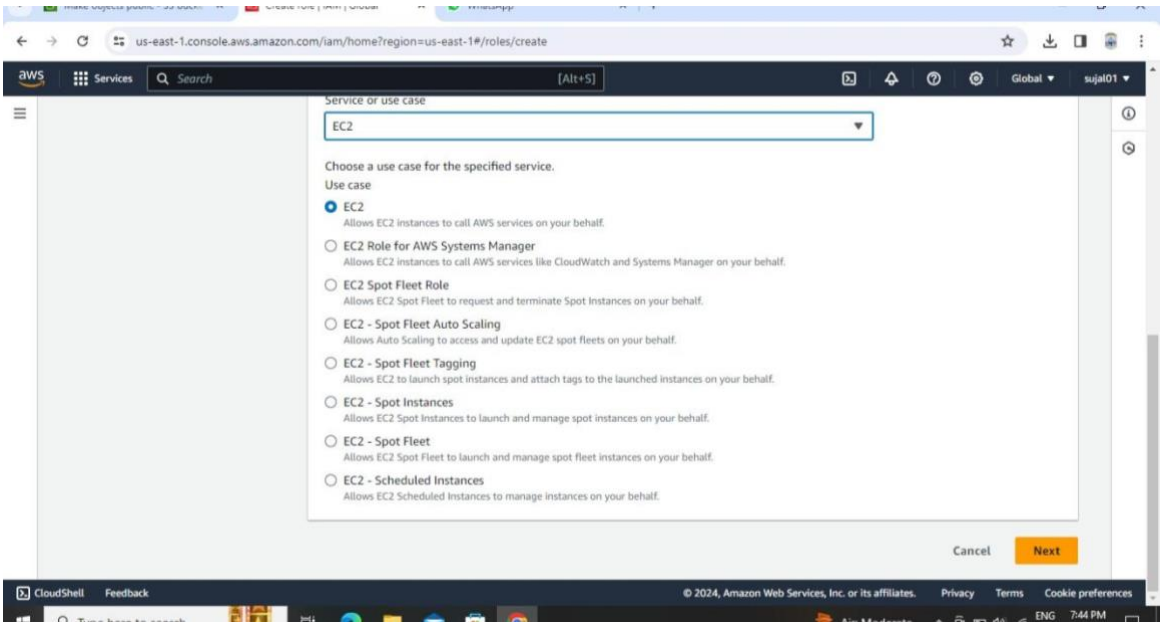
12) Upload the required configuration files in the bucket.



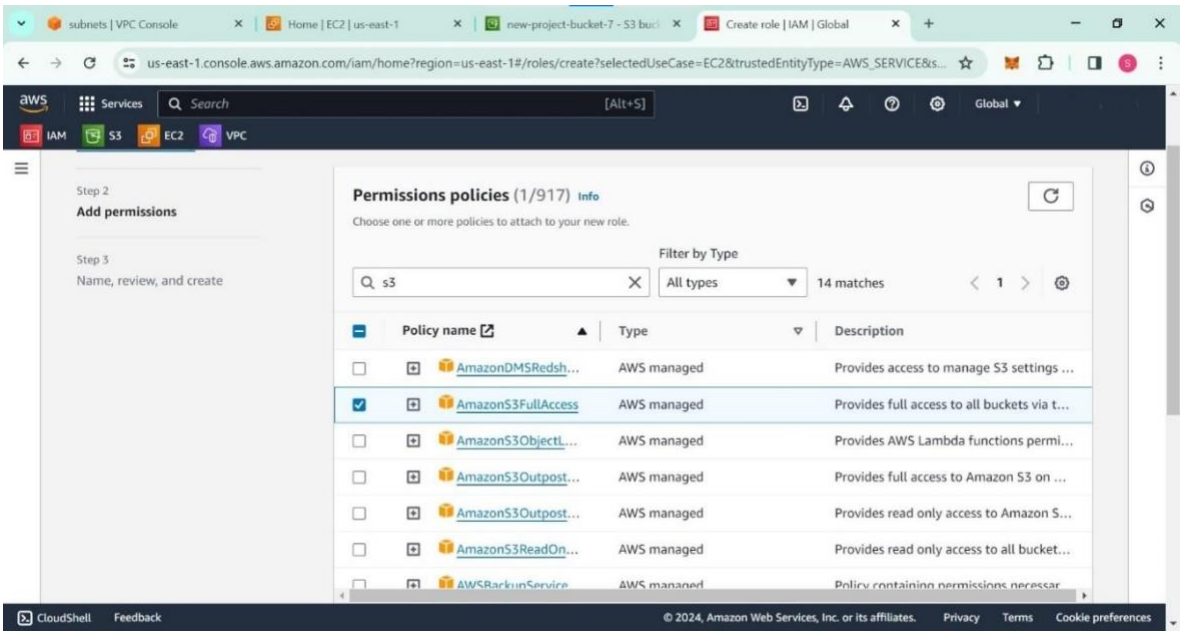
13) Select all uploaded objects in the bucket and click on actions, then click on make public using ACL.



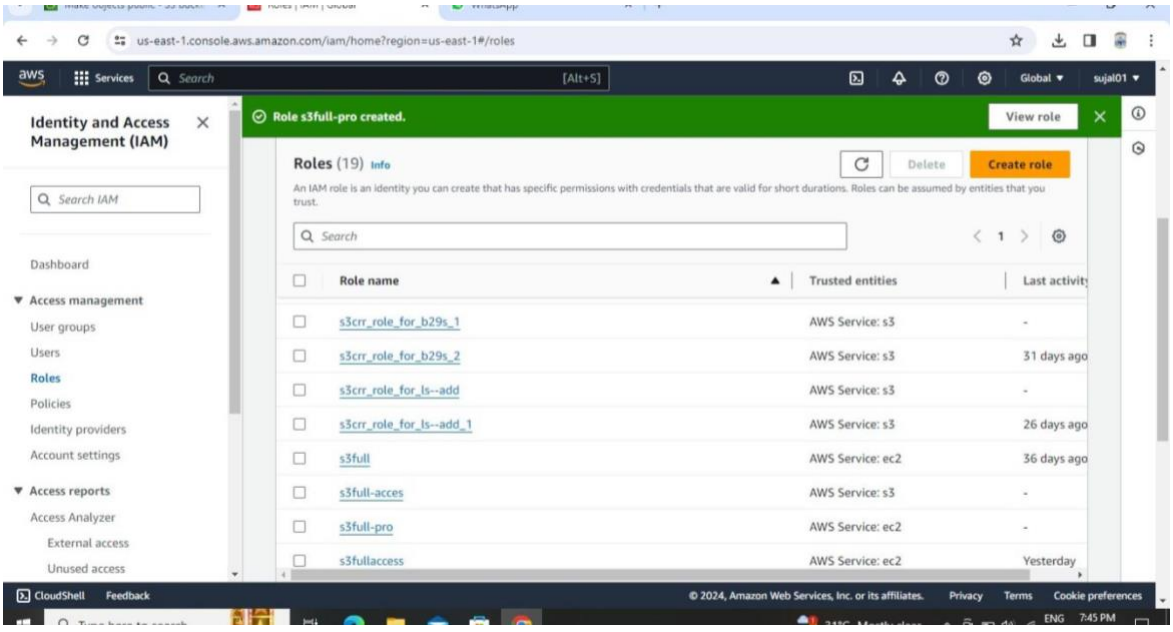
14) Now open the IAM service dashboard and click on create role. Choose the EC2 service under use case and click on next.



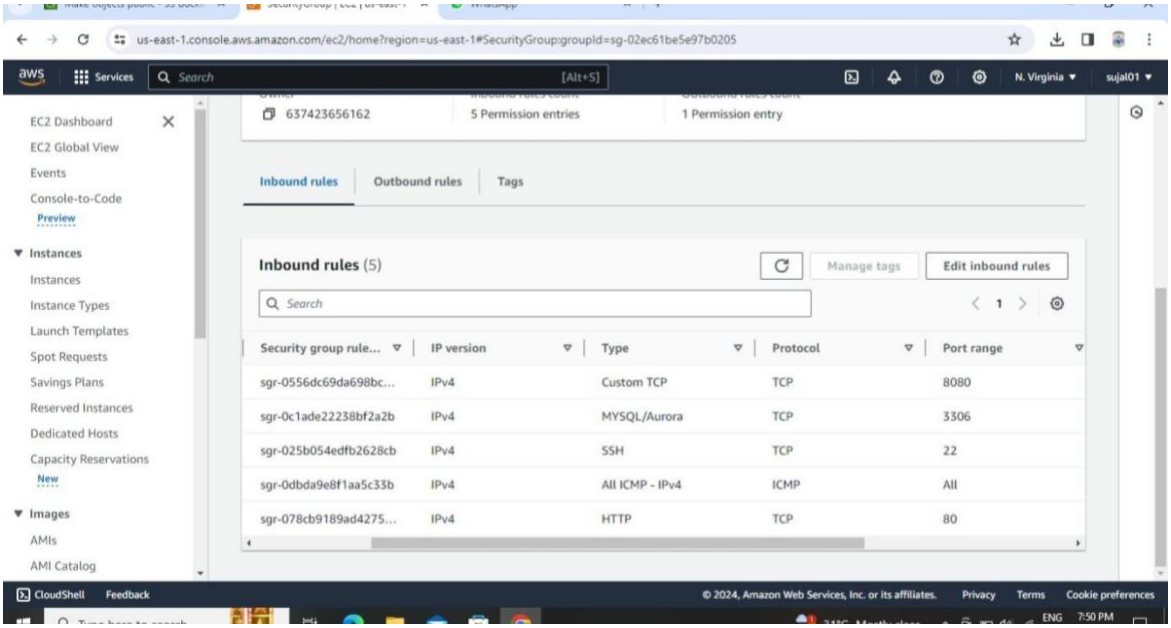
15) In the permissions policy section choose the ‘AmazonS3FullAccess’ and click on next. Enter a valid role name and click on create role.



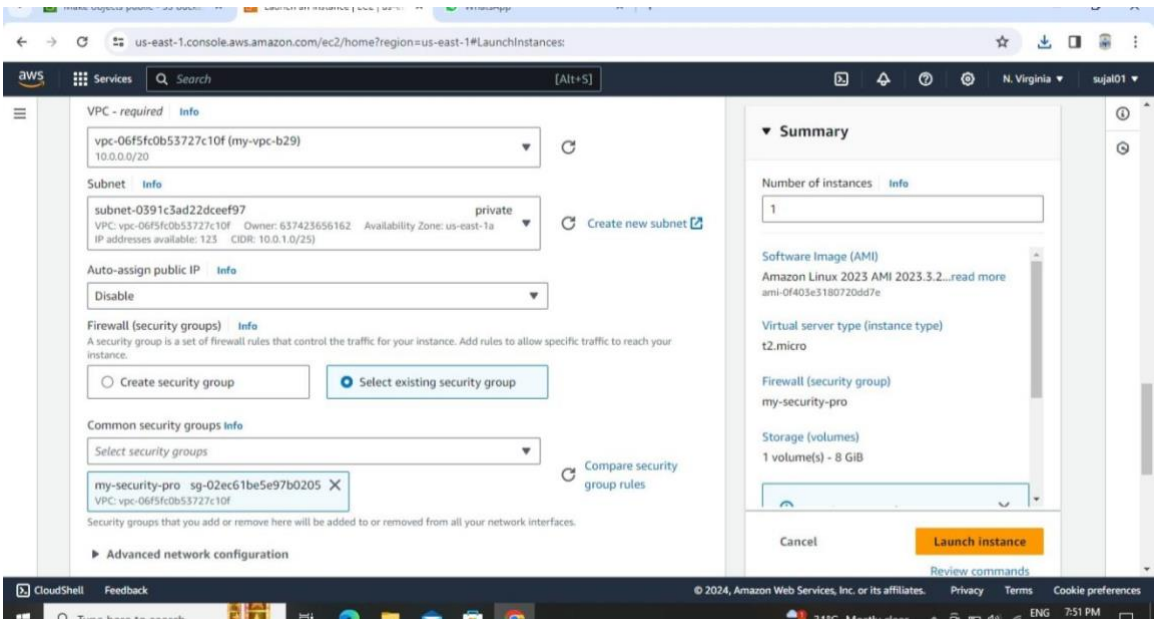
16) ‘project-role’ has been successfully created.



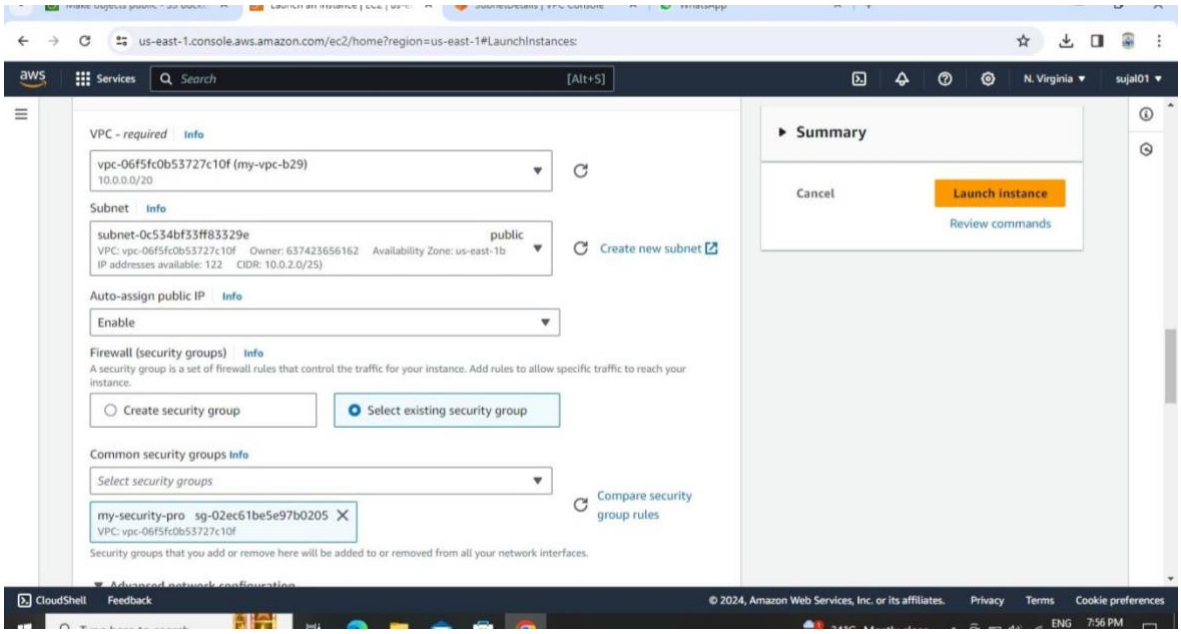
17) Go to EC2 dashboard and from left navigation bar select Security group. Create a security group with a valid name (here launch-wizard-7) and add all the following rules as inbound rules: SSH, HTTP, MYSQL/Aurora, ALL ICMP and Custom TCP with port range as 8080. Click on create security group.



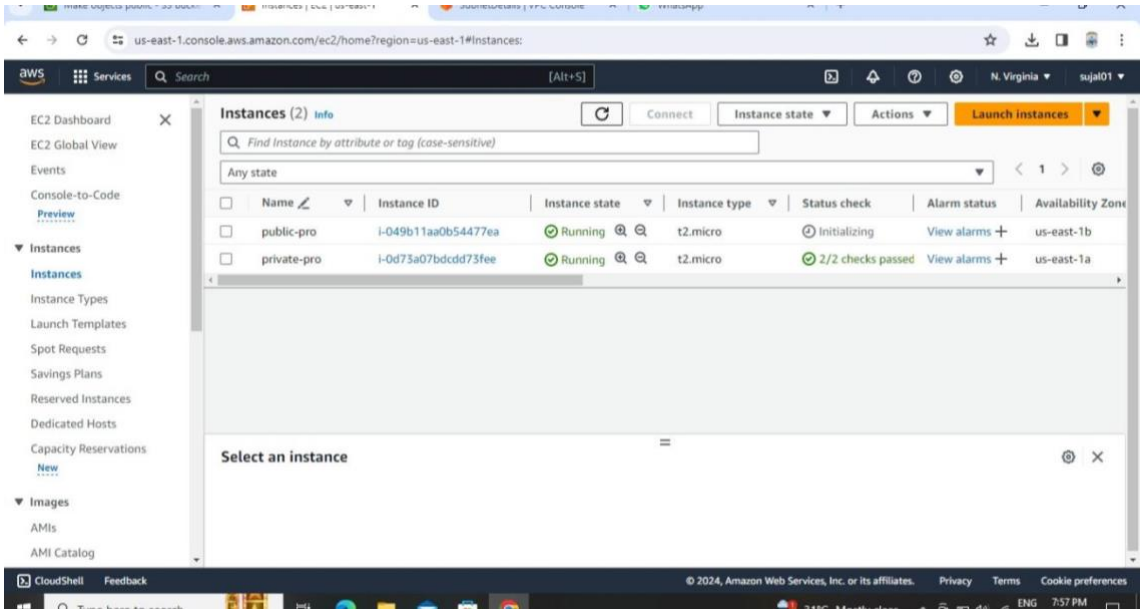
18) Go to EC2 instance dashboard and click on launch instance. Enter a valid instance name (project_private) and select a key-pair. Click edit in network settings and choose the newly created VPC. For private instance select private subnet. Add the existing security group 'launch-wizard-7' as security group. Click on launch instance.



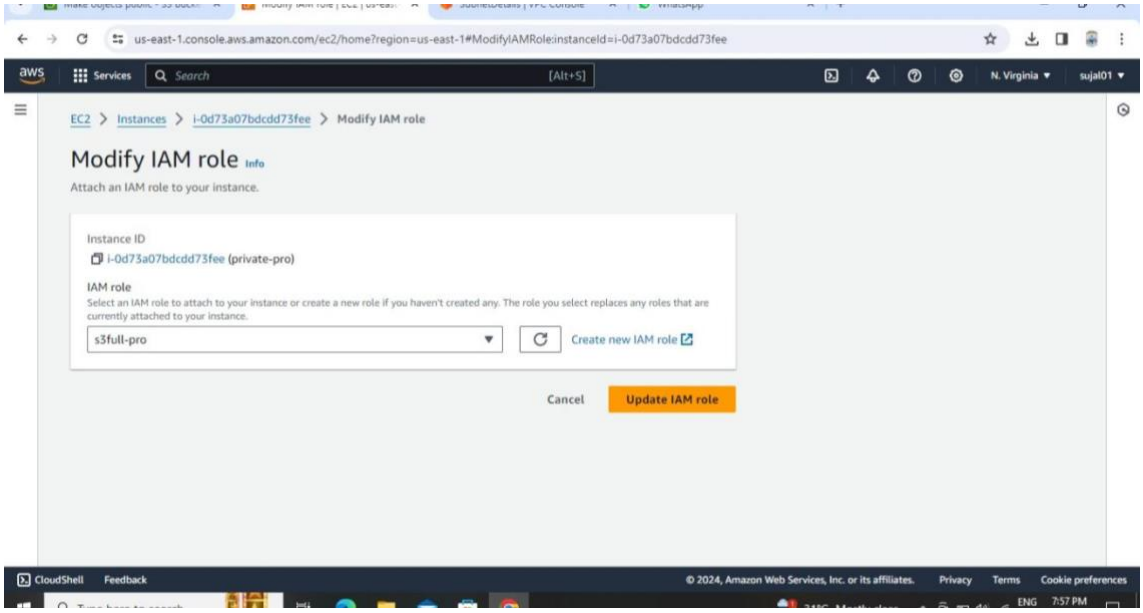
19) Again click on launch instance to create a second instance. Enter a valid instance name (project_public) and select a key-pair. Click edit in network settings and choose the newly created VPC. For public instance select public subnet. Add SSH and ICMP rules in security group and click on launch instance. Add the existing security group ‘launch-wizard-7’ as security group. Click on launch instance.



20) The two instances have been successfully created.



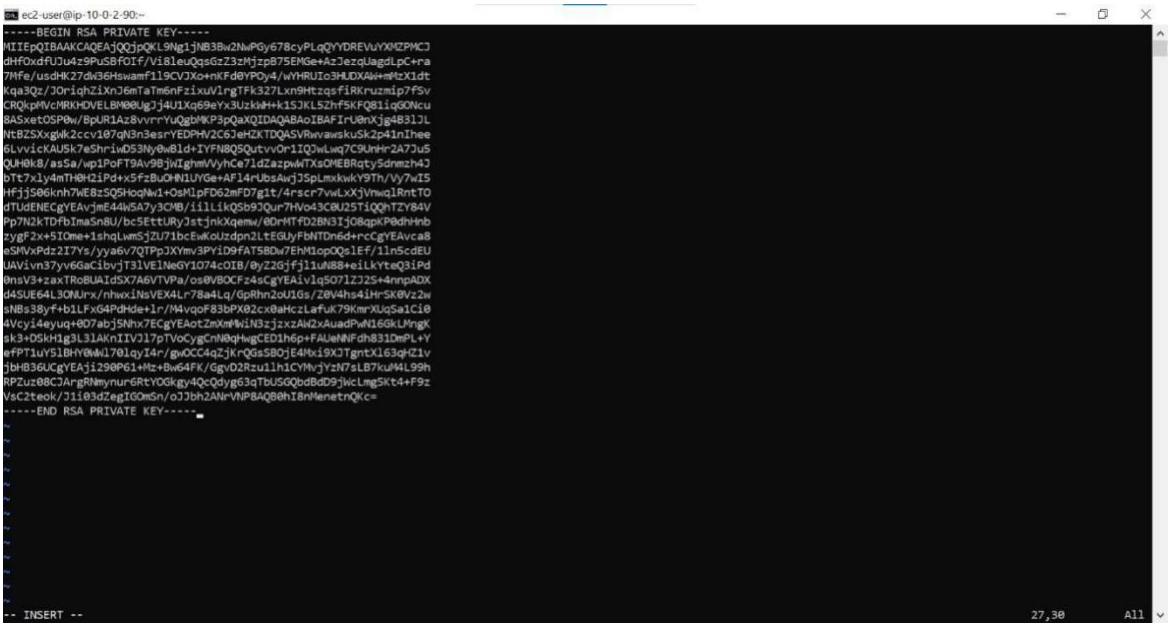
21) Select the private instance. Click on actions, security group and then Modify IAM role. Select the newly created role (project-role) and click on update IAM role.



22) Open cmd in the downloads folder where the private key is present. Run the following command to connect to the public instance:

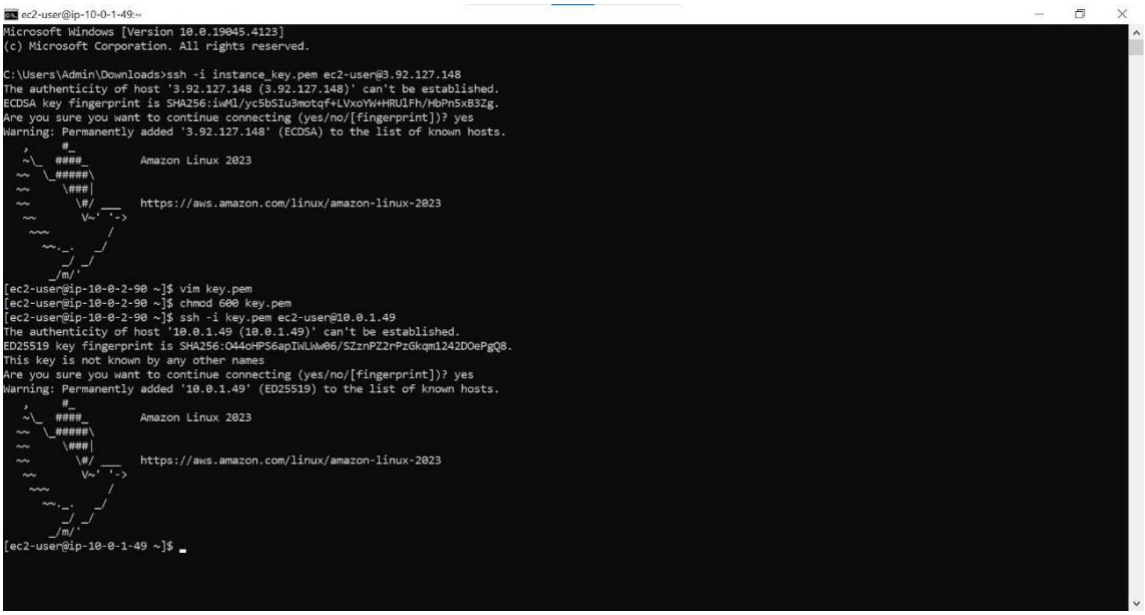
- **ssh -i instance_key.pem ec2-user@public_ip_public_instance**

Use command '**vim newkey.pem**' to create a file. Open the key in cmd in notepad and copy the key. Paste the key in our 'newkey.pem' file and save the file.



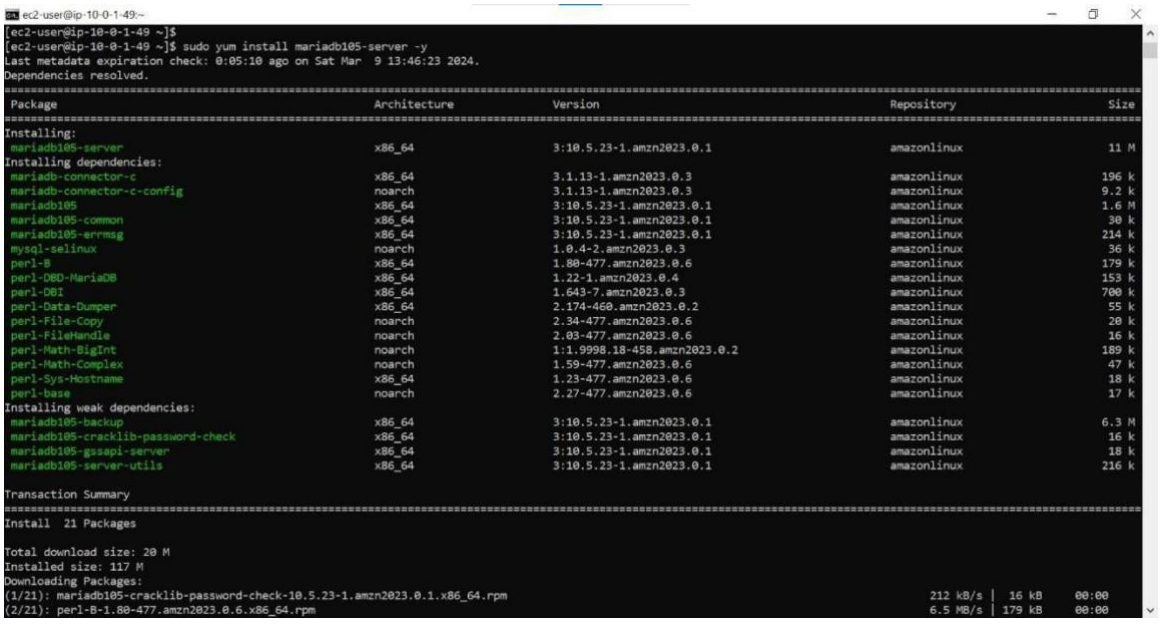
23) Once the key is copied, use the command '**chmod 600 newkey.pem**' to change permissions of the file. Then run the following command to take remote access of private instance on our public instance:

- ssh -i newkey.pem ec2-user@private_ip_private_instance



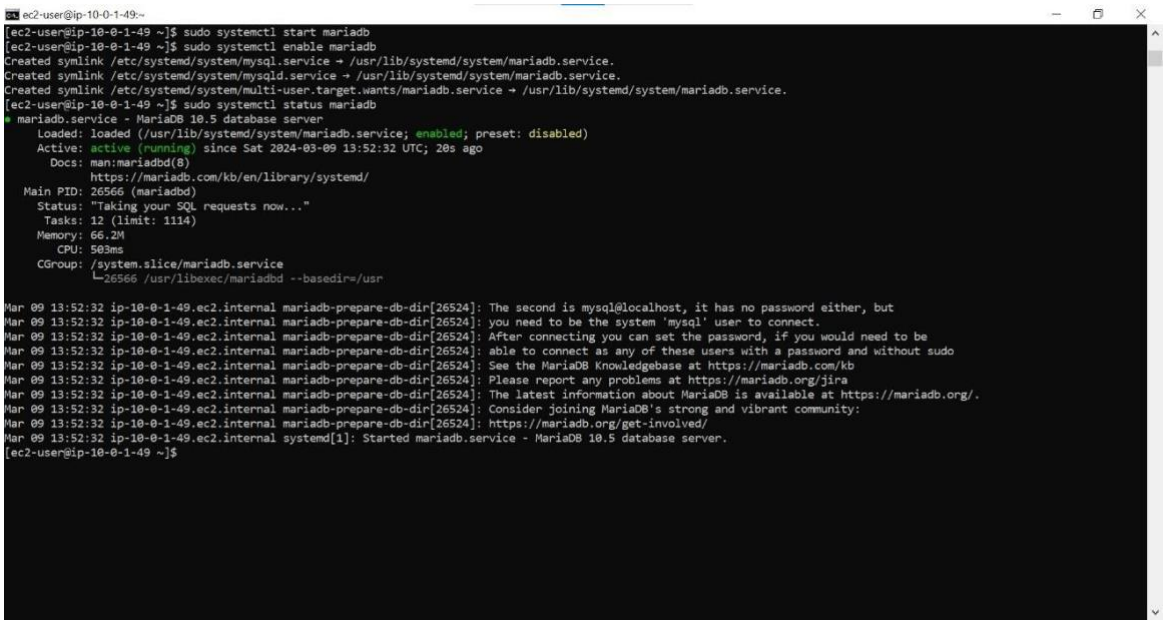
24) Now, to create a database download the mariadb package using the command:

- sudo yum install mariadb105-server -y

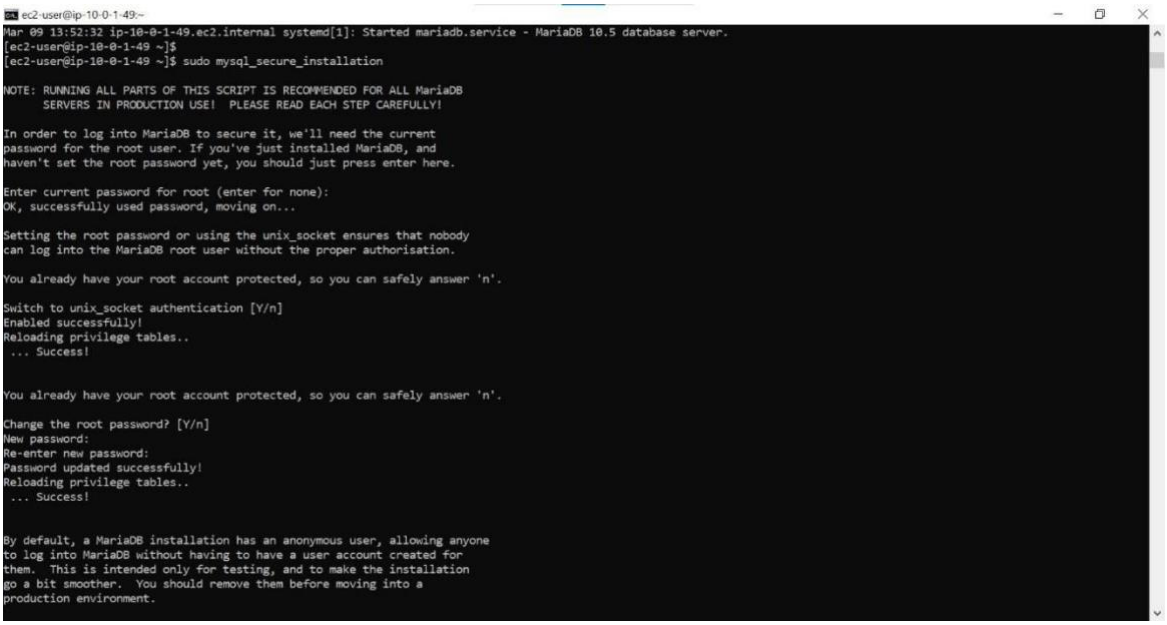


25) Start and enable the service once the package is downloaded.

- **sudo systemctl start mariadb**
- **sudo systemctl enable mariadb**



26) Using the '**sudo mysql_secure_installation**' command change the root user password to '1'.



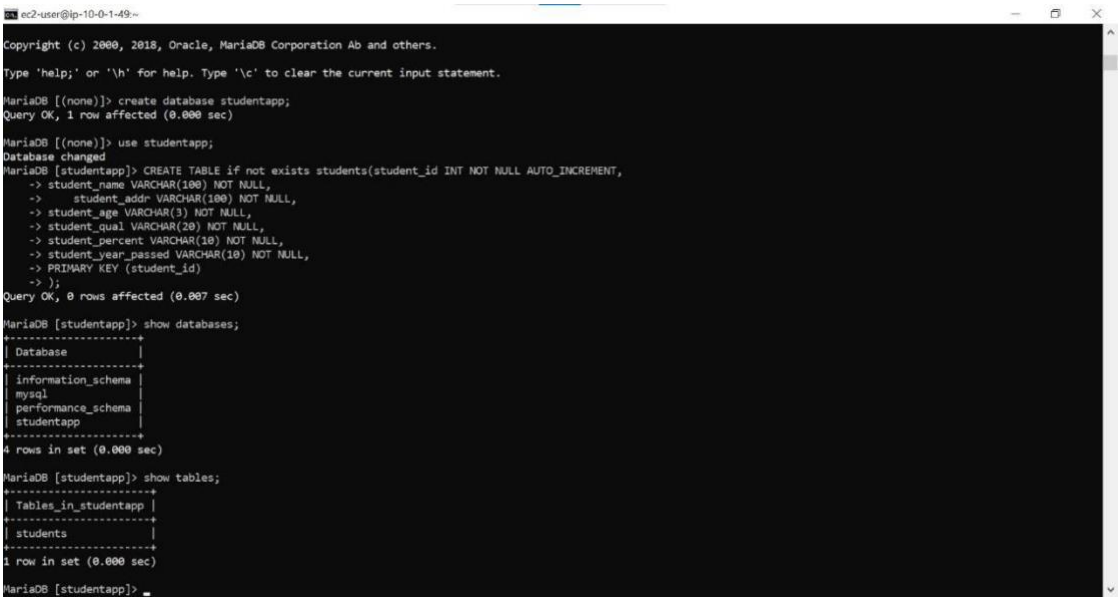
27) Login to mariadb server using the '**mysql -u root -p1**'command. Create a database named studentapp and a students table in it to store the data. Refer to the code bellow:

create database

studentapp; use

studentapp;

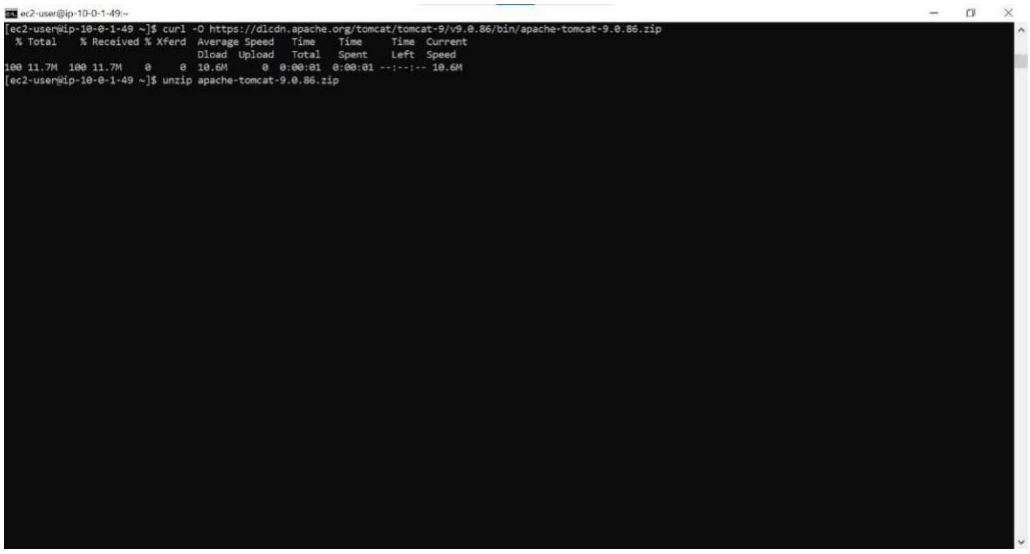
CREATE TABLE if not exists students(student_id INT NOT NULL AUTO_INCREMENT, student_name VARCHAR(100) NOT NULL, student_addr VARCHAR(100) NOT NULL, student_age VARCHAR(3) NOT NULL, student_qual VARCHAR(20) NOT NULL, student_percent VARCHAR(10) NOT NULL, student_year_passed VARCHAR(10) NOT NULL, PRIMARY KEY (student_id));



28) We need to download apache tomcat package to host the webpage, use command:

- **curl -O download_link_from_browser**

A zip file will be downloaded. Unzip the file using the '**unzip**' command.


ec2-user@ip-10-0-1-49:~
[ec2-user@ip-10-0-1-49 ~]$ curl -O https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.86/bin/apache-tomcat-9.0.86.zip
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 11.7M 100 11.7M 0 0 10.6M 0 0:00:01 0:00:01 --:--:-- 10.6M
[ec2-user@ip-10-0-1-49 ~]$ unzip apache-tomcat-9.0.86.zip

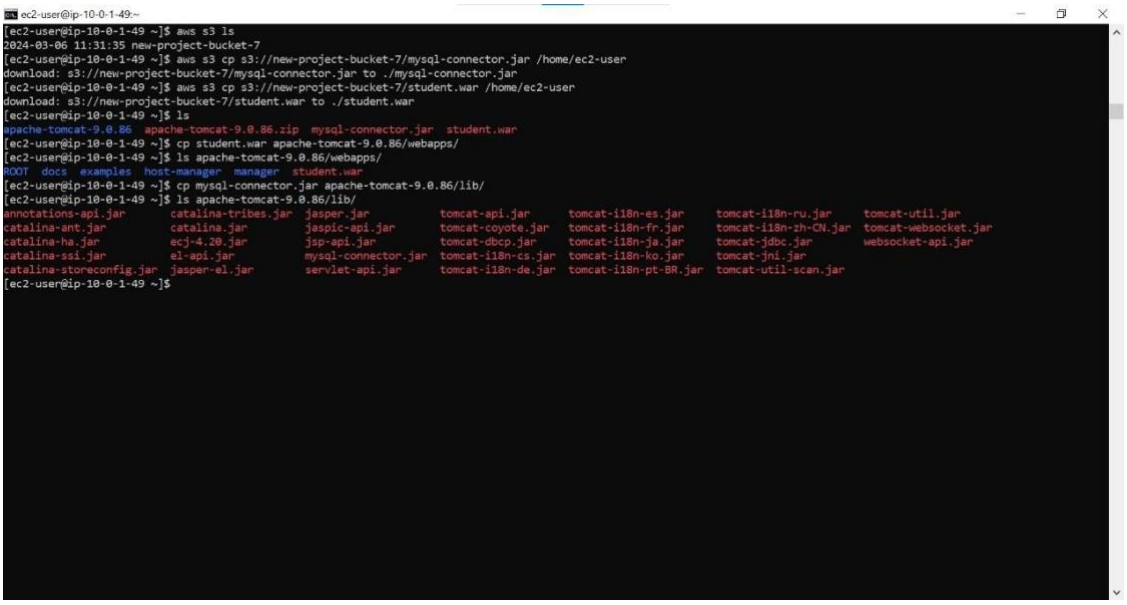
29) Next we need to download the files from bucket in the instance. Use command '**aws s3 ls**' to check if there is connection between ec2 instance and s3 bucket. Use the following command to download file from bucket into the terminal:

- **aws s3 cp S3_URI /home/ec2-user**

Once the files are downloaded, copy web page file in webapps directory and connector file in lib directory using the following commands:

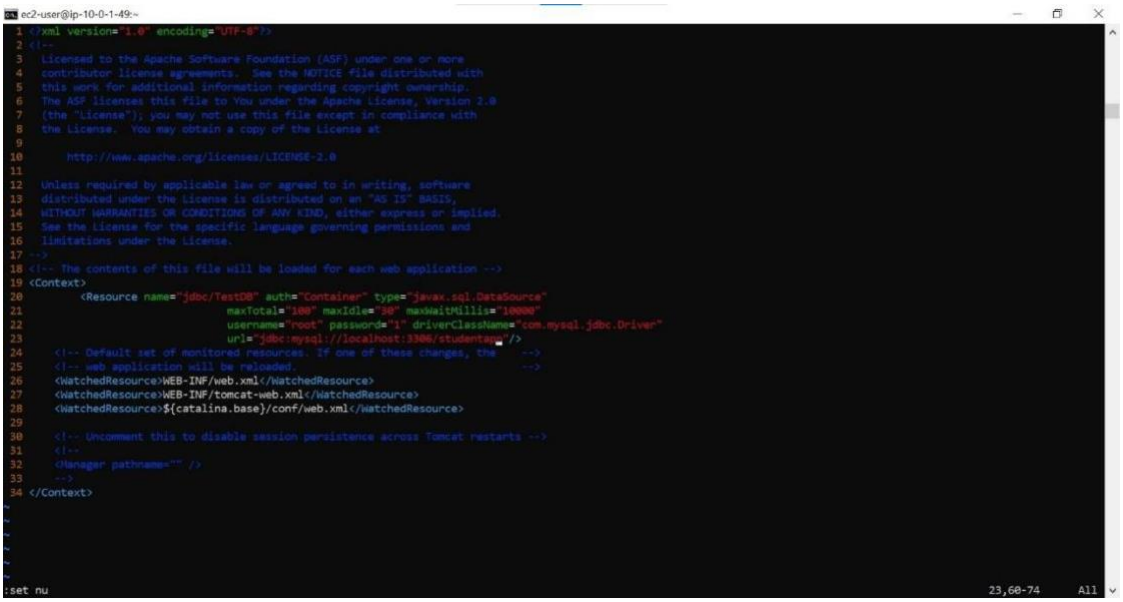
- **cp student.war apache-tomcat-9.0.86/webapps/**
- **cp mysql-connector.jar apache-tomcat-9.0.86/lib/**

Check is files have been copied using the '**ls**' command.

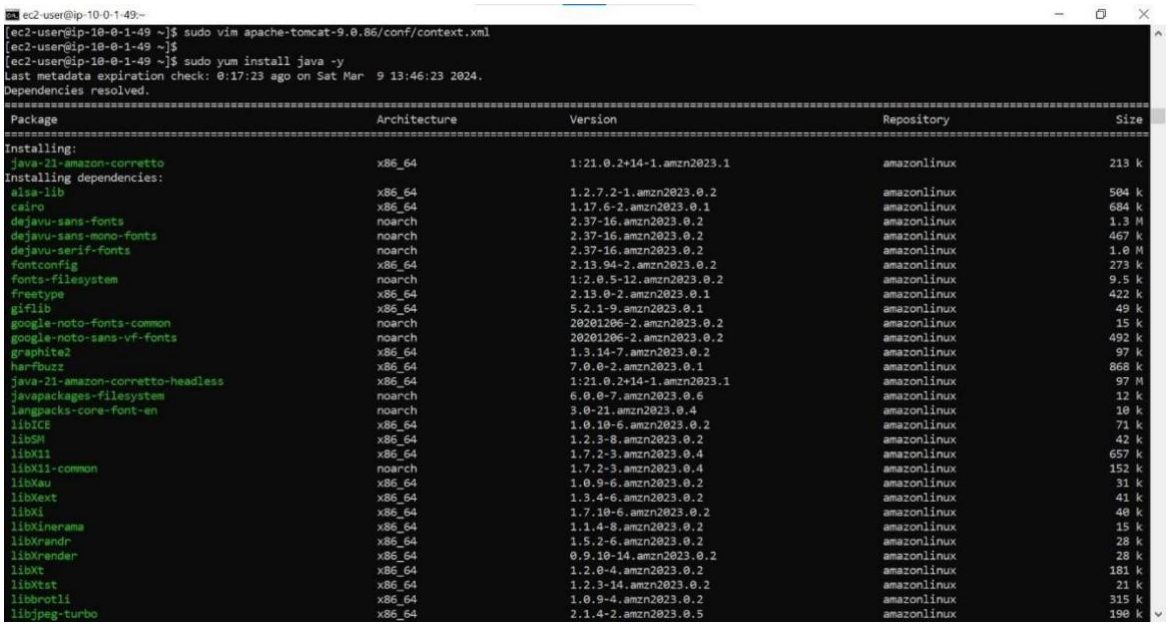


30) Using '**sudo vim apache-tomcat-9.0.86/conf/context.xml**' open the xml file and enter the following code on line 20 :

```
<Resource name="jdbc/TestDB" auth="Container"
 type="javax.sql.DataSource" maxTotal="100"
 maxIdle="30" maxWaitMillis="10000"
 username="root" password="1"
 driverClassName="com.mysql.jdbc.Driver"
 url="jdbc:mysql://localhost:3306/studentapp"/>
```

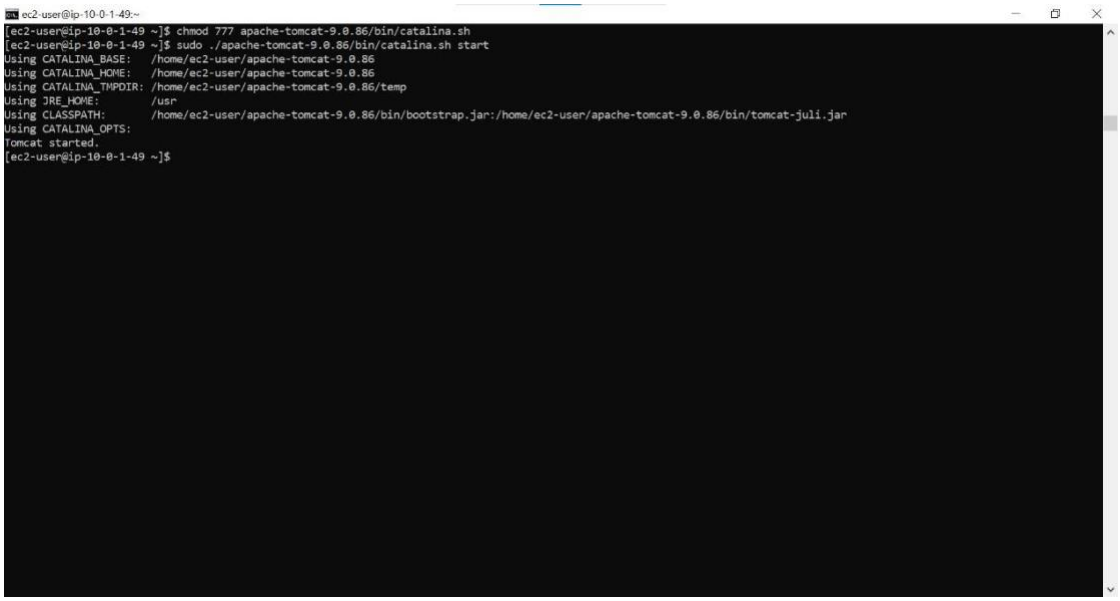


31) Download java package '**sudo yum install java -y**'. We need this package to start the apache tomcat server.



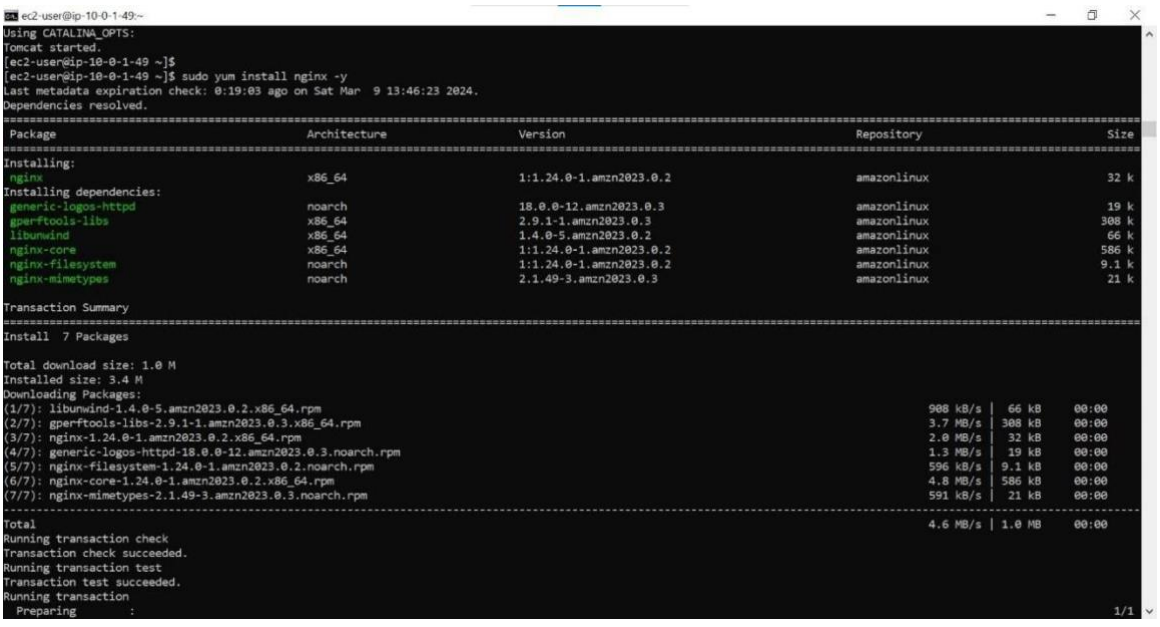
32) Use the command '**chmod 777 apache-tomcat-9.0.86/bin/catalina.sh**' to change permissions of the catalina file, then start the tomcat server:

- **sudo ./apache-tomcat-9.0.86/bin/catalina.sh start**



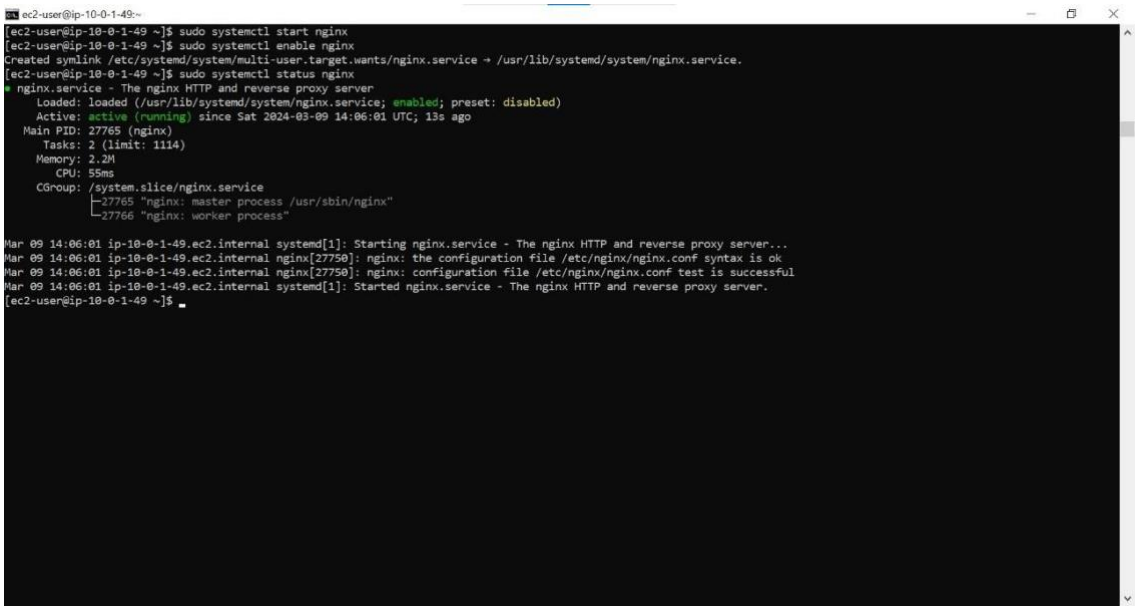
33) Now we need the nginx package to pass the incoming request from port 80 to port 8080.

- **sudo yum install nginx -y**



34) Start and enable the nginx service:

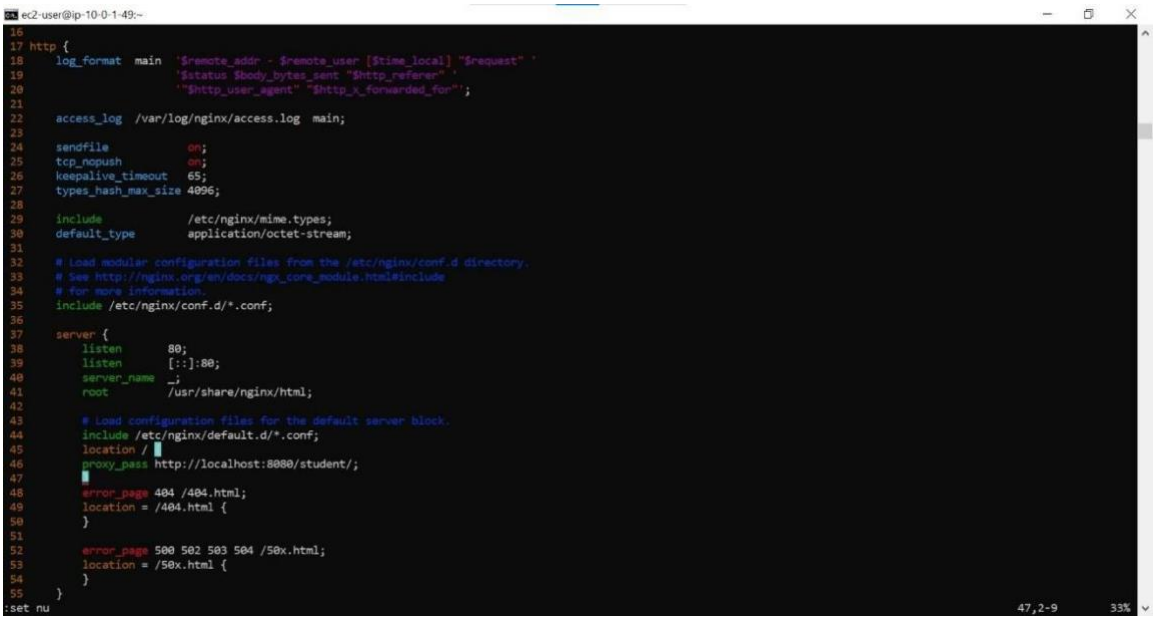
- **sudo systemctl start nginx**
- **sudo systemctl enable nginx**



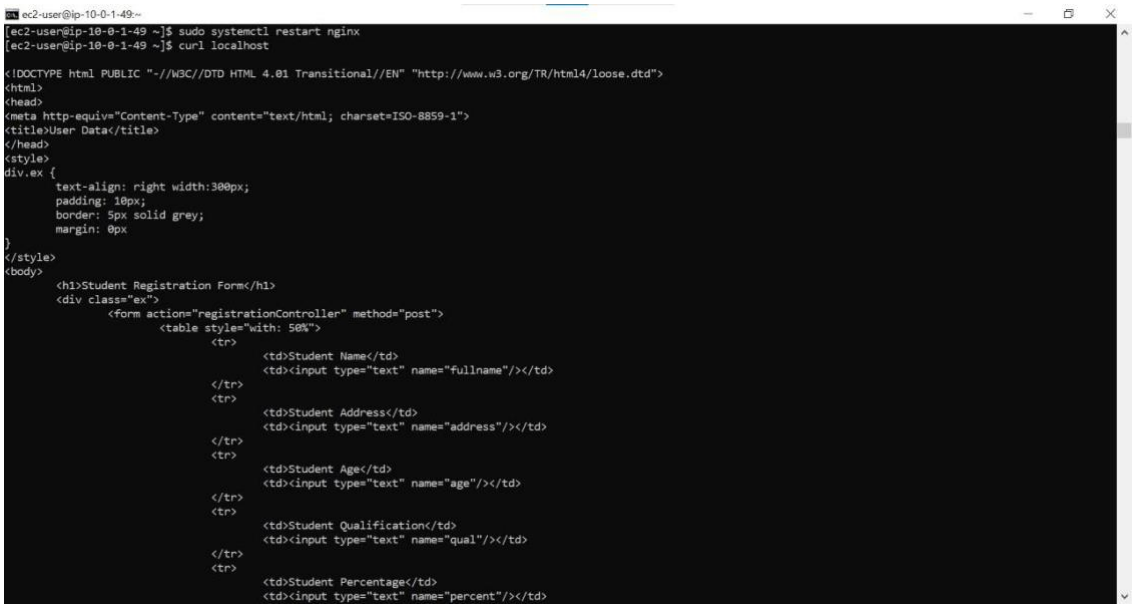
35) Open the nginx configuration file using '**sudo vim /etc/nginx/nginx.conf**' command and enter the following code on line 45:

```
location / {
proxy_pass http://localhost:8080/student/;
}
```

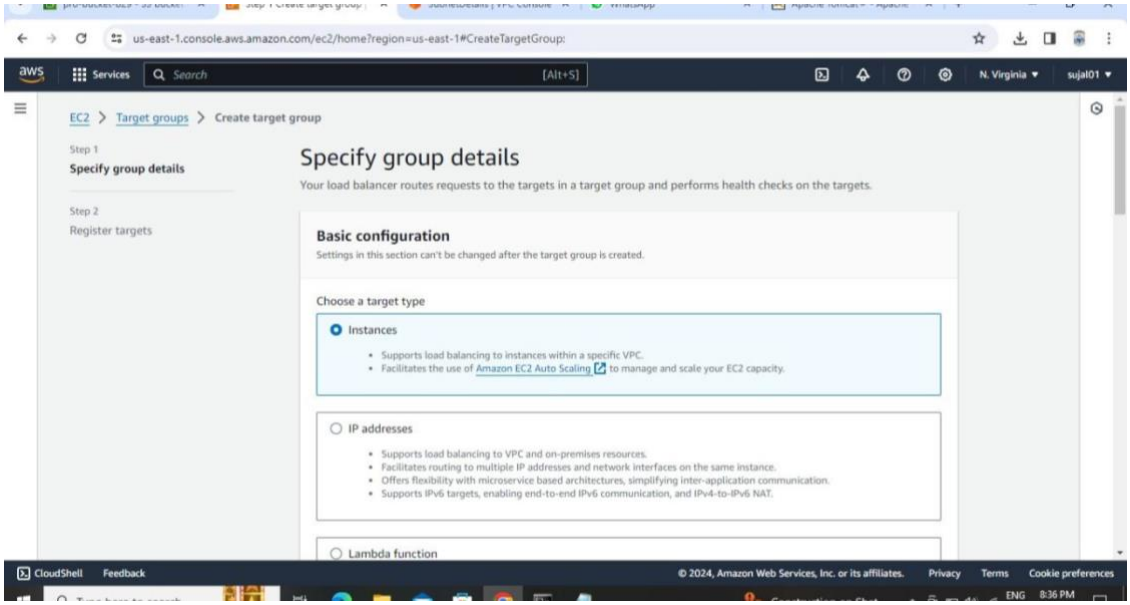
Save the file and restart the nginx service. (sudo systemctl restart nginx)


ec2-user@ip-10-0-1-49:~
http {
log_format main '$remote_addr - $remote_user [$time_local] "$request" '
'$status $body_bytes_sent "$http_referer" '
'"$http_user_agent" "$http_x_forwarded_for"';
access_log /var/log/nginx/access.log main;
sendfile on;
tcp_nopush on;
keepalive_timeout 65;
types_hash_max_size 4096;
include /etc/nginx/mime.types;
default_type application/octet-stream;
Load modular configuration files from the /etc/nginx/conf.d directory.
See http://nginx.org/en/docs/ngx_core_module.html#include
for more information.
include /etc/nginx/conf.d/*.conf;
server {
listen 80;
listen [::]:80;
server_name _;
root /usr/share/nginx/html;
Load configuration files for the default server block.
include /etc/nginx/default.d/*.conf;
location /
proxy_pass http://localhost:8080/student/;
error_page 404 /404.html;
location = /404.html {
}
error_page 500 502 503 504 /50x.html;
location = /50x.html {
}
}
:set nu
47,2-9
33%

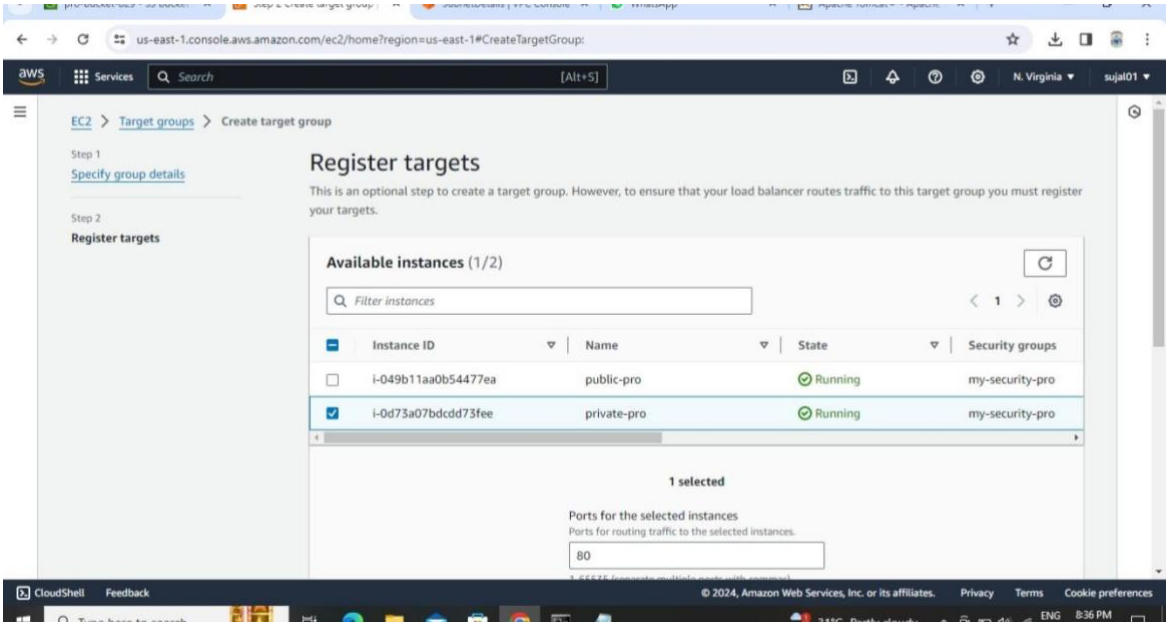
36) Now run the '**curl localhost**' command, you will get an output that is the code of our student registeration page. This output is a indication that we have completed all the configurations correctly.



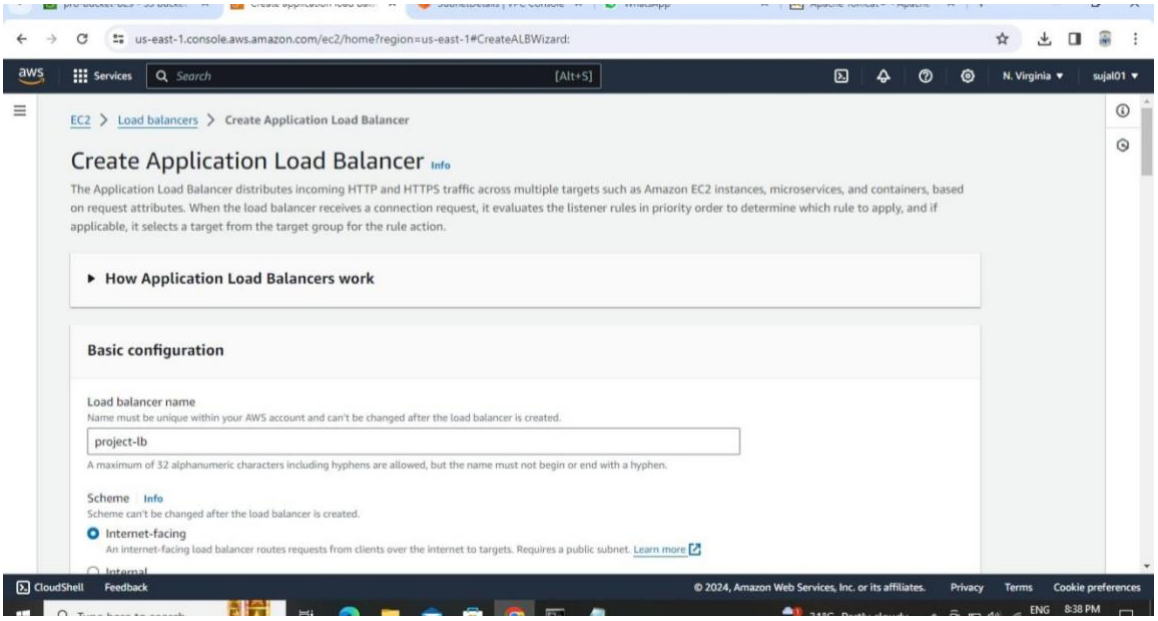
37) In the aws console, go to target groups dashboard under EC2 service and click on create target group. Enter a valid target group name (project-tg) and select our custom VPC in the VPC drop-down. Click on next.



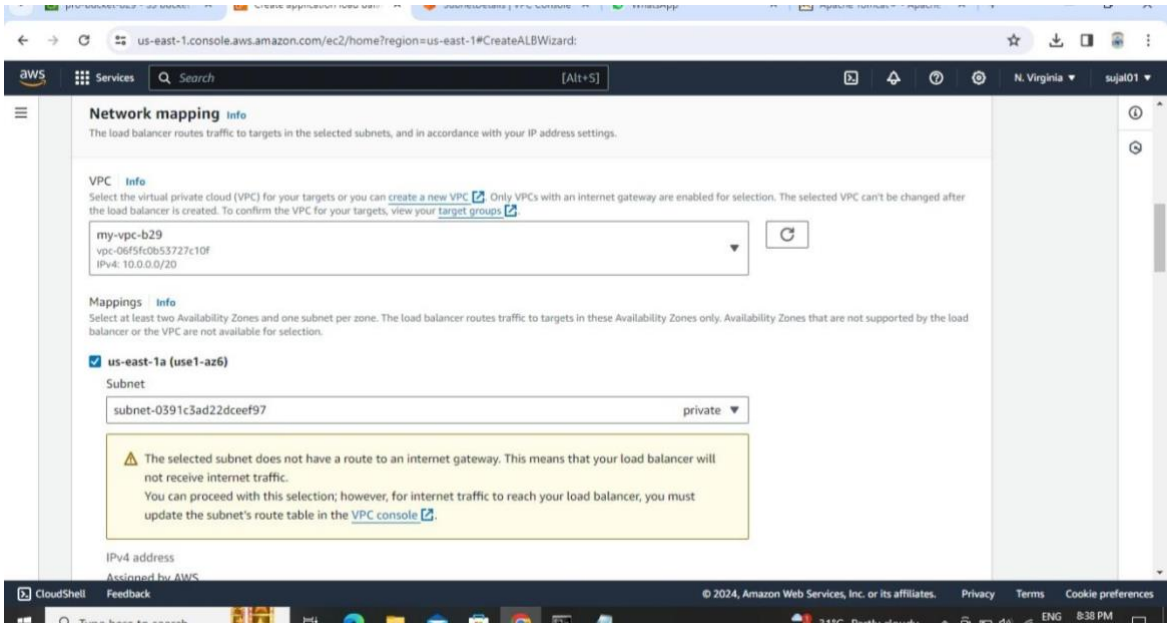
38) Add only the private instance as target and click on create target group.



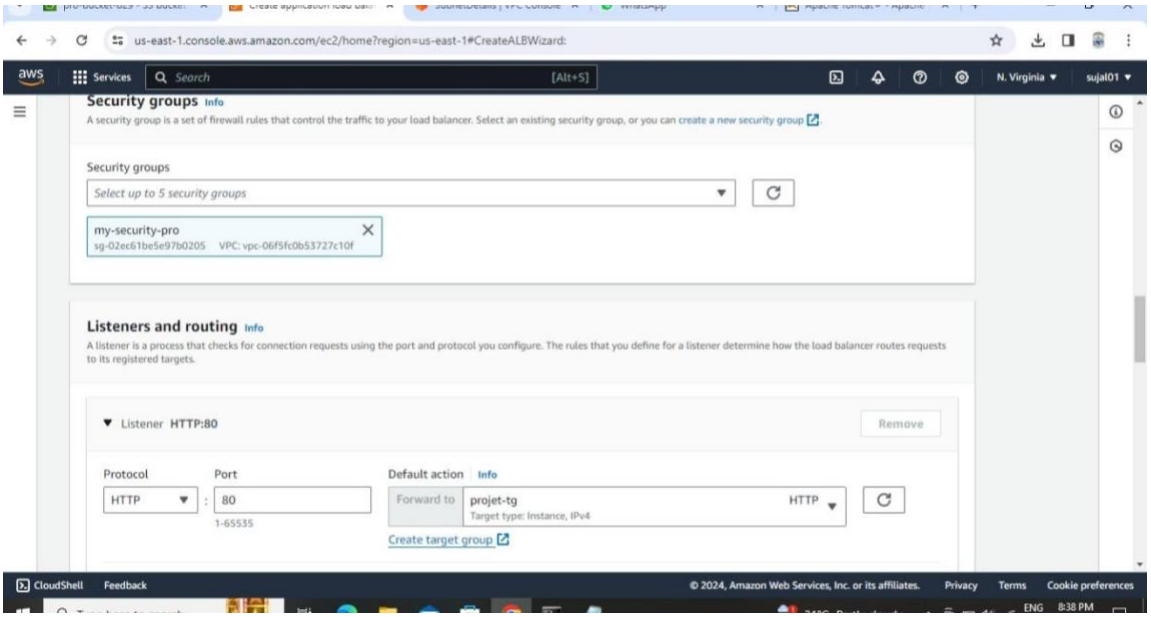
39) Now go to load balancers dashboard and click on create load balancer. Under Application Load Balancer click on create. Enter a valid name (sample-lb) and choose scheme as 'internet-facing'.



40) Select our custom VPC and check both the availability zones under mappings.



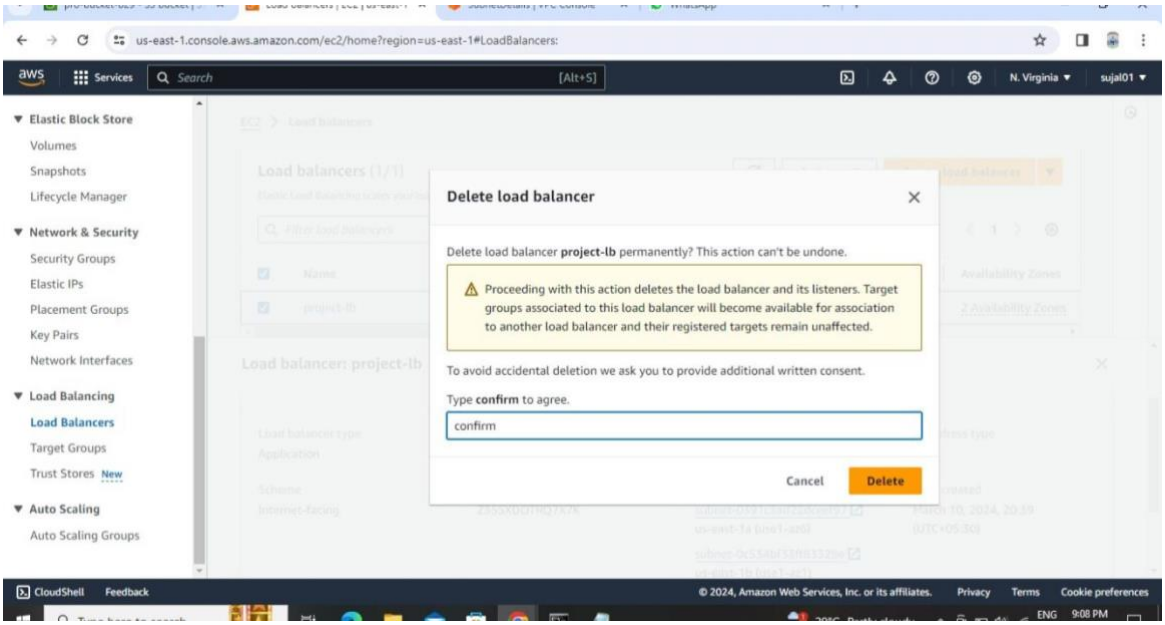
41) Under security group choose the group that we created (launch-wizaard-7) and add the newly created target group (project-tg) in listeners and routing section. Click on create load balancer.



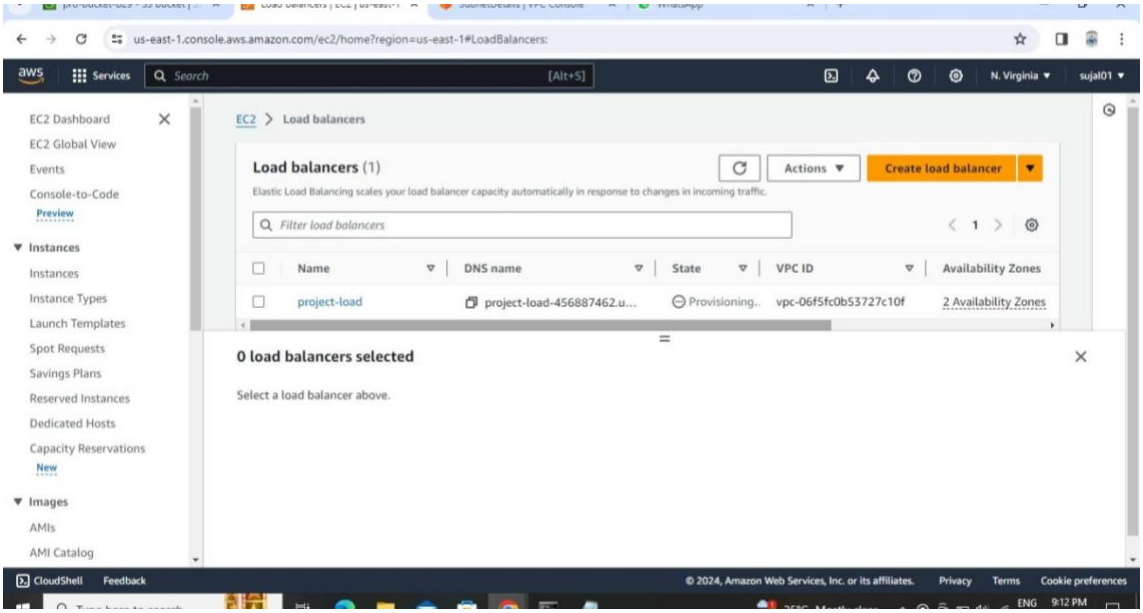
42) Once the load balancer is created and goes into active state, copy the dns name and paste it in a new incognito window. You will see that the web page loads successfully and you can also register the users here.

Student ID	StudentName	Student Addrs	Student Age	Student Qualification	Student Percentage	Student Year Passed	Edit	Delete
1	vaibhav	dhagat	89	7th	25	1999	edit	delete

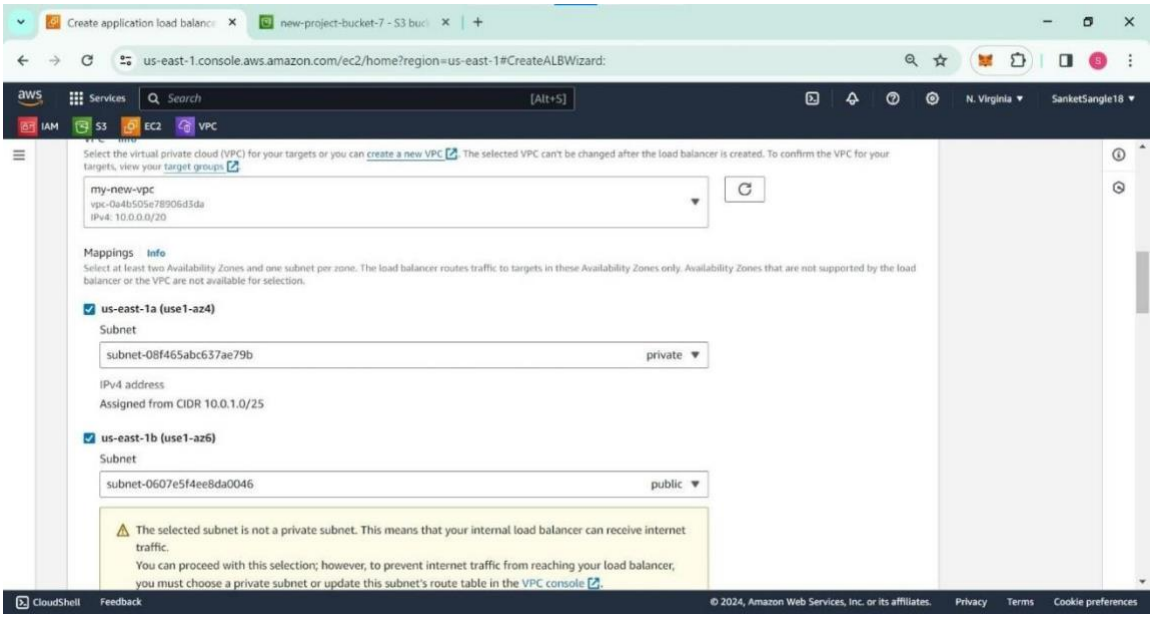
43) On the load balancers dashboard delete the newly created load balancer (sample-lb). We only needed the load balancer for testing purpose.



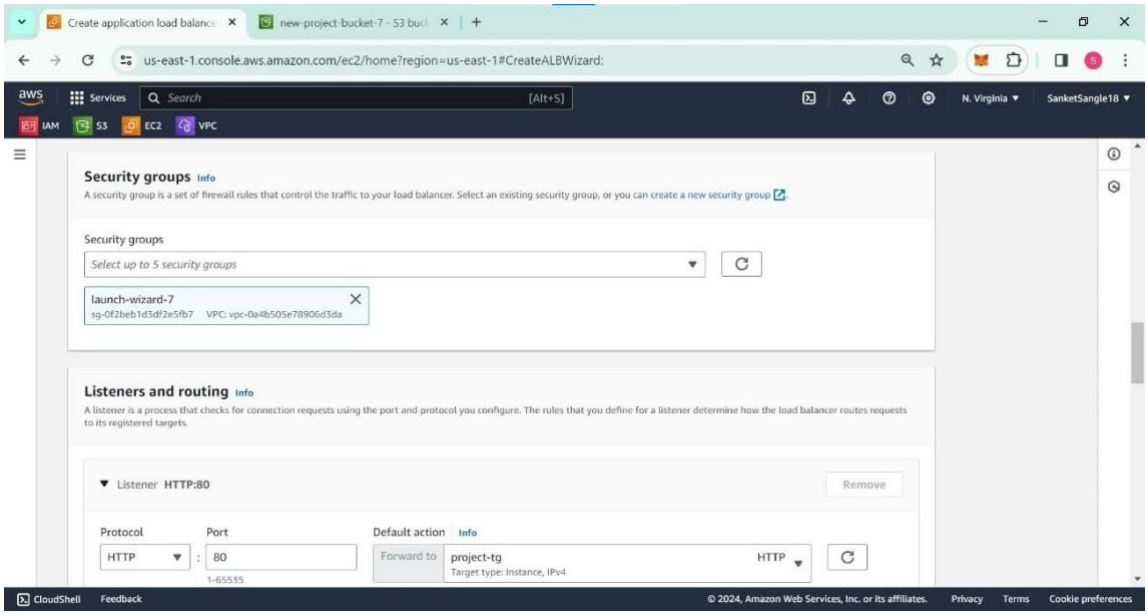
44) Again, on load balancers dashboard click on create load balancer. Under Application Load Balancer click on create. Enter a valid name (project-lb) and this time choose scheme as ‘internal’.



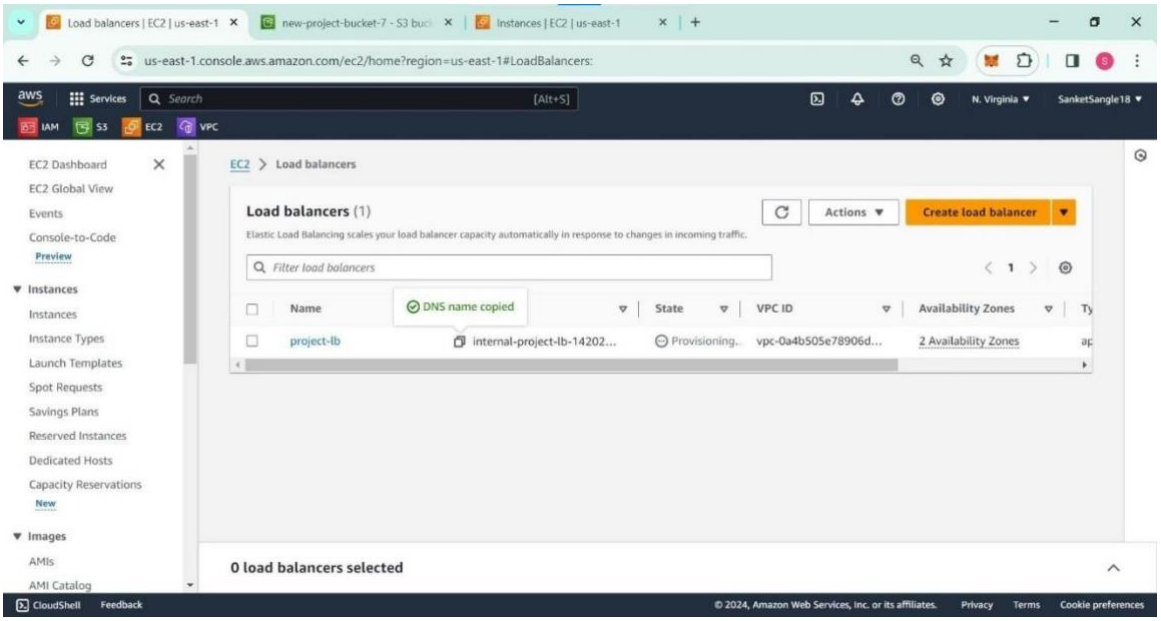
45) Select our custom VPC and check both the availability zones under mappings.



46) Under security group choose the group that we created (launch-wizaard-7) and add the newly created target group (project-tg) in listeners and routing section. Click on create load balancer.



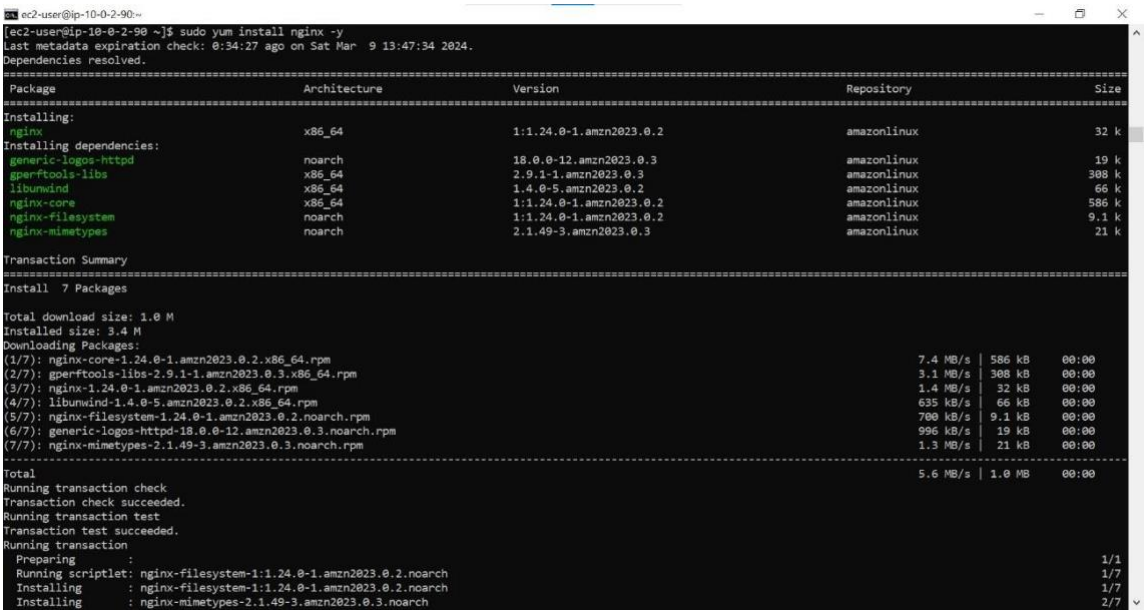
47) Our load balancer is successfully created. Copy the dns name of load balancer.



48) Open cmd in the downloads folder where the private key is present. Run the following command to connect to the public instance:

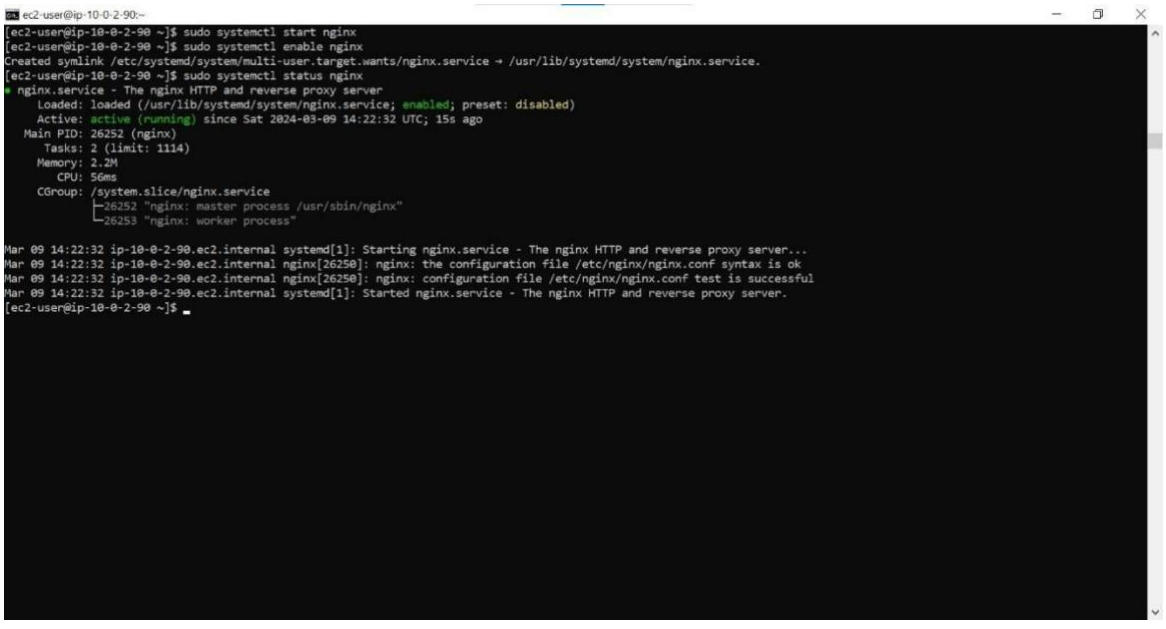
- **ssh -i instance_key.pem ec2-user@public_ip_public_instance**

Download the nginx package: **sudo yum install nginx -y**



49) Start and enable the nginx service:

- **sudo systemctl start nginx**
- **sudo systemctl enable nginx**



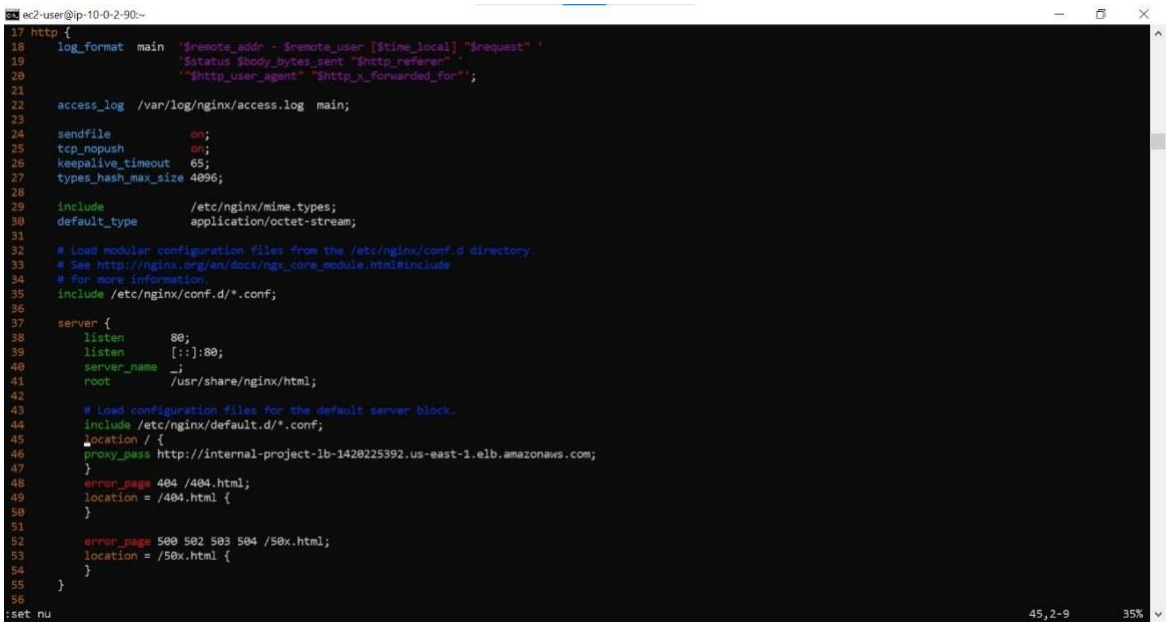
50) Open the nginx configuration file using '**sudo vim /etc/nginx/nginx.conf**' command and enter the following code on line 45:

location / {

proxy_pass http://dns_of_load_balancer;

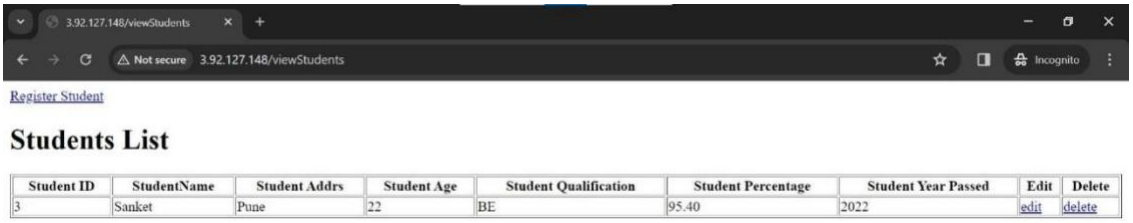
}

Save the file and restart the nginx service. (sudo systemctl restart nginx)

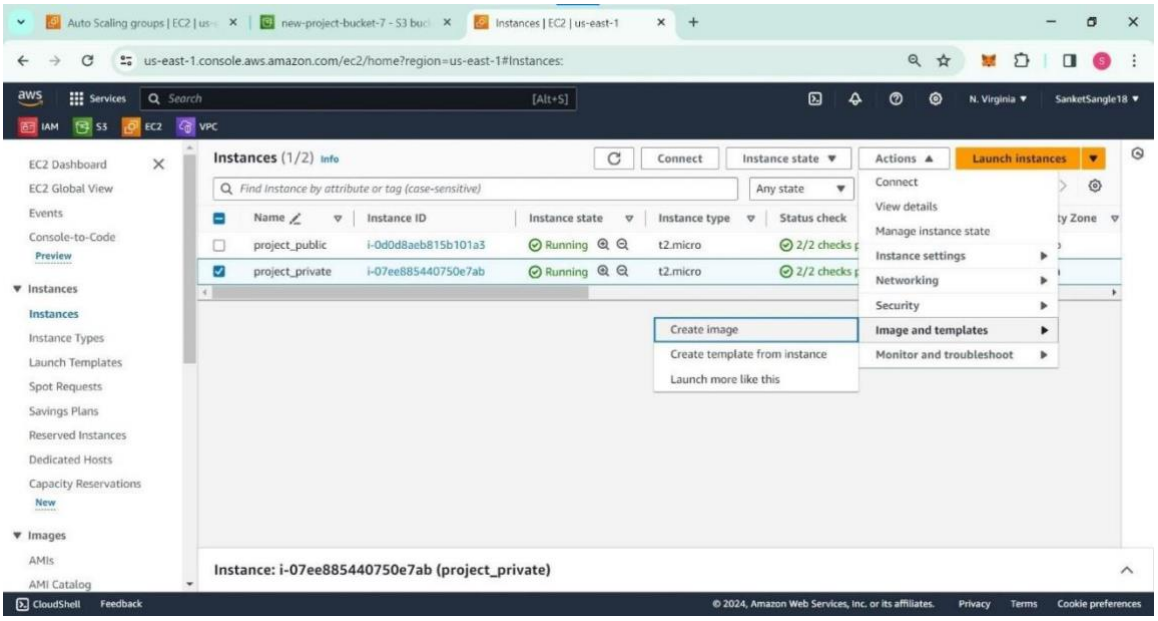


51) In chrome browser open new incognito window and paste the public ip address of public instance (project_public). Enter the student data and click on register.

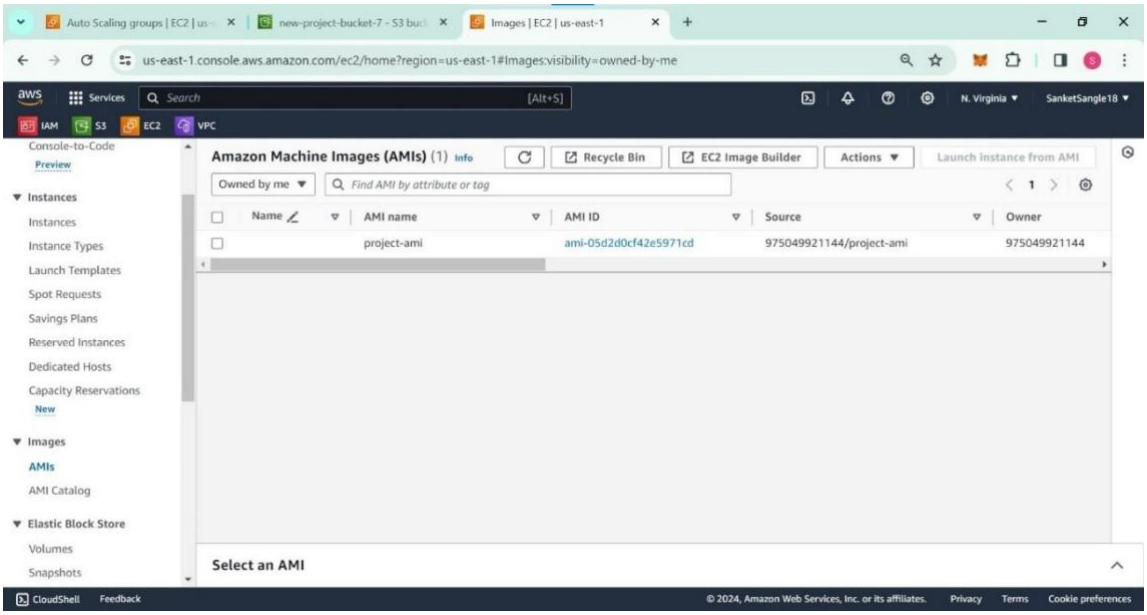
52) The student data has been successfully registered.



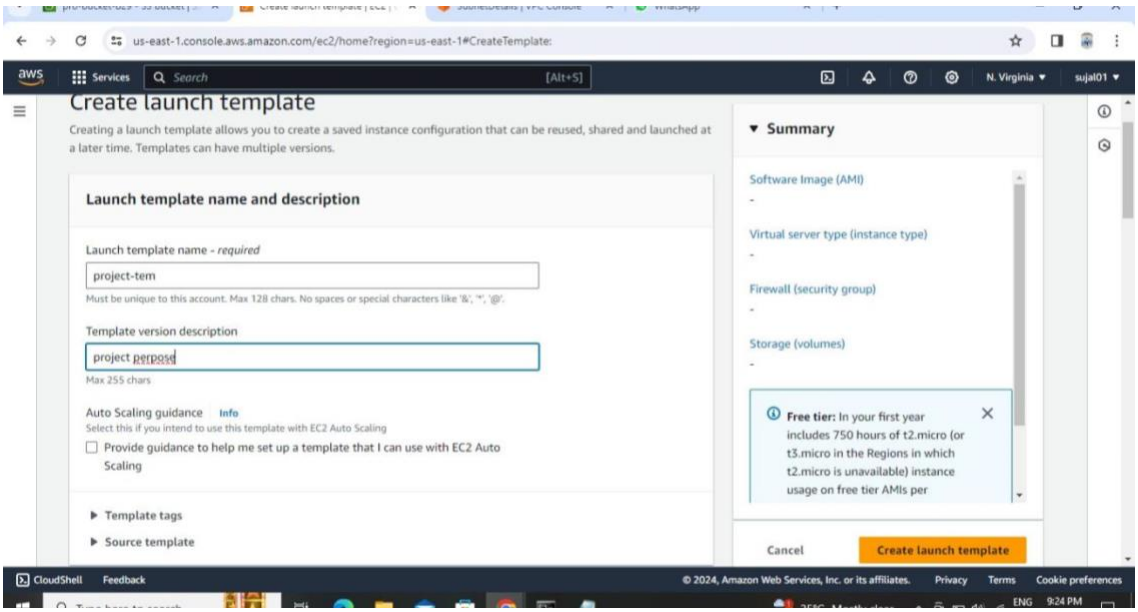
53) On instances dashboard select the private instance and click on actions then image and templates and click on create image.



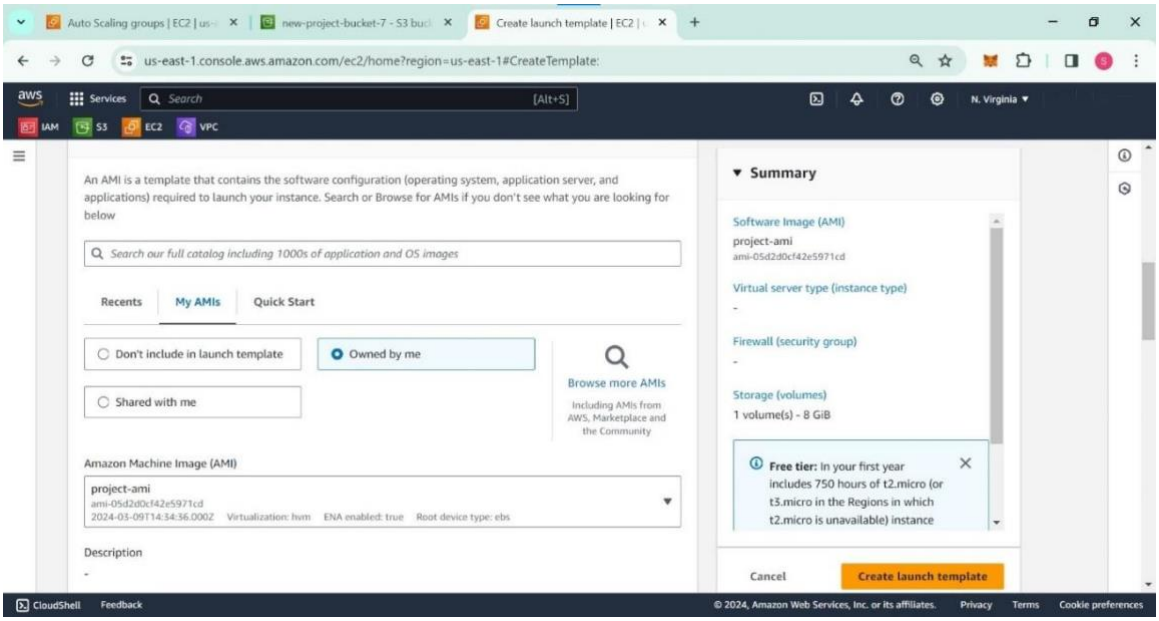
54) In AMIs dashboard you can see that the image has been successfully created.



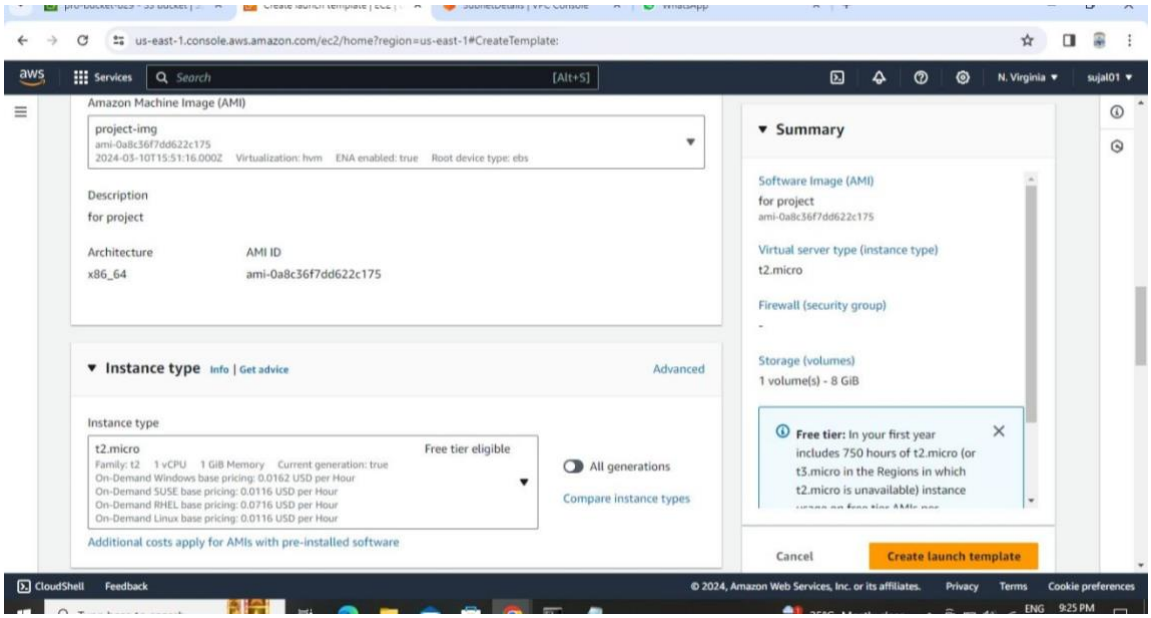
55) Go to launch templates dashboard under the EC2 service and click on create launch template. Enter a valid name (project-template) and add some description.



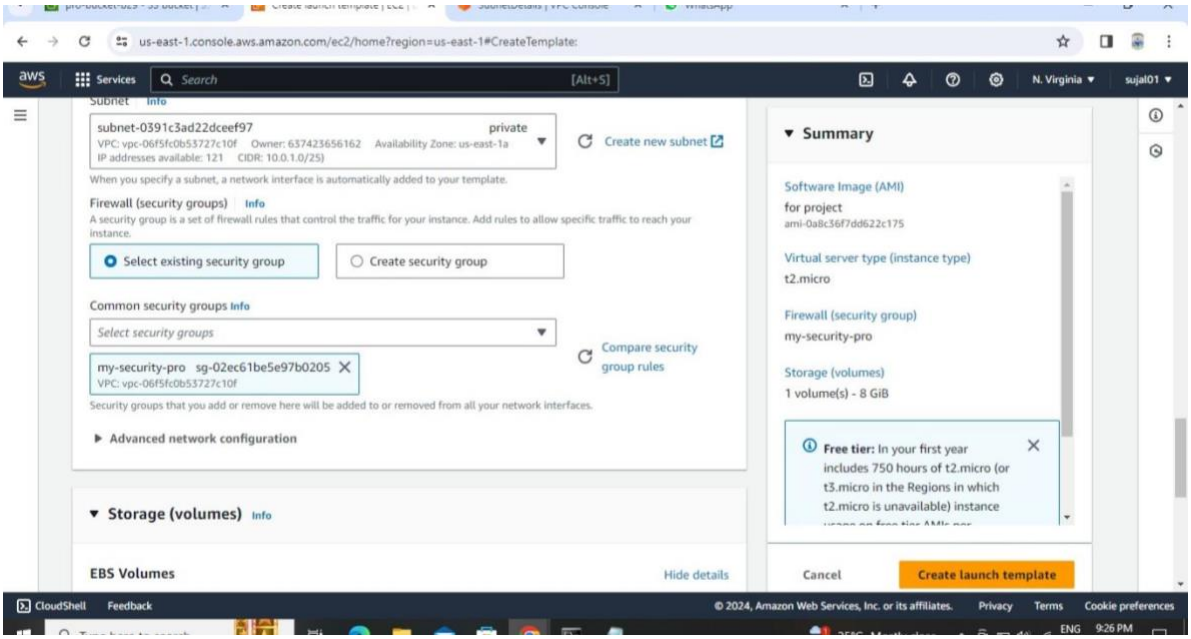
56) Select the newly created AMI under My AMIs section.



57) Select instance type as ‘t2.micro’ and the key pair same as that of created instances.



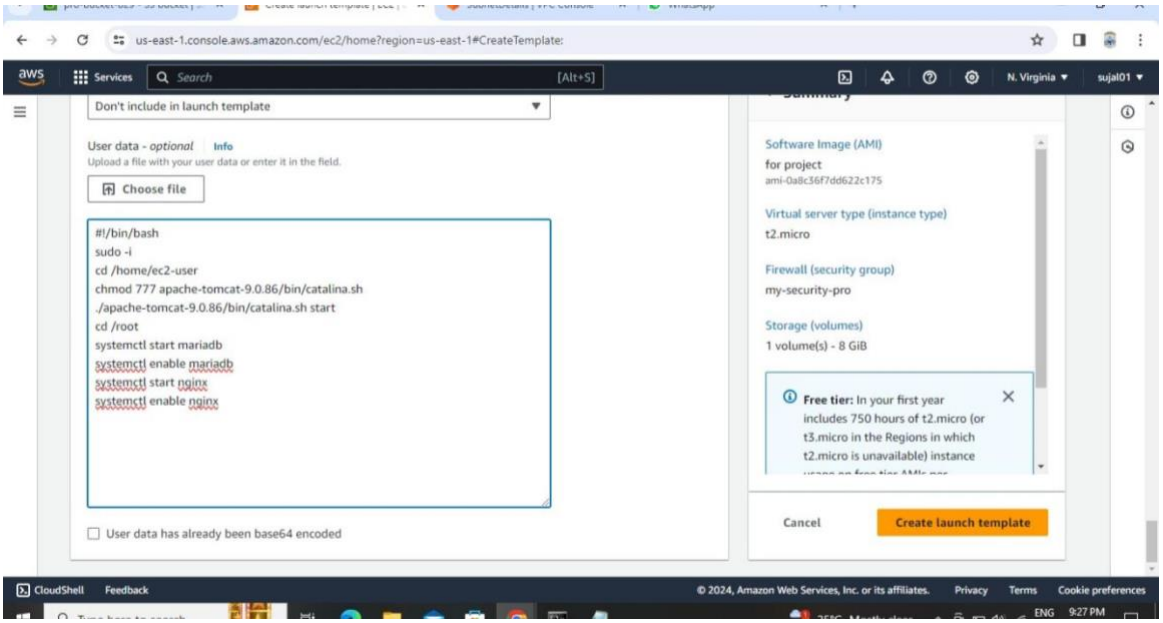
58) In network settings select the custom VPC and the private subnet. Choose the created security group (launch-wizard-7).



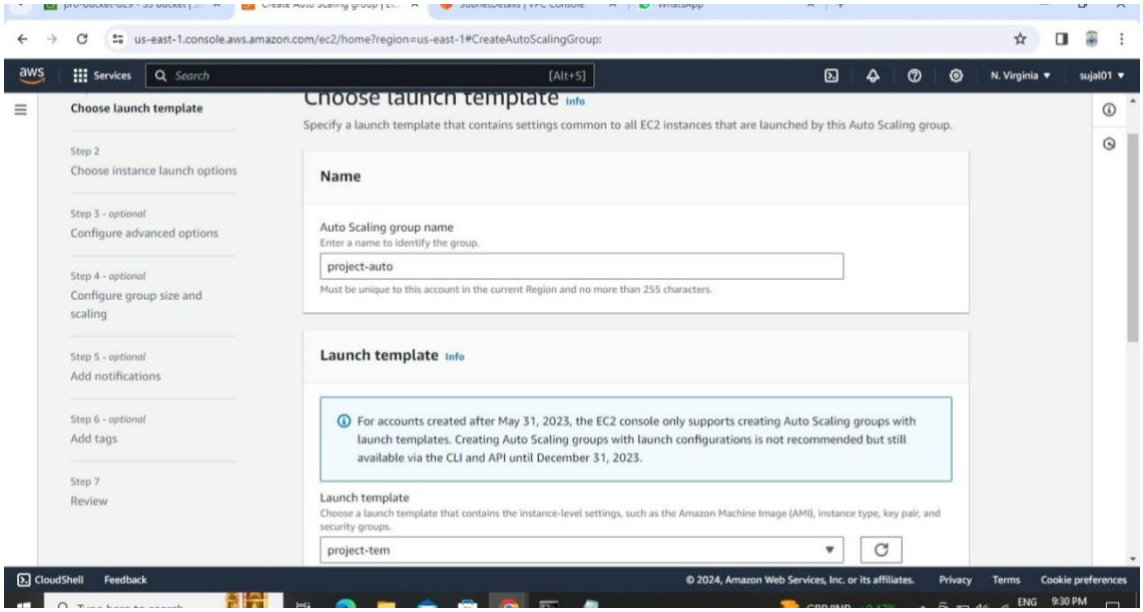
59) Click on advanced settings and scroll down to user data section and add the following script:

```
#!/bin/bash
sudo -i
cd /home/ec2-user
chmod 777 apache-tomcat-9.0.86/bin/catalina.sh
./apache-tomcat-9.0.86/bin/catalina.sh  start
cd /root
systemctl start mariadb
systemctl enable mariadb
systemctl start nginx
systemctl enable nginx
```

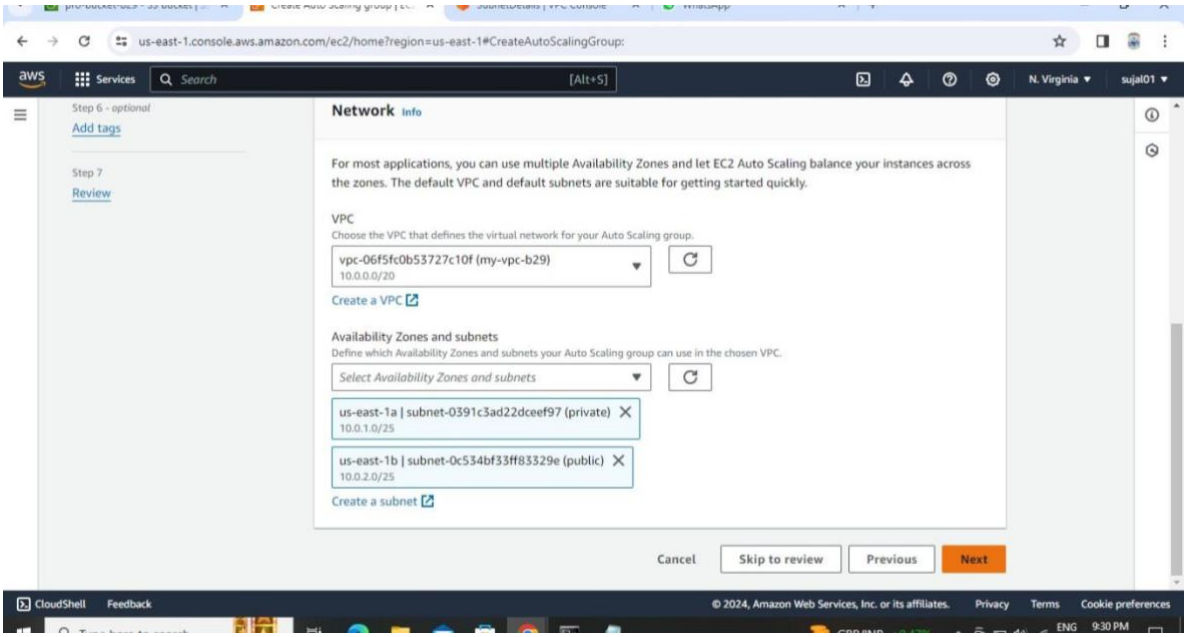
Click on create launch template.



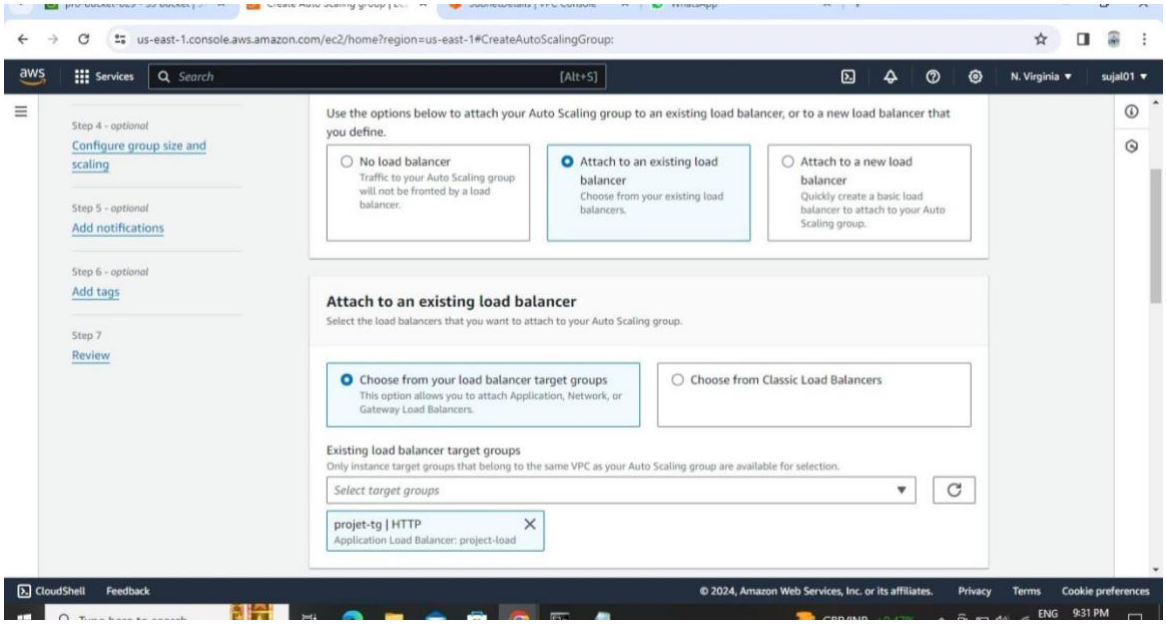
60) Go to Auto Scaling Group dashboard and click on create auto scaling group. Enter a valid name (project-asg) and select the newly created launch template (project-template). Enter data in other fields if required.



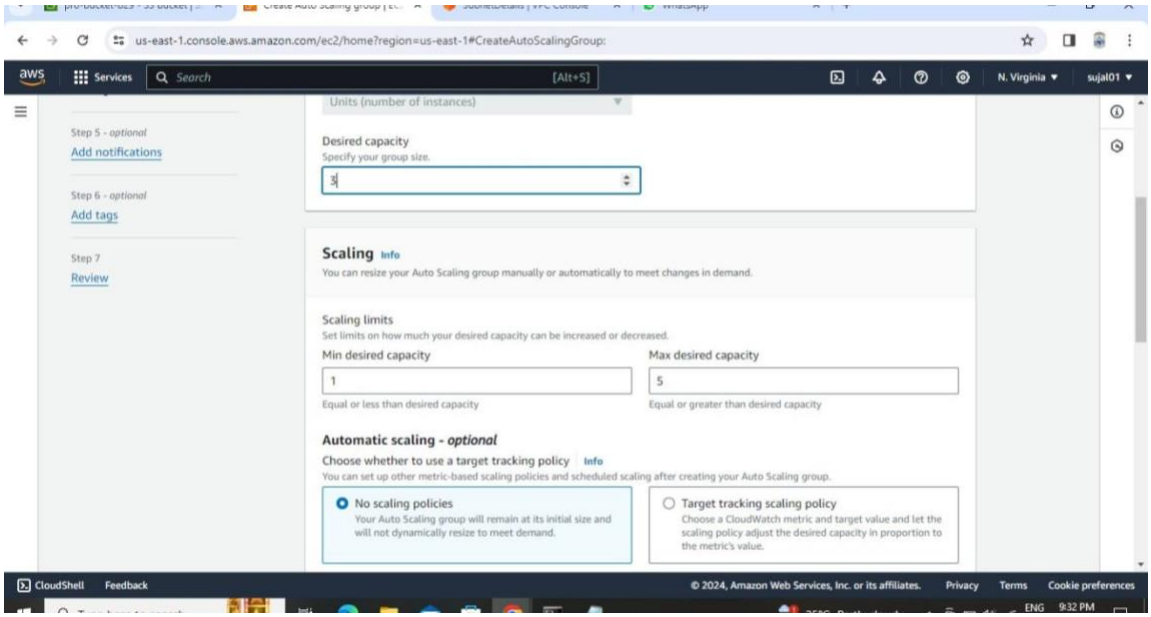
61) Select the custom VPC and choose both the availability zones. Click on next.



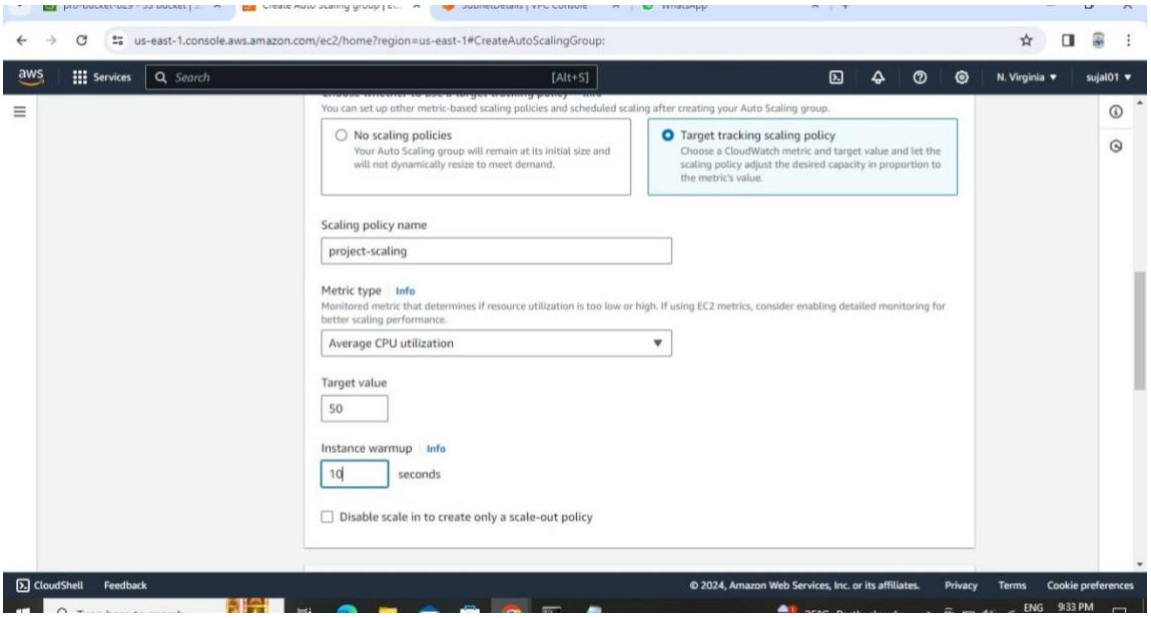
62) Choose the attach to an existing load balancer option and select the created target group (project-tg) under existing load balancer target groups.



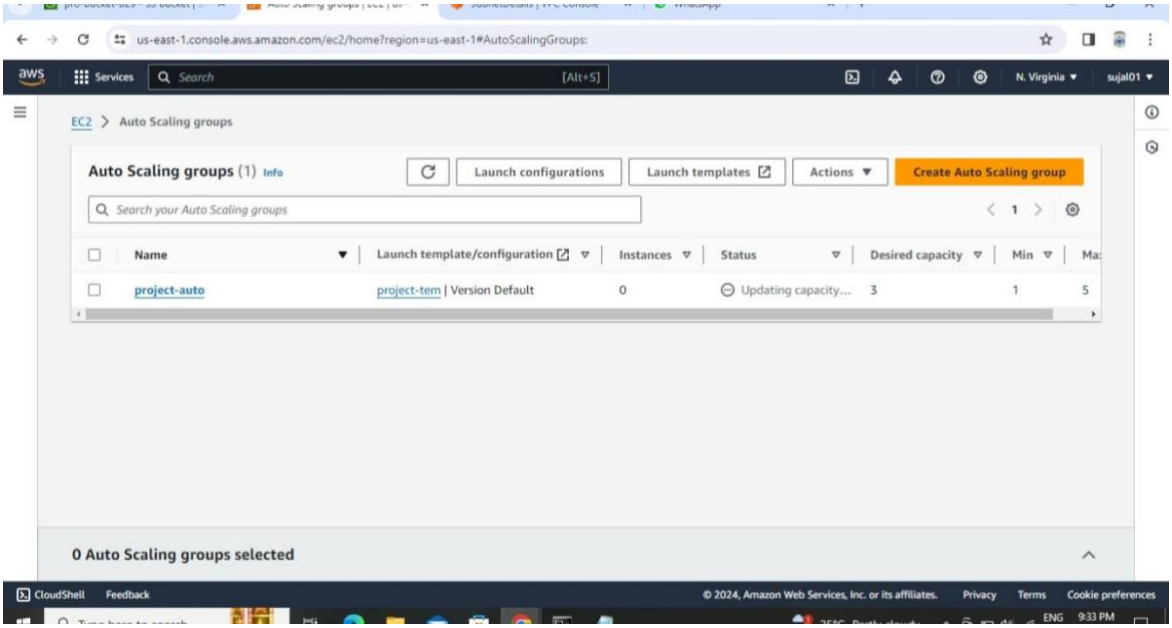
63) Enter the desired capacity, min desired capacity and max desired capacity as per requirement.



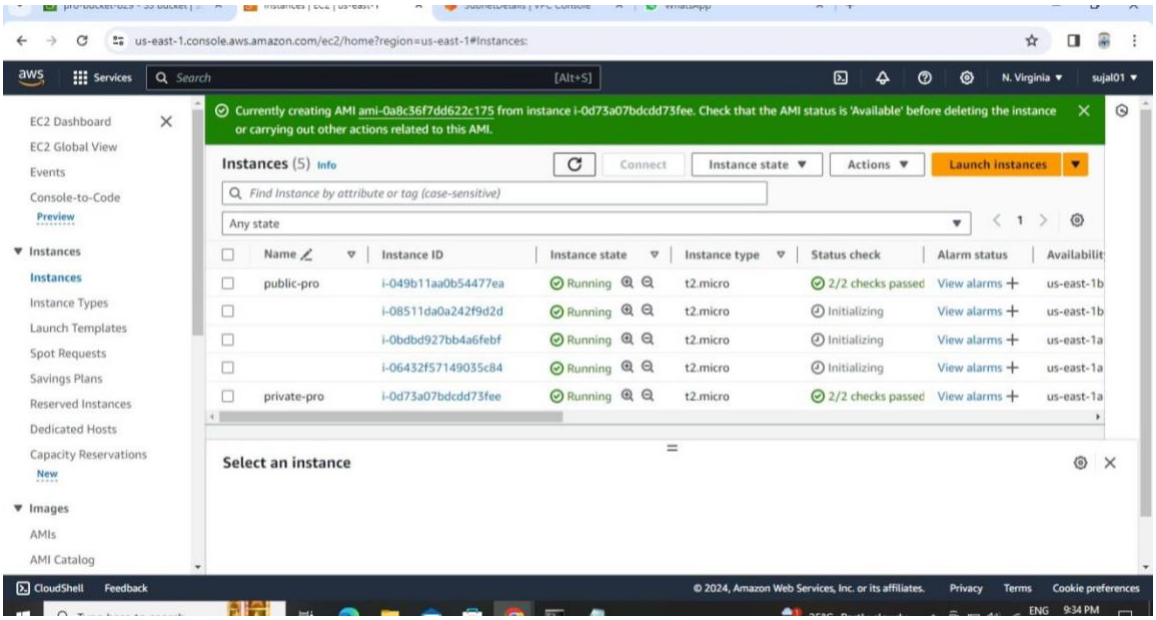
64) Choose the target tracking scaling policy option and fill other fields as per requirement. Review and click on create auto scaling group.



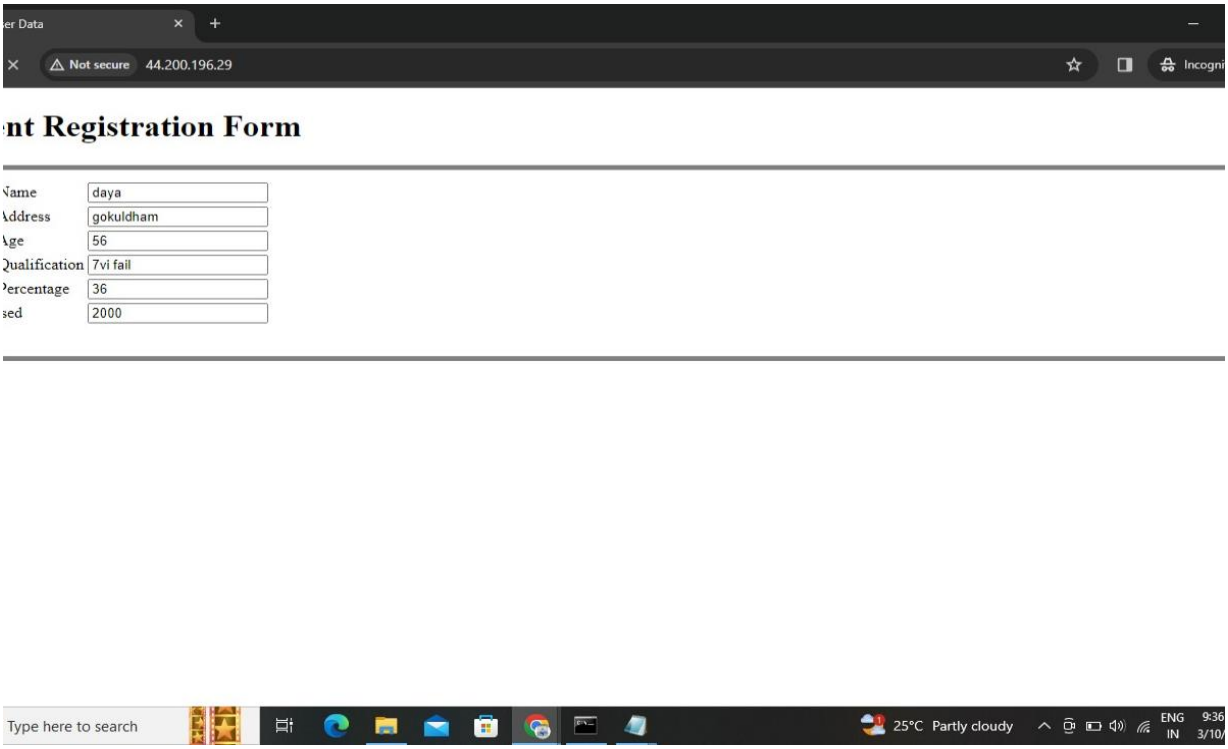
65) The auto scaling group has been successfully created.



66) The new instances have been automatically created by the scaling group.



67) In new incognito window paste the public ip of public instance (project_public). The web page will be loaded successfully. Enter the student data and click on register.



68) Keep refreshing the page to see traffic being balanced by the load balancer.

Register Student

Students List

Student ID	StudentName	Student Addrs	Student Age	Student Qualification	Student Percentage	Student Year Passed	Edit	Delete
1	vaibhav	dhagat	89	7th	25	1999	edit	delete
2	prathm	aundh	aws	none	35	1967	edit	delete
3	daya	gokuldham	56	7vi fail	36	2000	edit	delete