Project-3 Deployment of Online Food Ordering and Delivery Application

Submitted By – Arya Chaurasia

Date of Submission – 16/12/2025

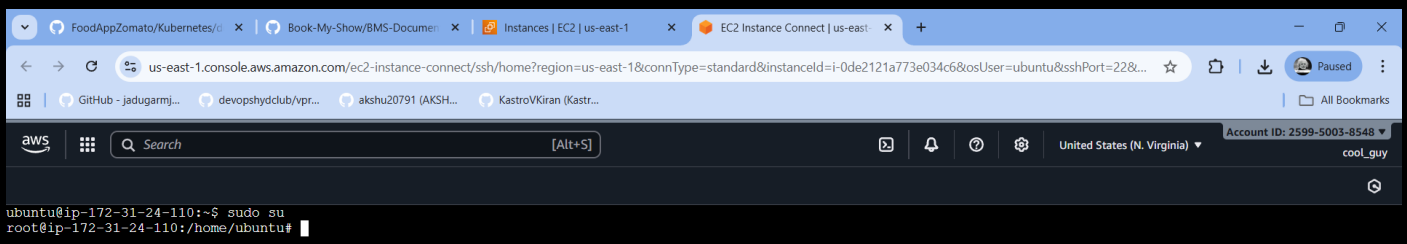
Project Overview

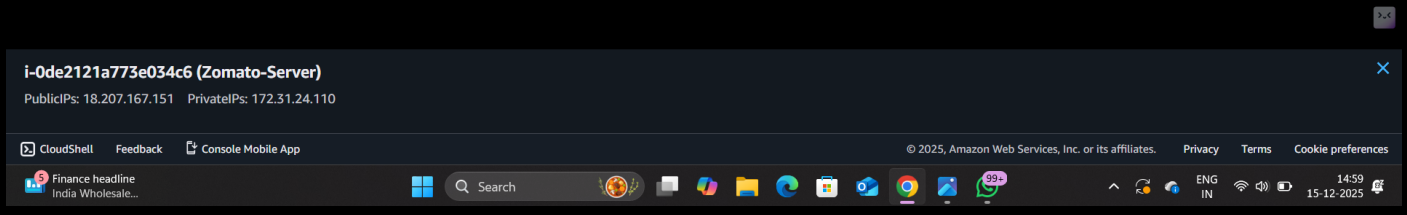
Zomato is a leading online food delivery and restaurant discovery platform. To handle increasing user demand and improve deployment efficiency, the company is transitioning from a monolithic architecture to a microservices-based architecture using DevOps practices.

GitHub link - https://github.com/Aryachaurasia23/FoodAppZomato.git

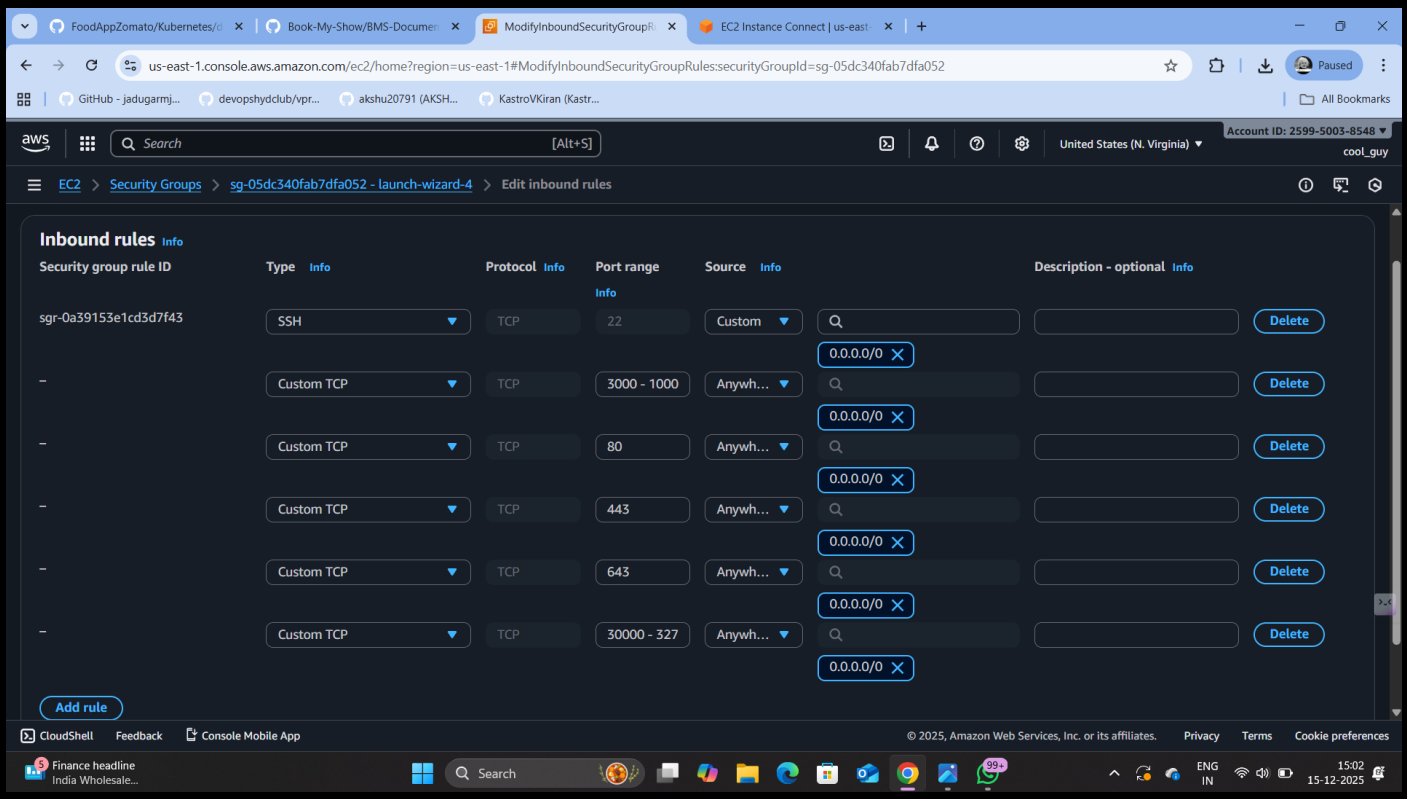
Steps to Deploy the Website successfully

Step-1- First we made an Instance - Zomato-Server (Ubuntu 24.04,t2.large) with 29 GiB storage and switched to root user.

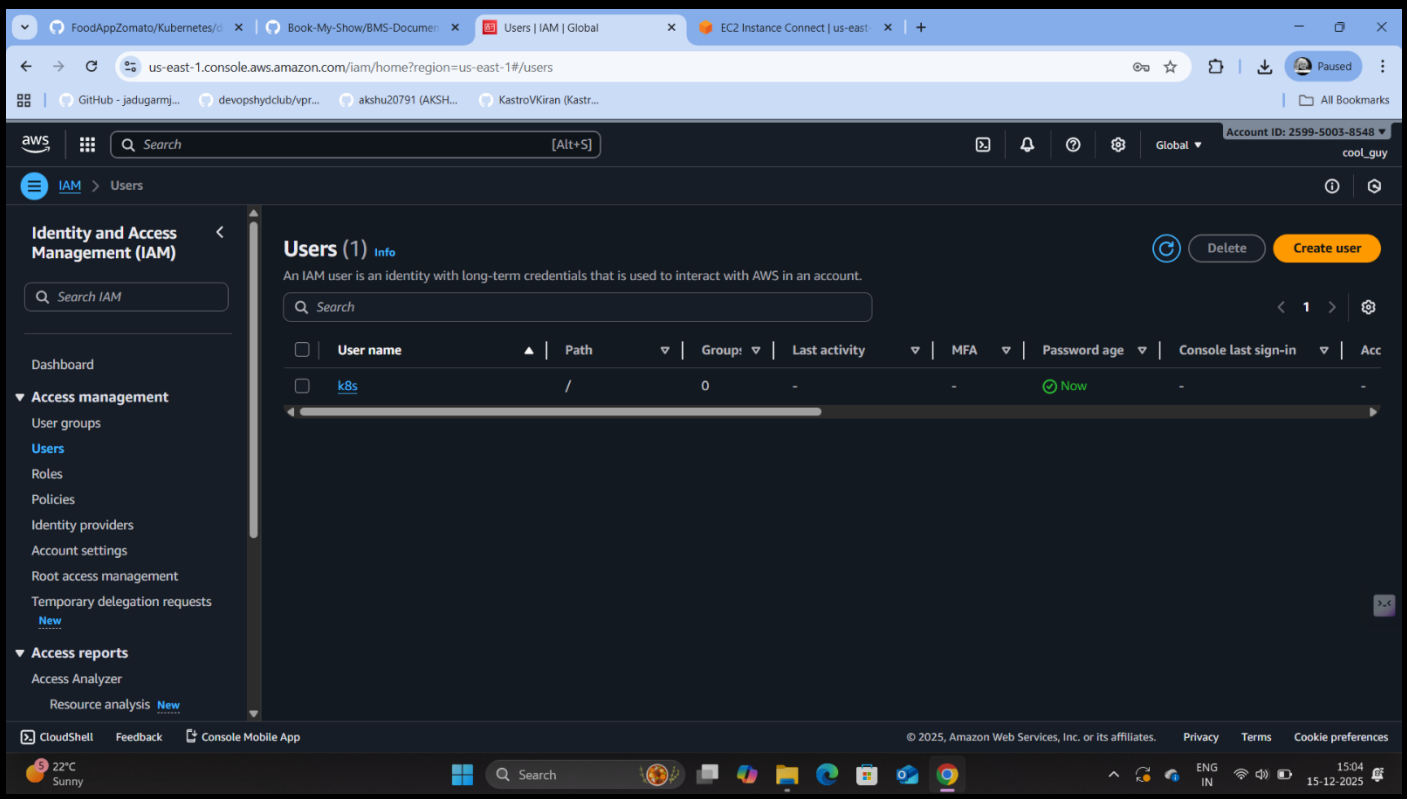




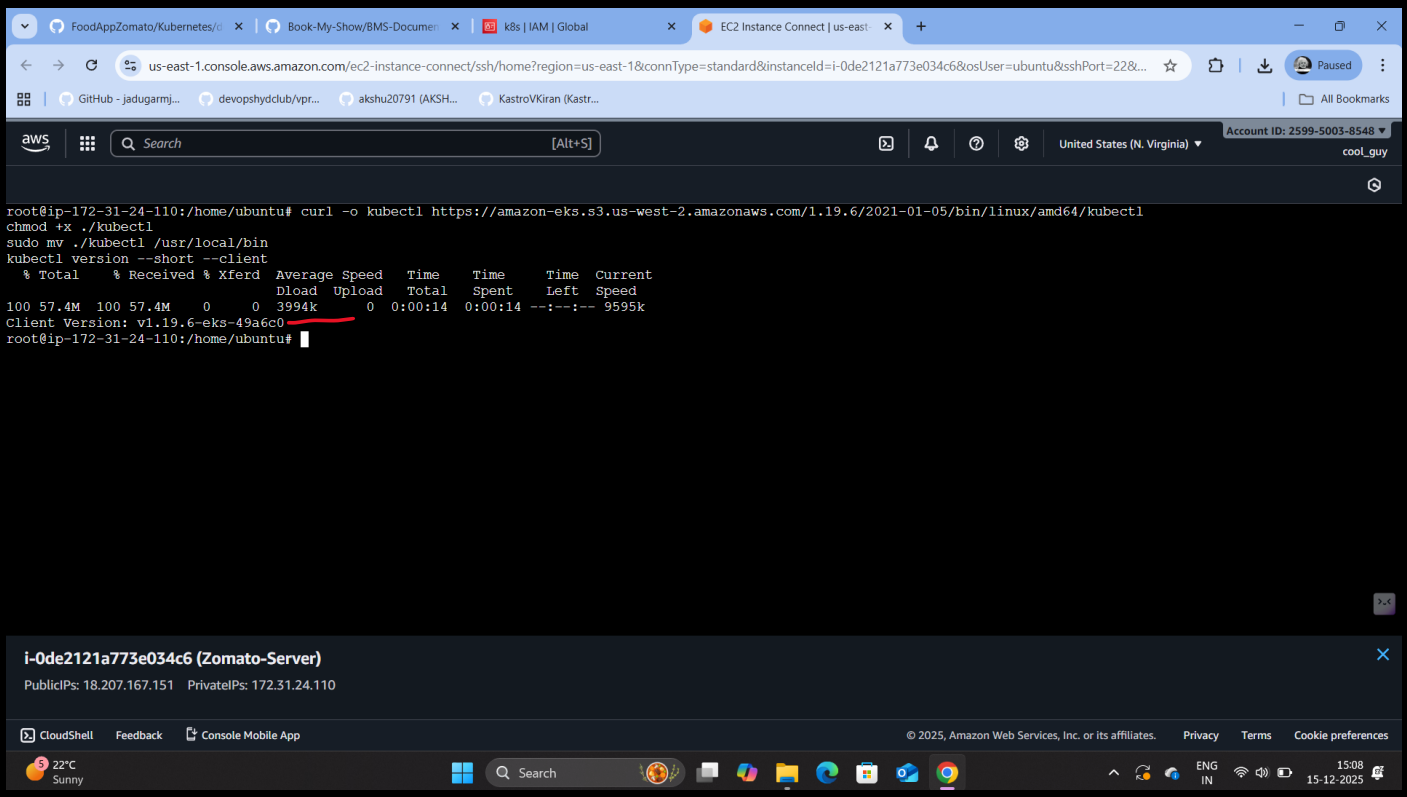
Step-2- We enabled the ports which are below for the Security Group.



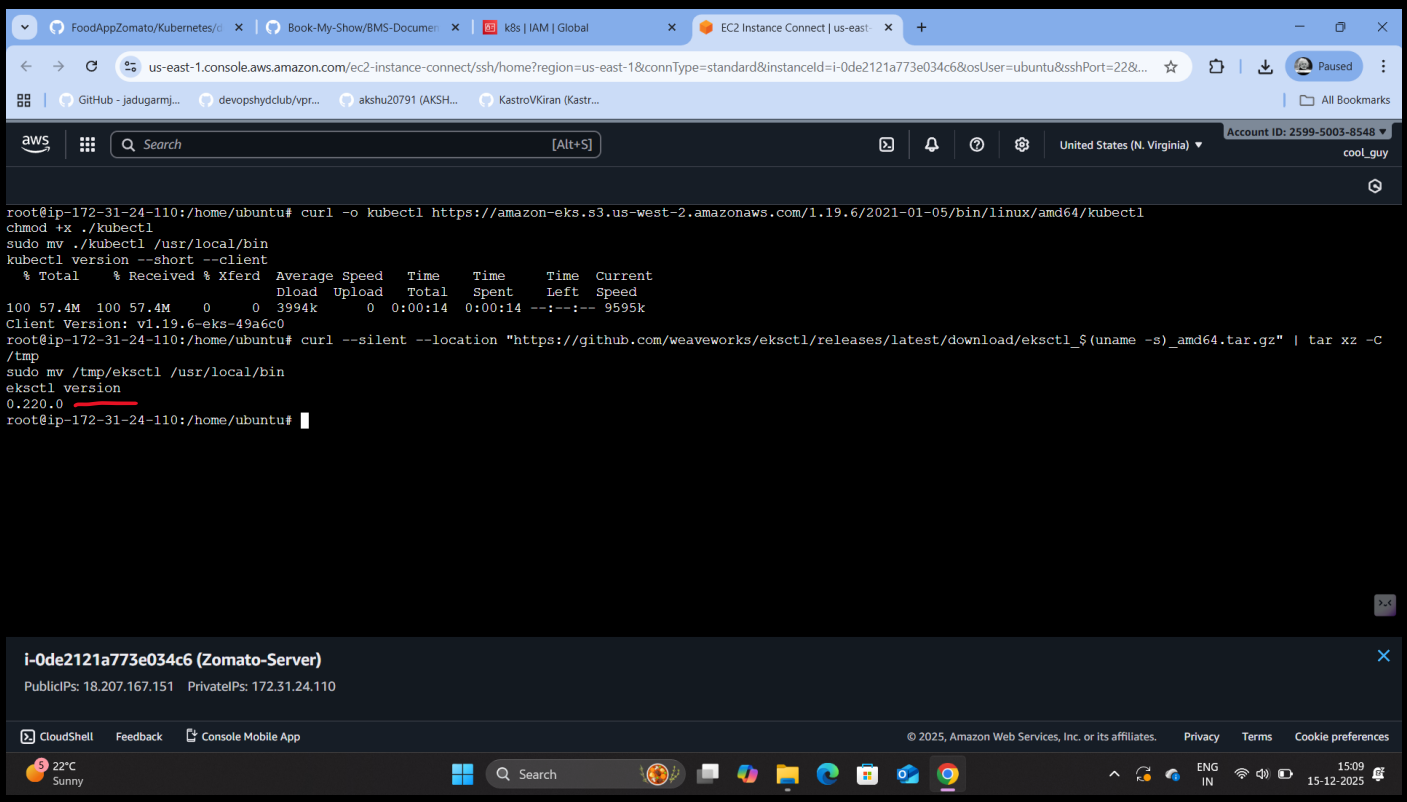
Step-3- We created an IAM User as it is recommended not to use root account for making the clusters.



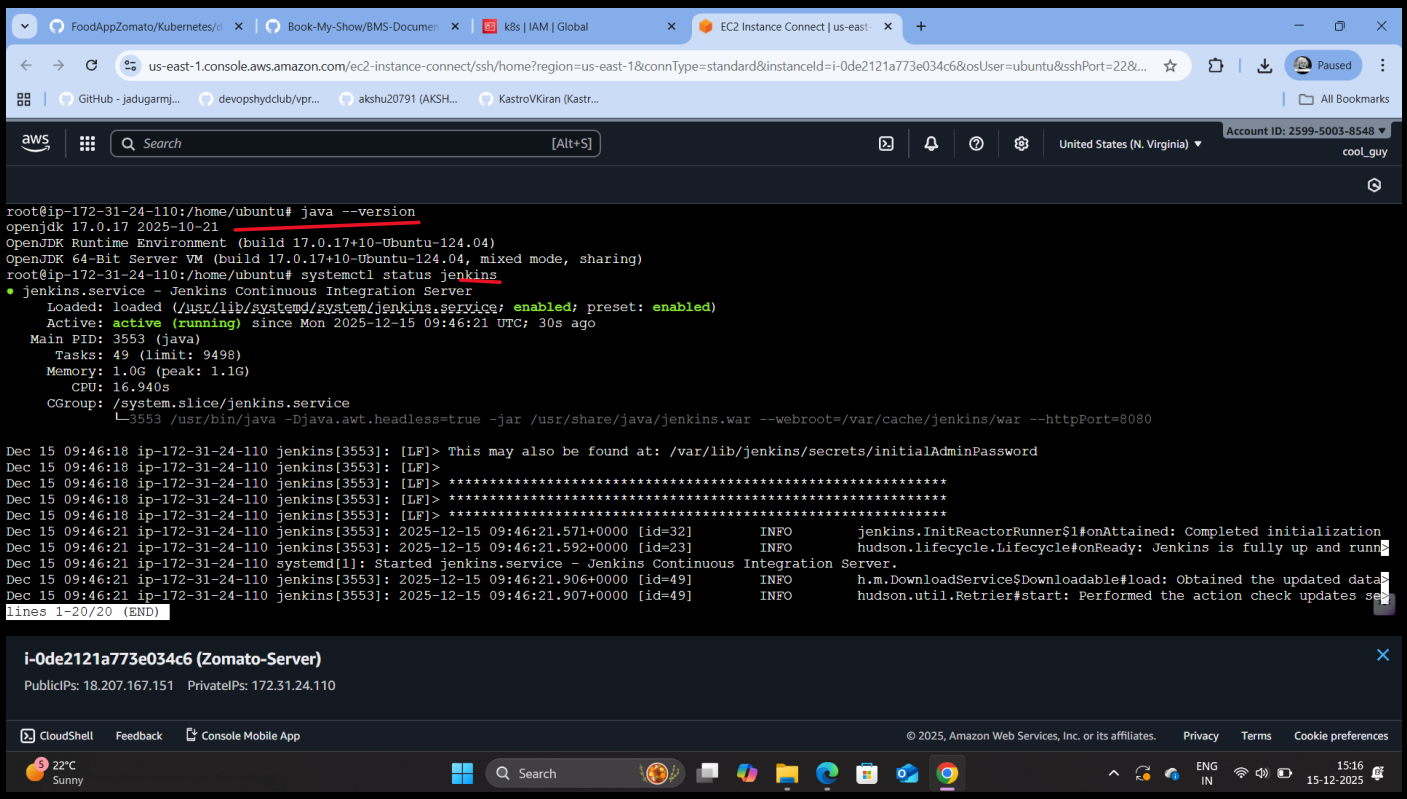
Step-4- We installed kubectl to interact with and manage K8S Cluster.



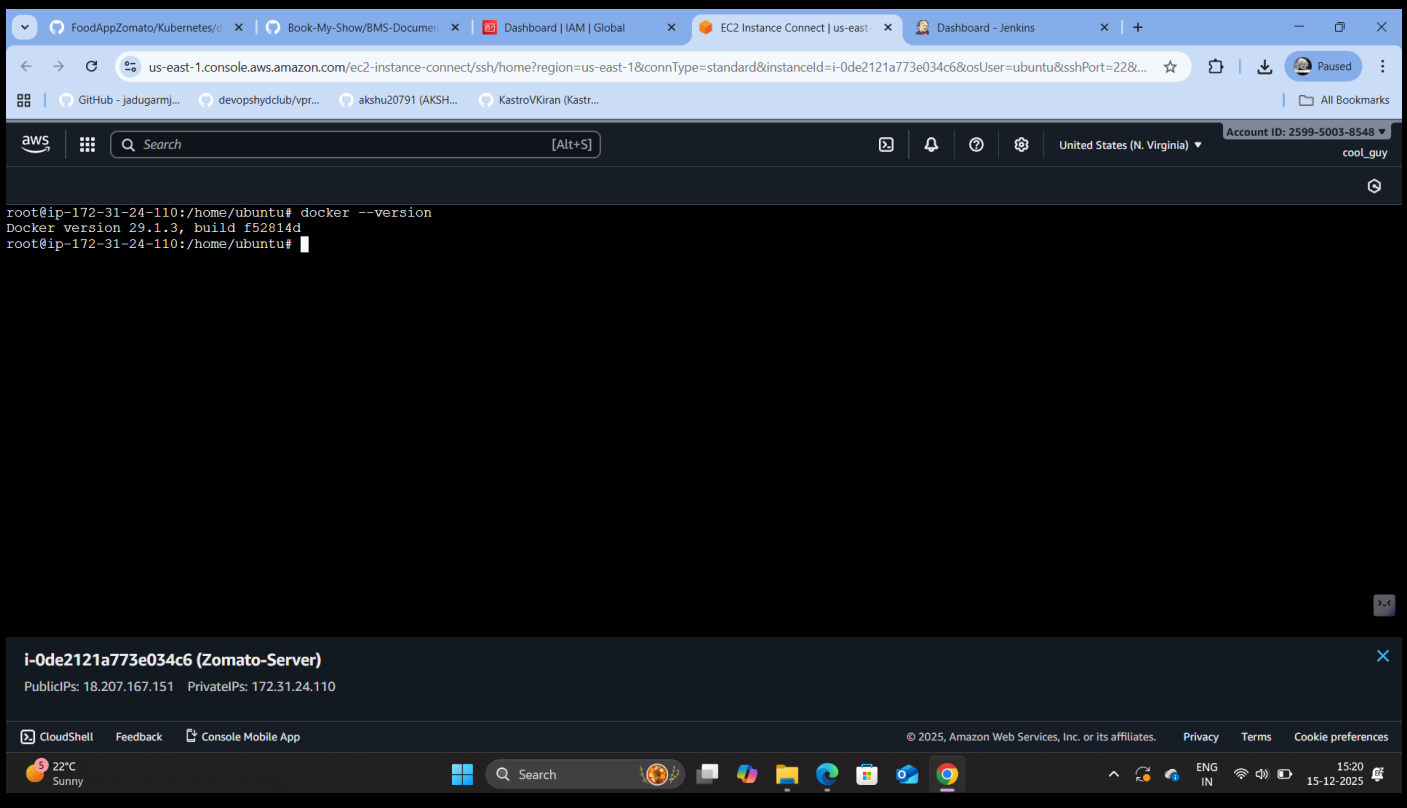
Step-5- Here we installed eksctl to deal with Amazon EKS.



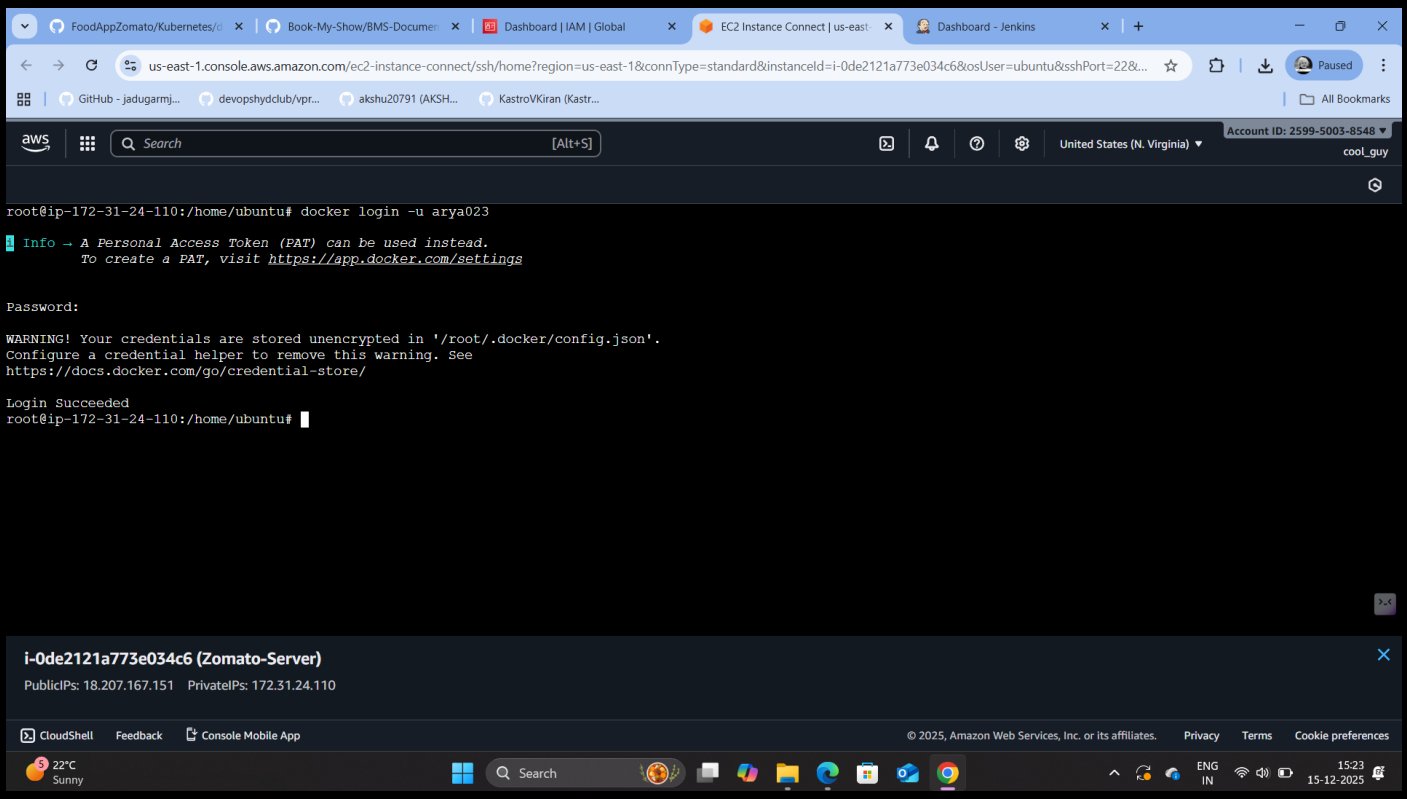
Step-6- We installed java and Jenkins and checked Jenkins's status.



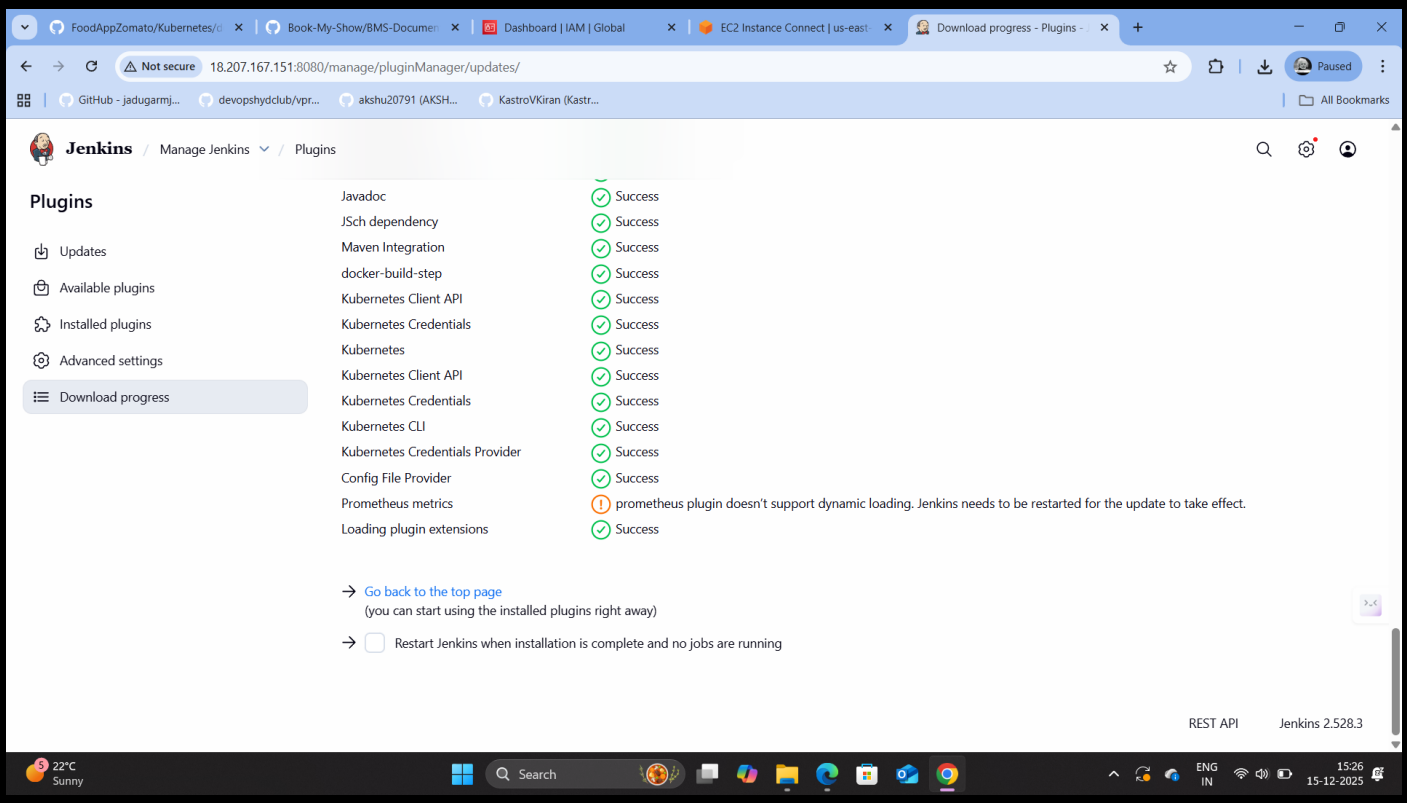
Step-7- We installed docker and configured it.



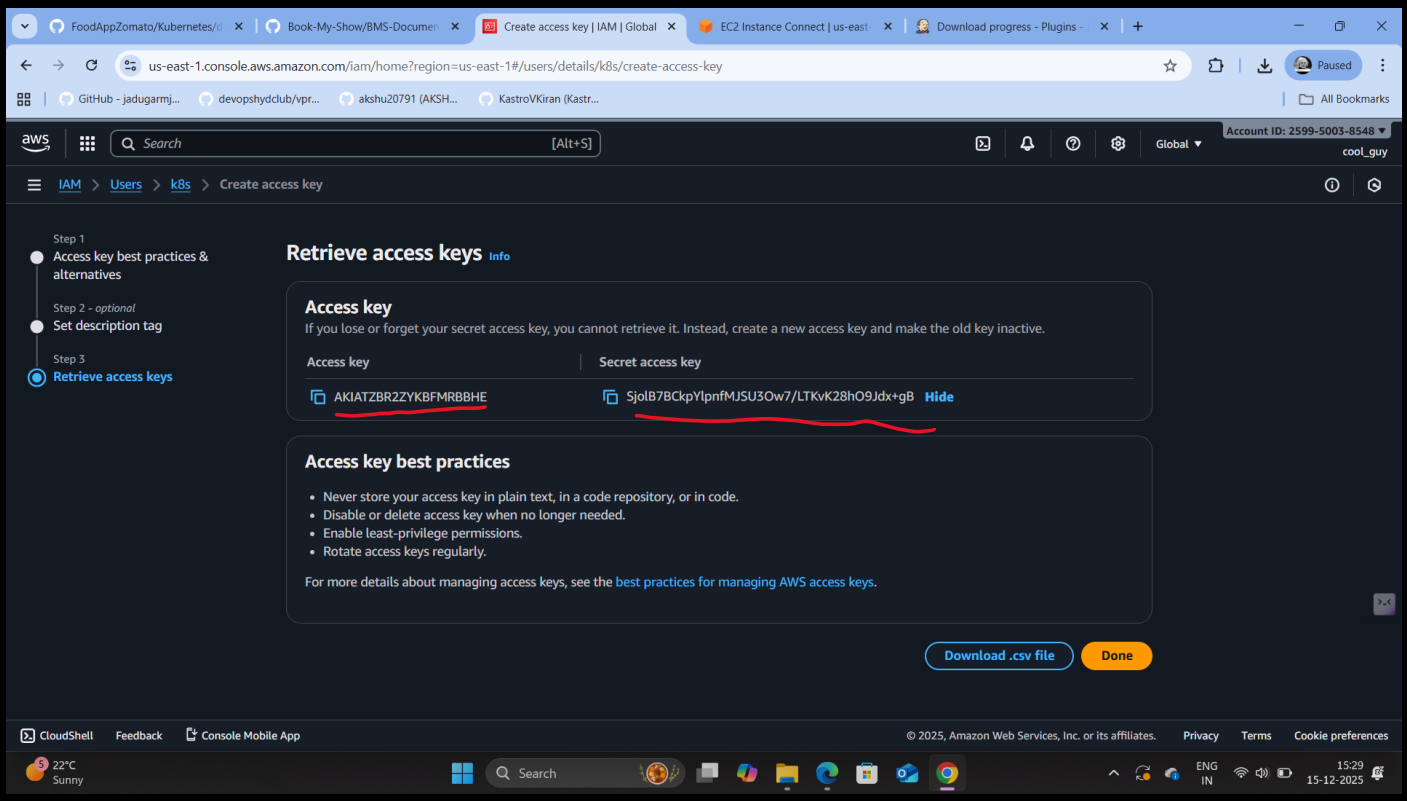
Step-8- We login to the DockerHub account to push the images.



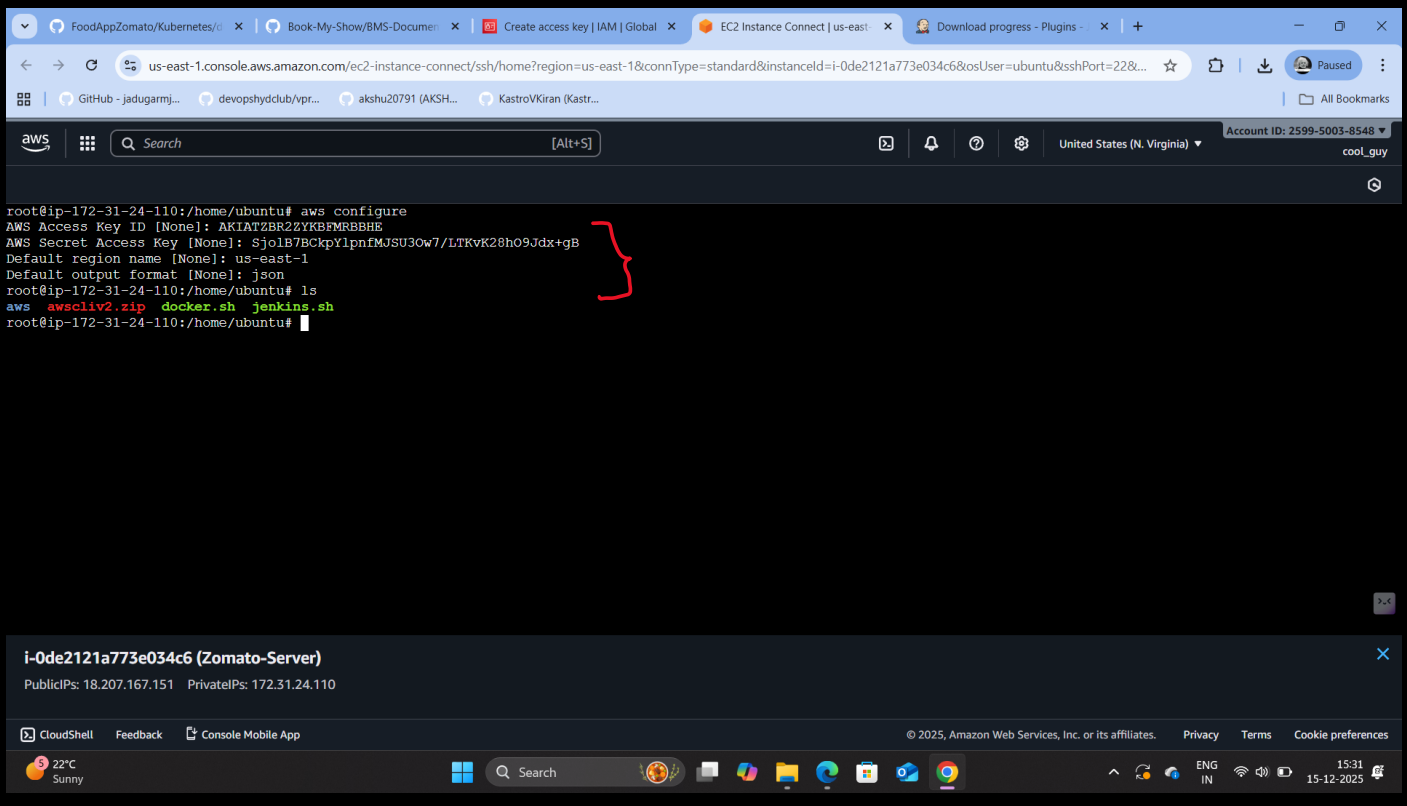
Step-9- In Jenkins we installed some plugins like eclipse, NodeJs, Docker, Kubernetes, Pipeline stage view, Config File Provider, Prometheus.



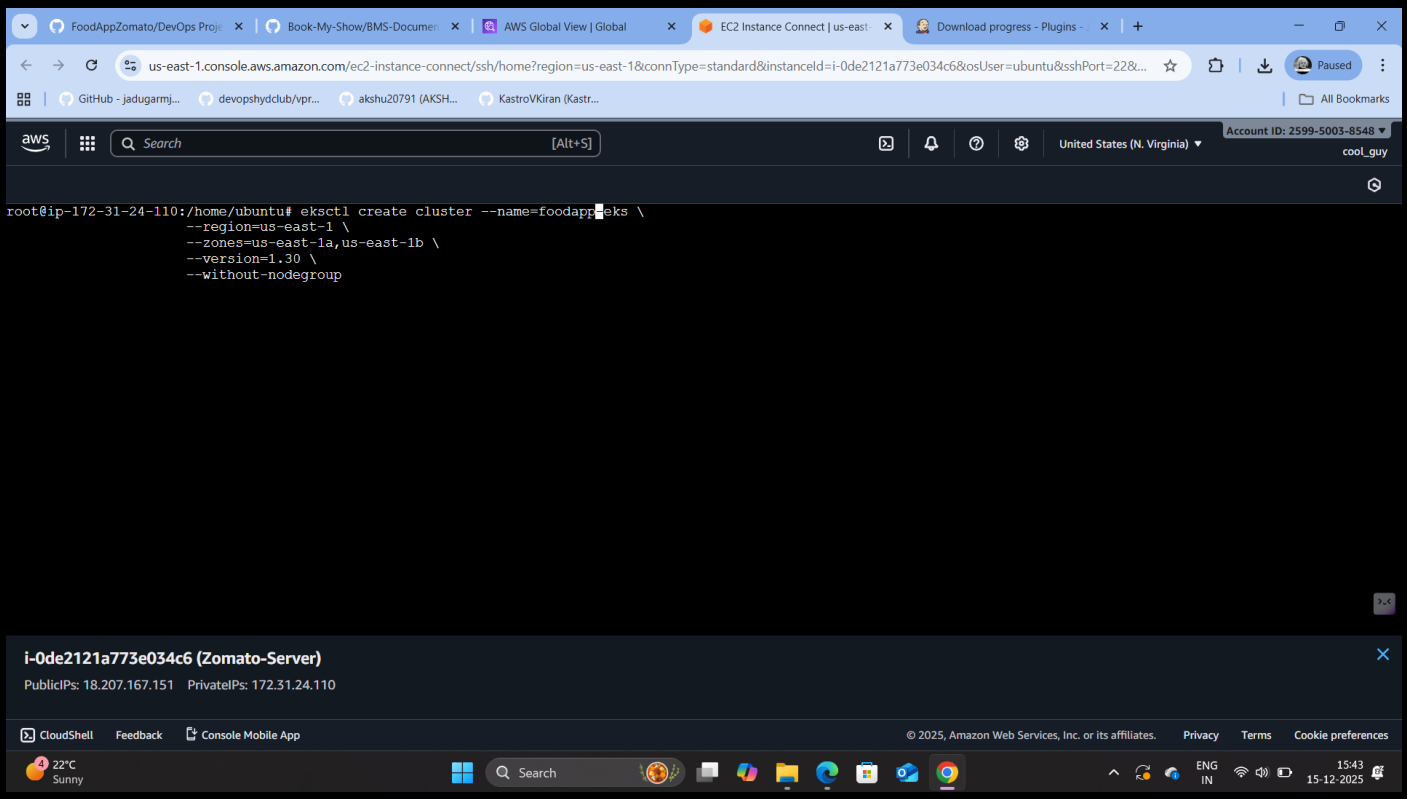
Step-10- We created the access key for the IAM User.



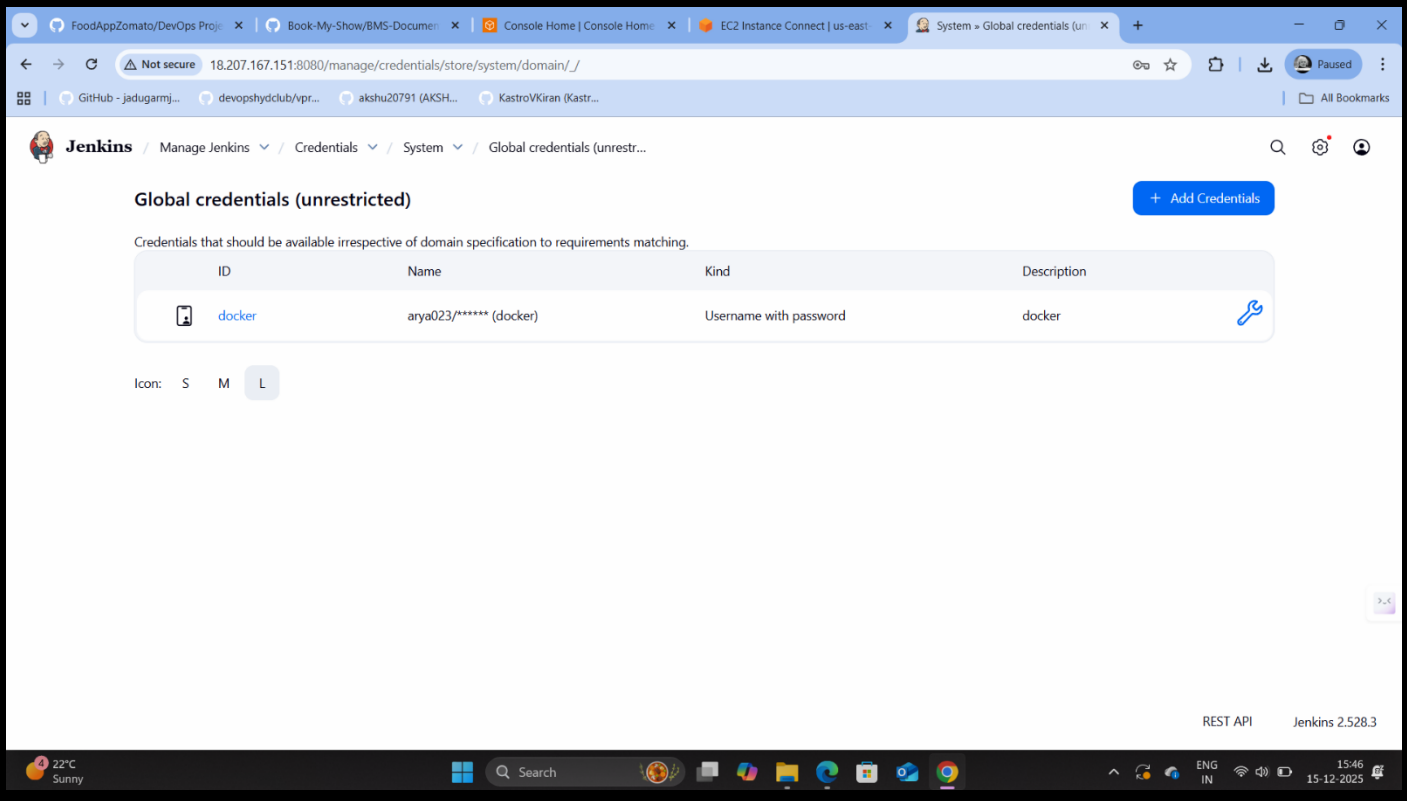
Step-11- We installed AWS CLI to manage AWS services from the terminal.



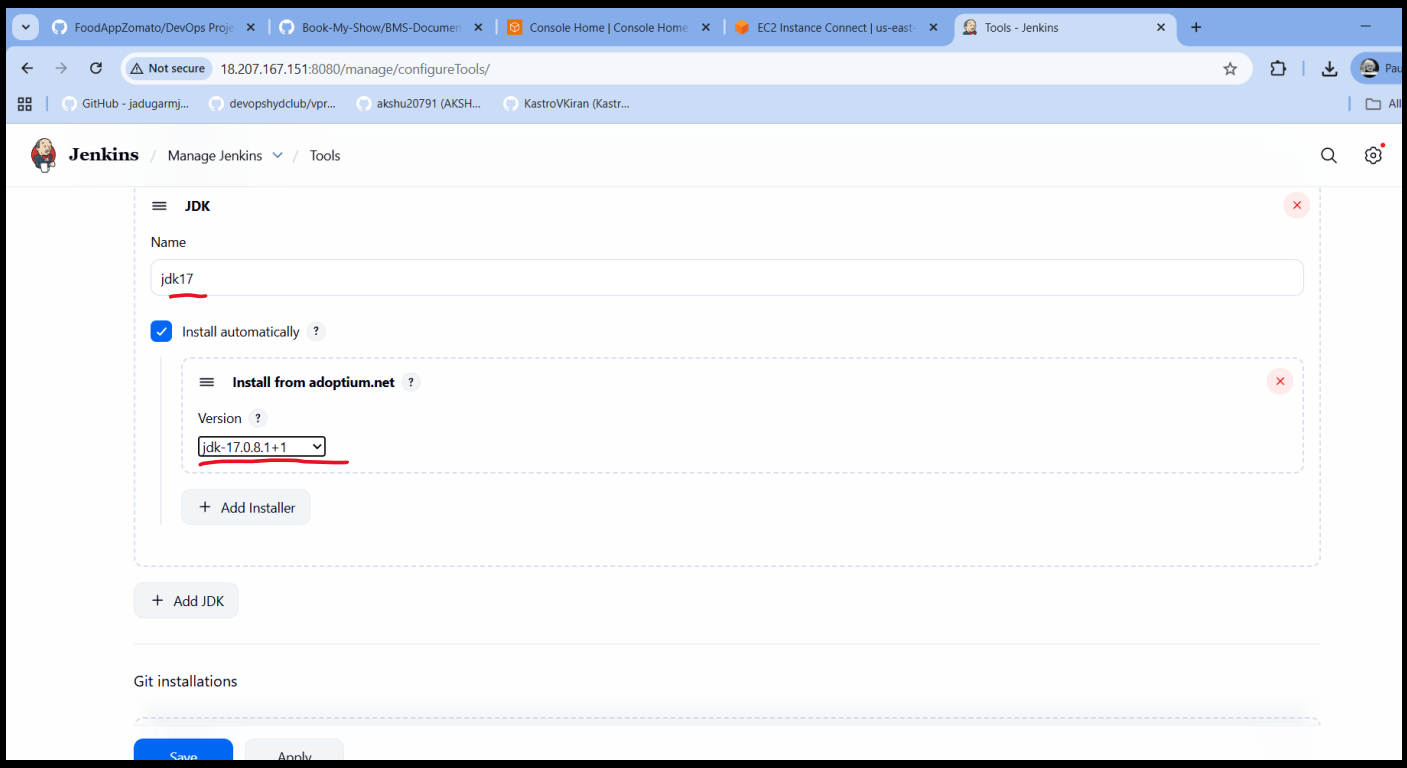
Step-12- Here we give commands to make cluster



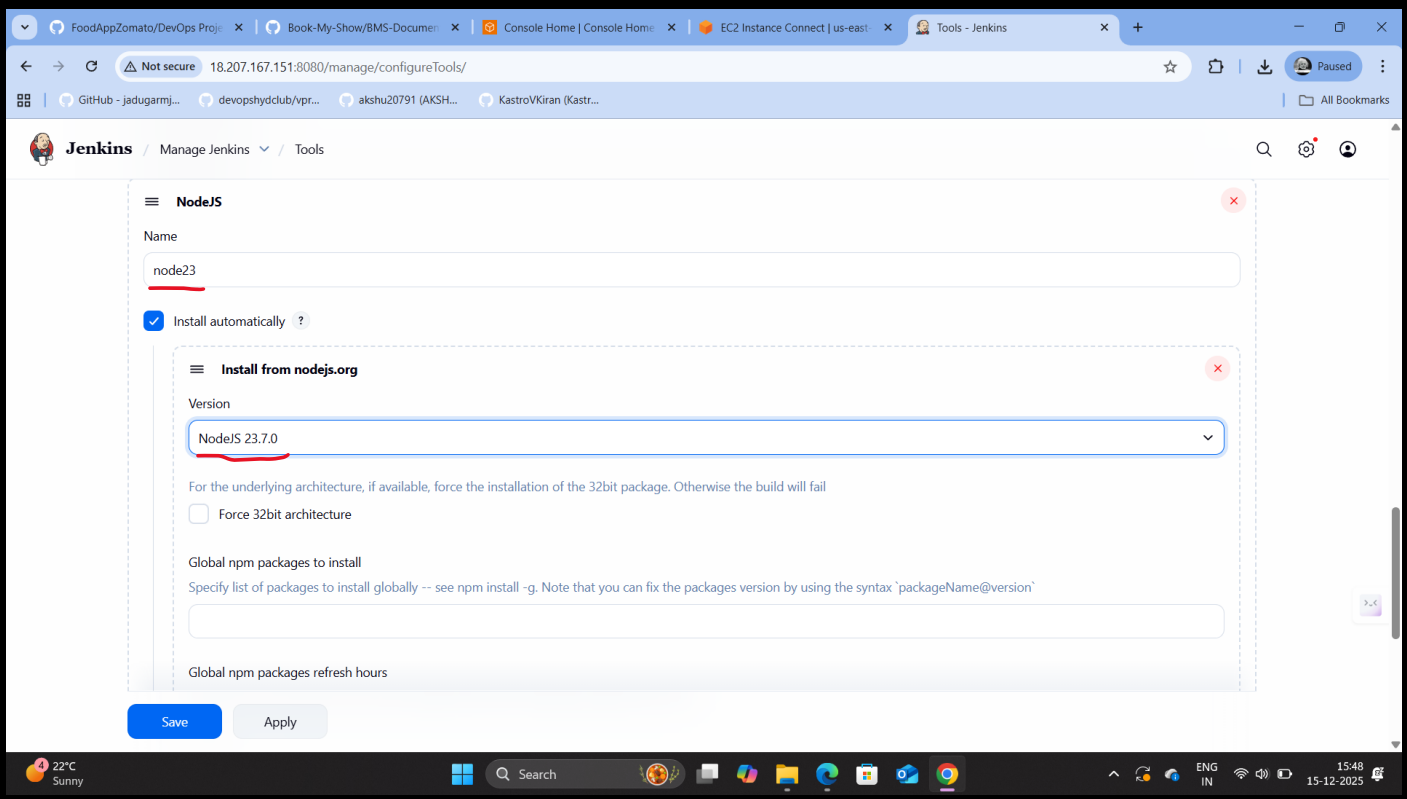
Step-13- We give credentials to Jenkins for docker



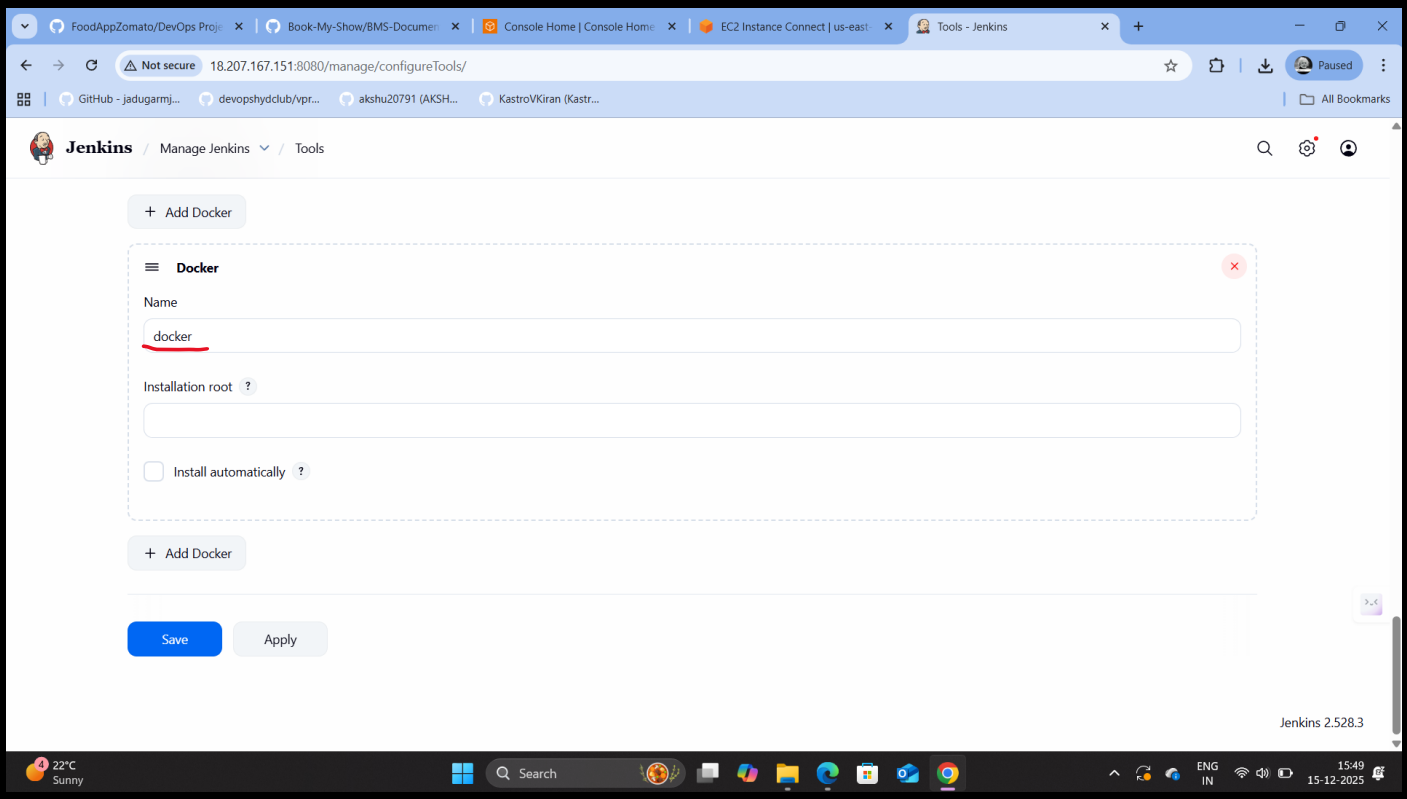
Step-14- We configure tool for java.



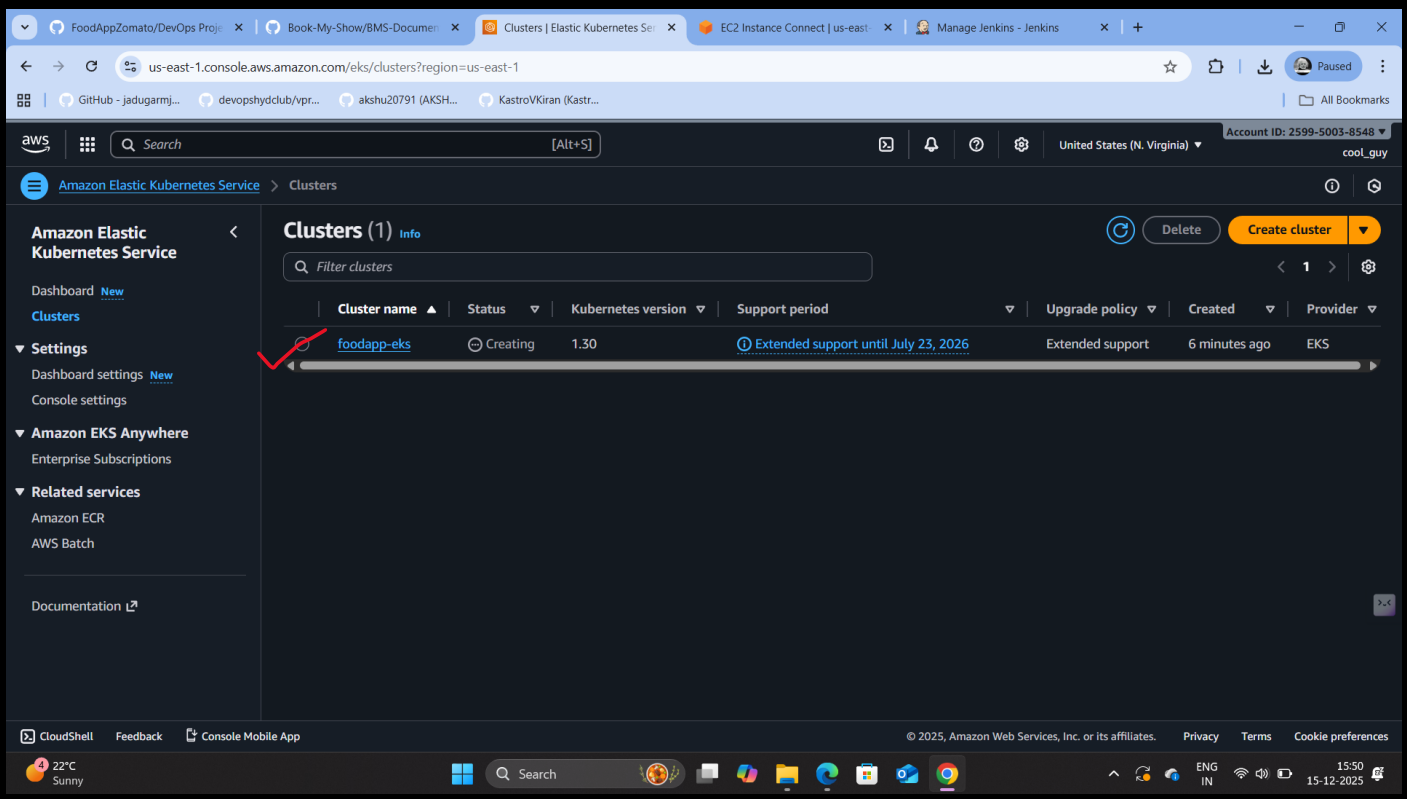
Step-15- We configure tool for NodeJS.



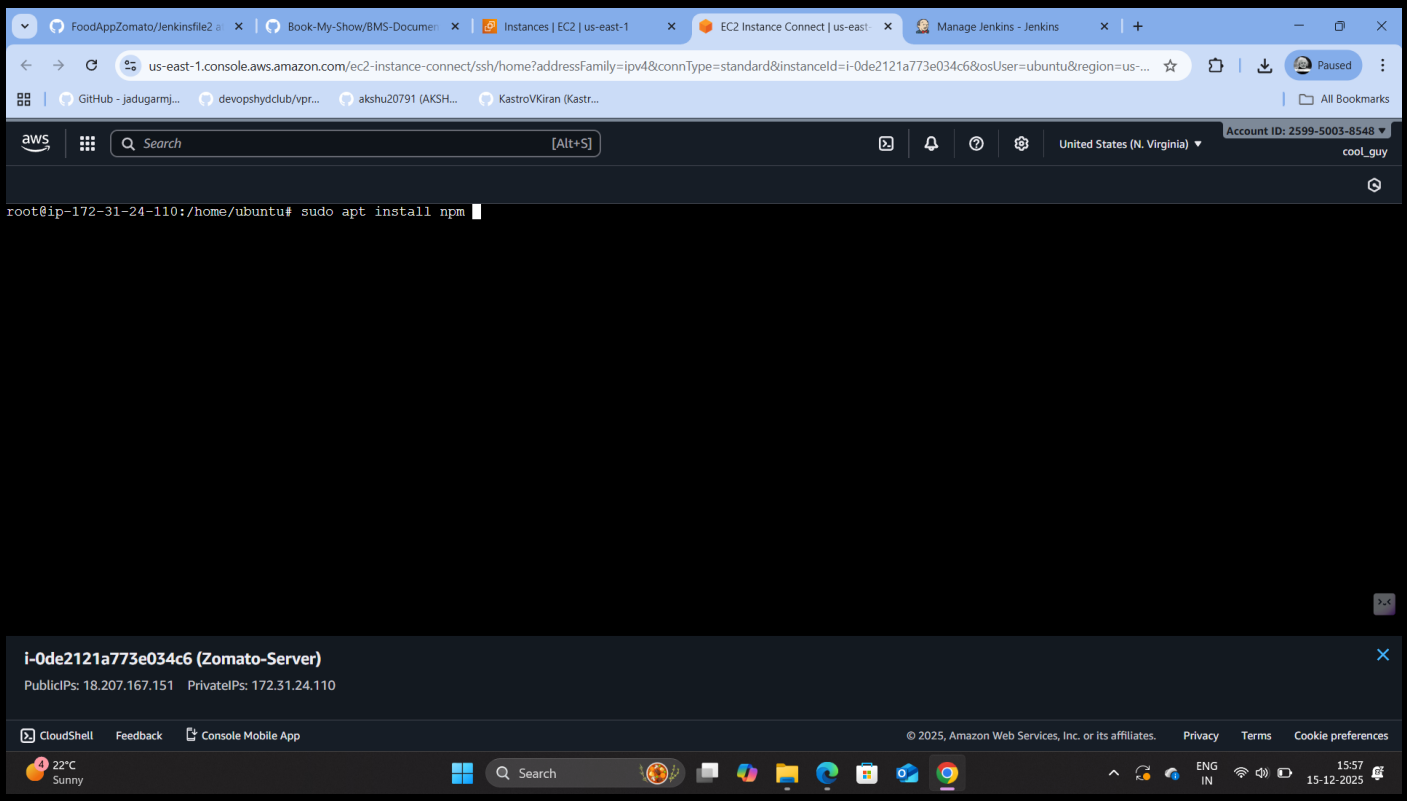
Step-16- We configure tool for Docker.



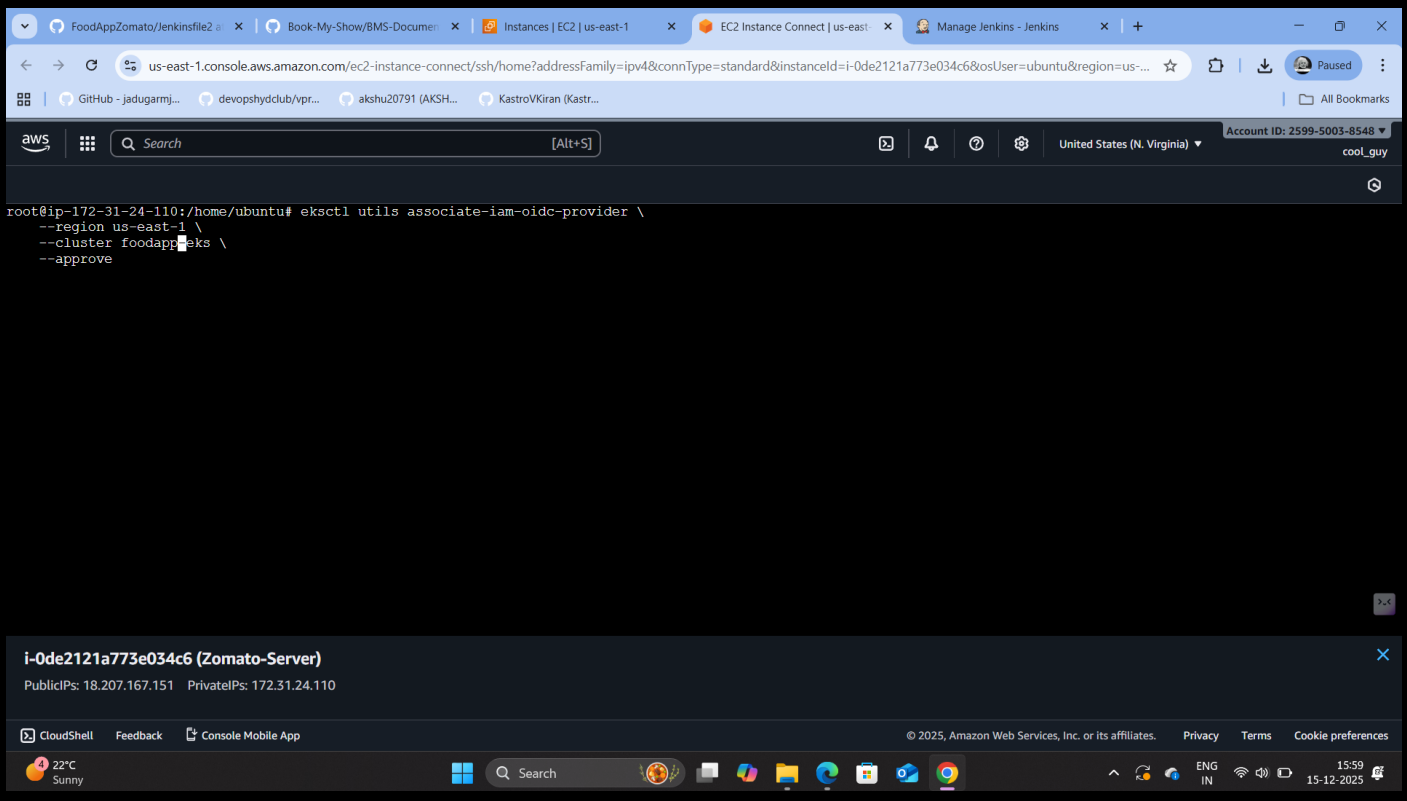
Step-17- We verified on Amazon EKS that foodapp-eks is created or not.



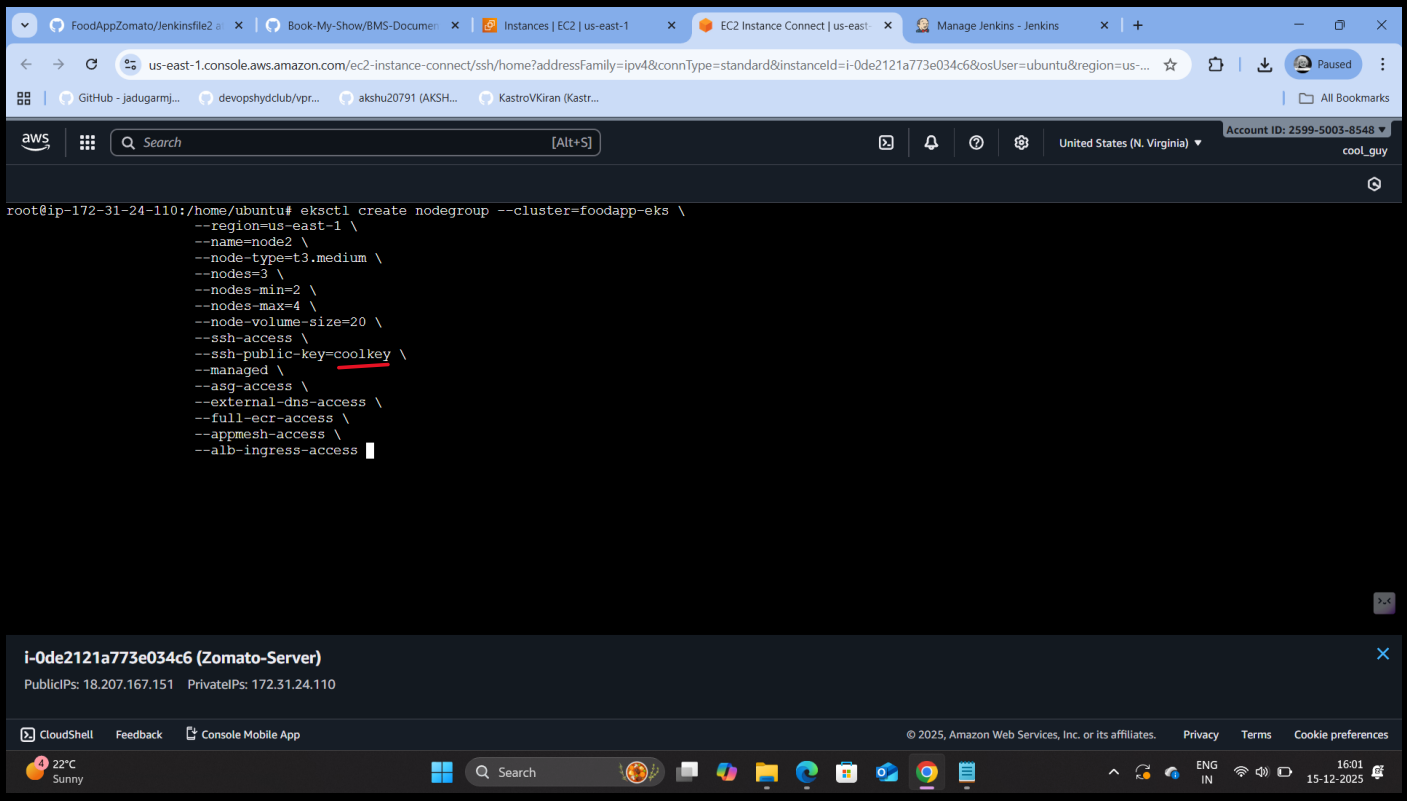
Step-18- We install the npm.



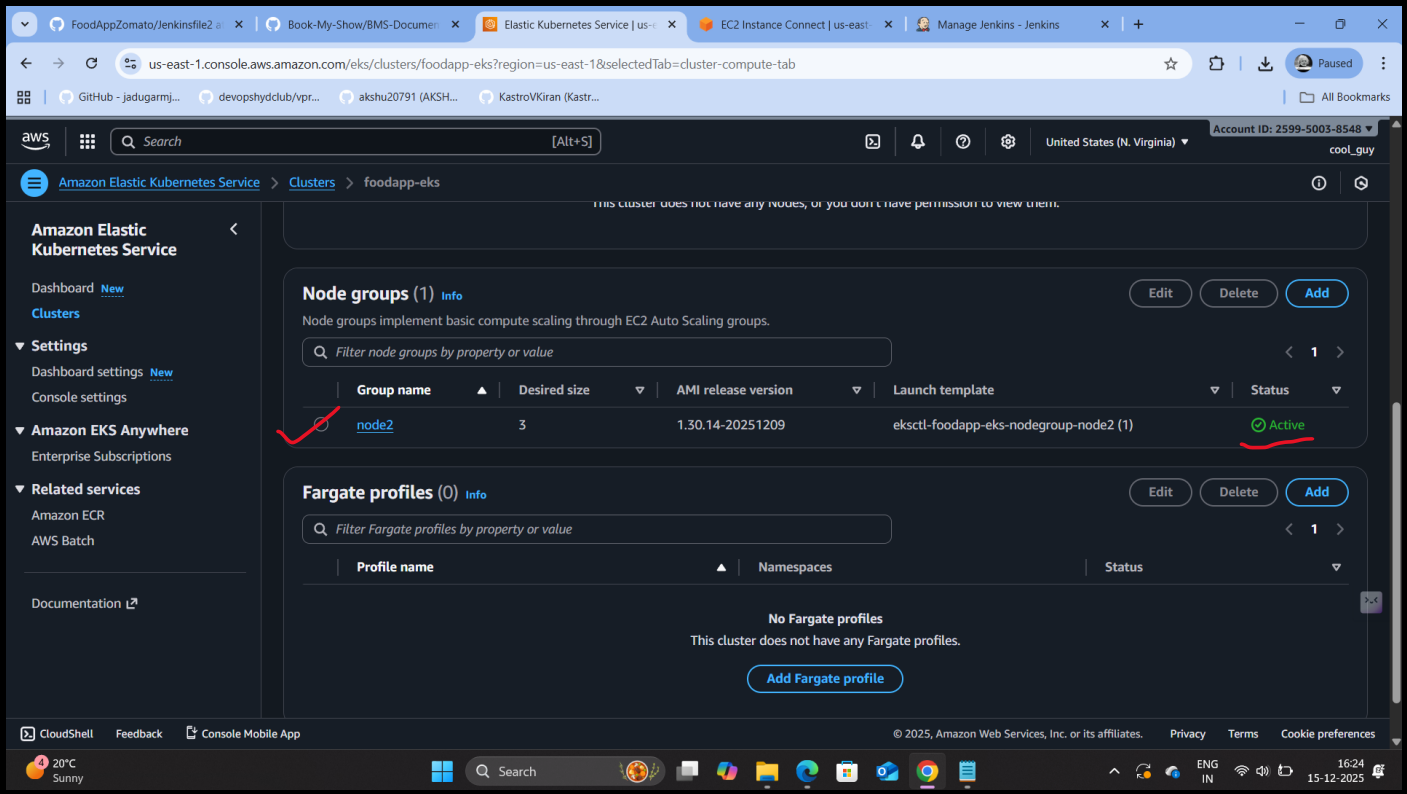
Step-19- Amazon EKS uses OpenID Connect (OIDC) to authenticate Kubernetes service accounts with IAM roles.



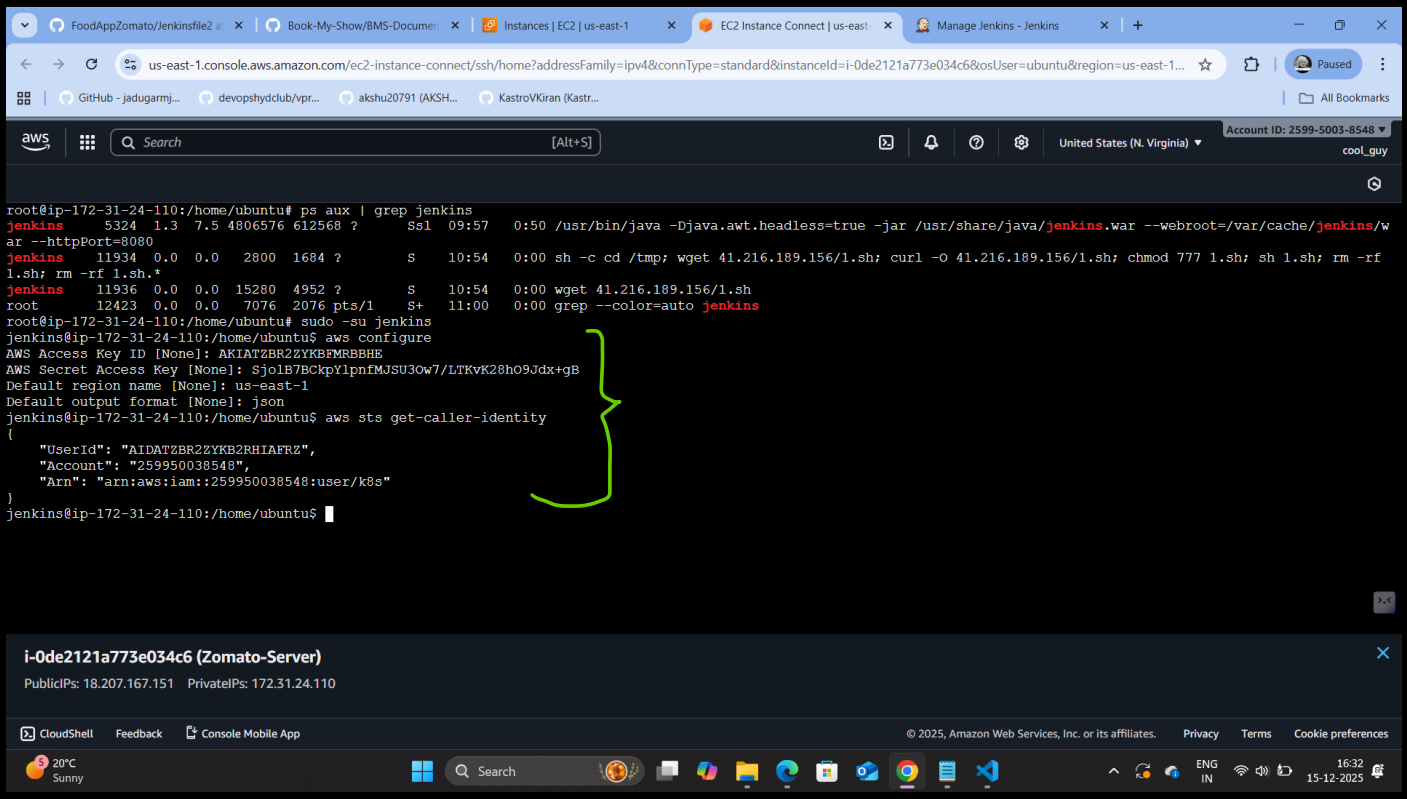
Step-20- Here we will create the nodegroup, in ssh-public-key just give name of the key (not .pem file name)



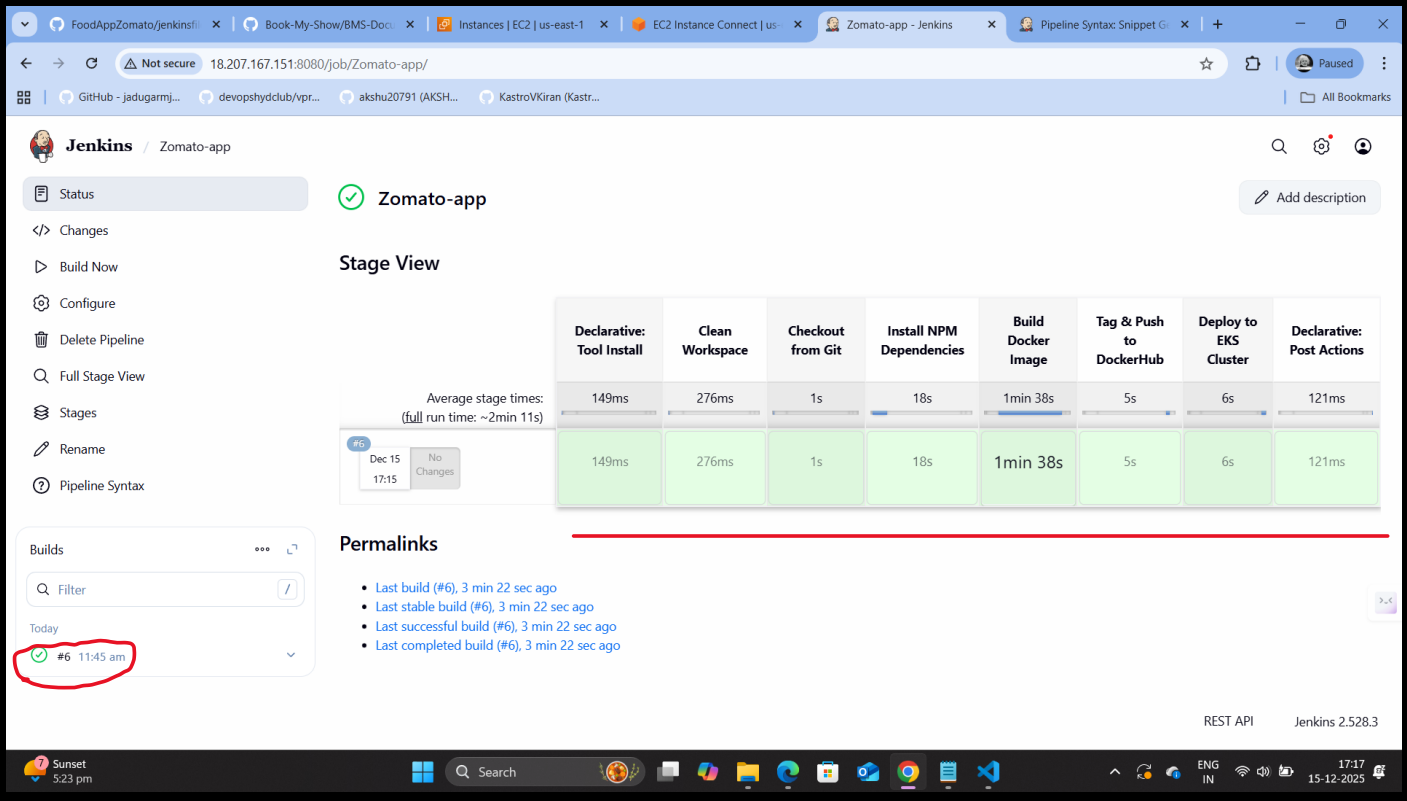
Step-21- Here we can see that the node group is created.



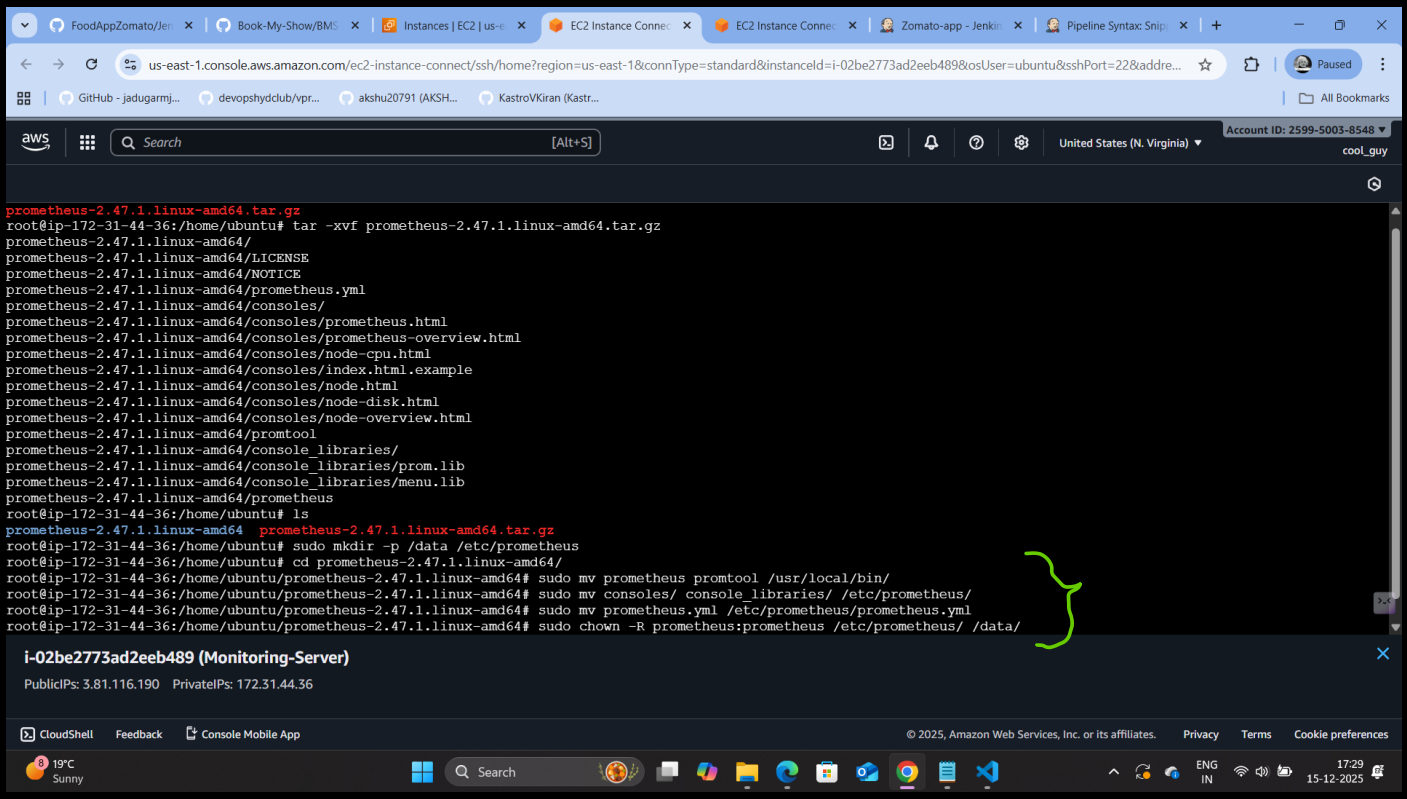
Step-22- Here we configure AWS in Jenkins user.



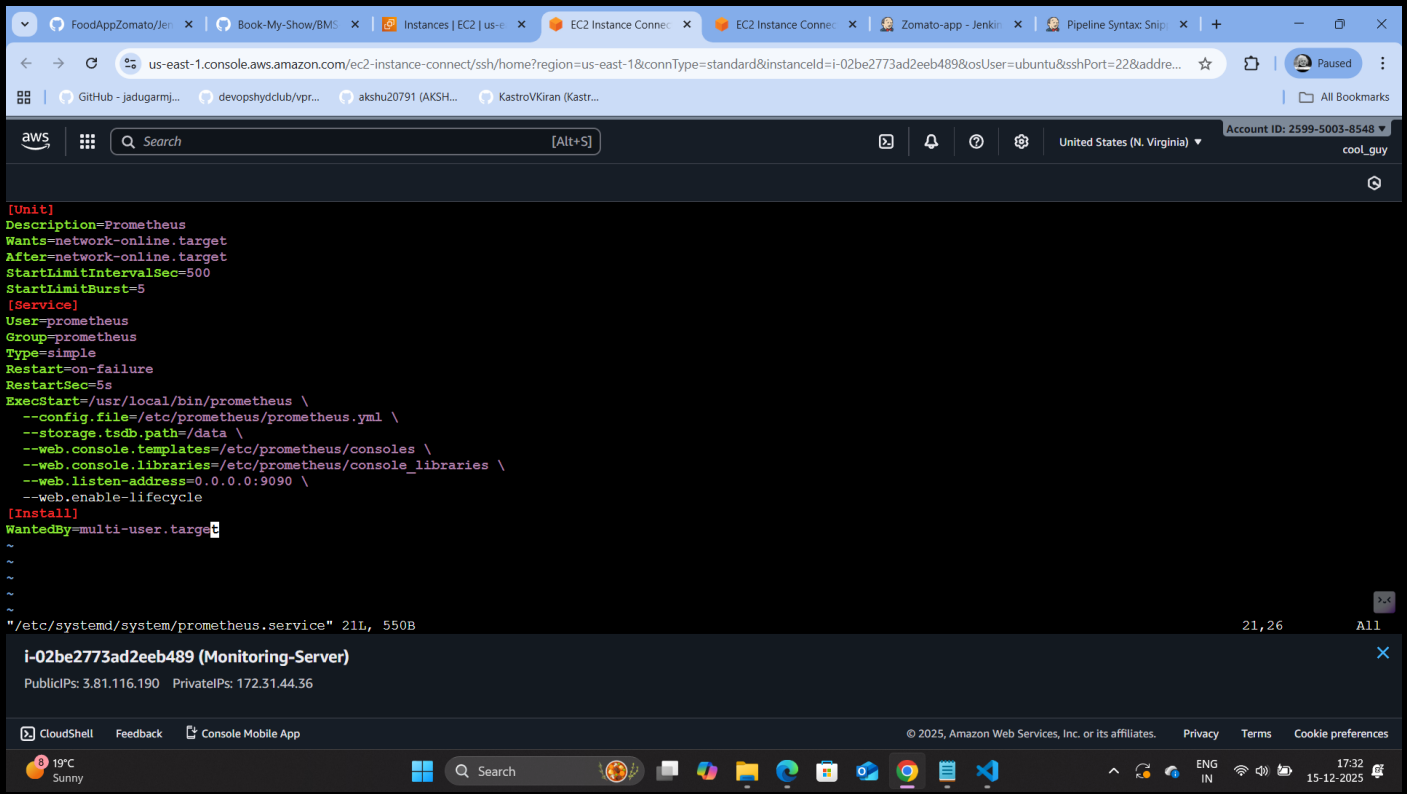
Step-23- Here we can see that our application is successfully deployed to the EKS cluster



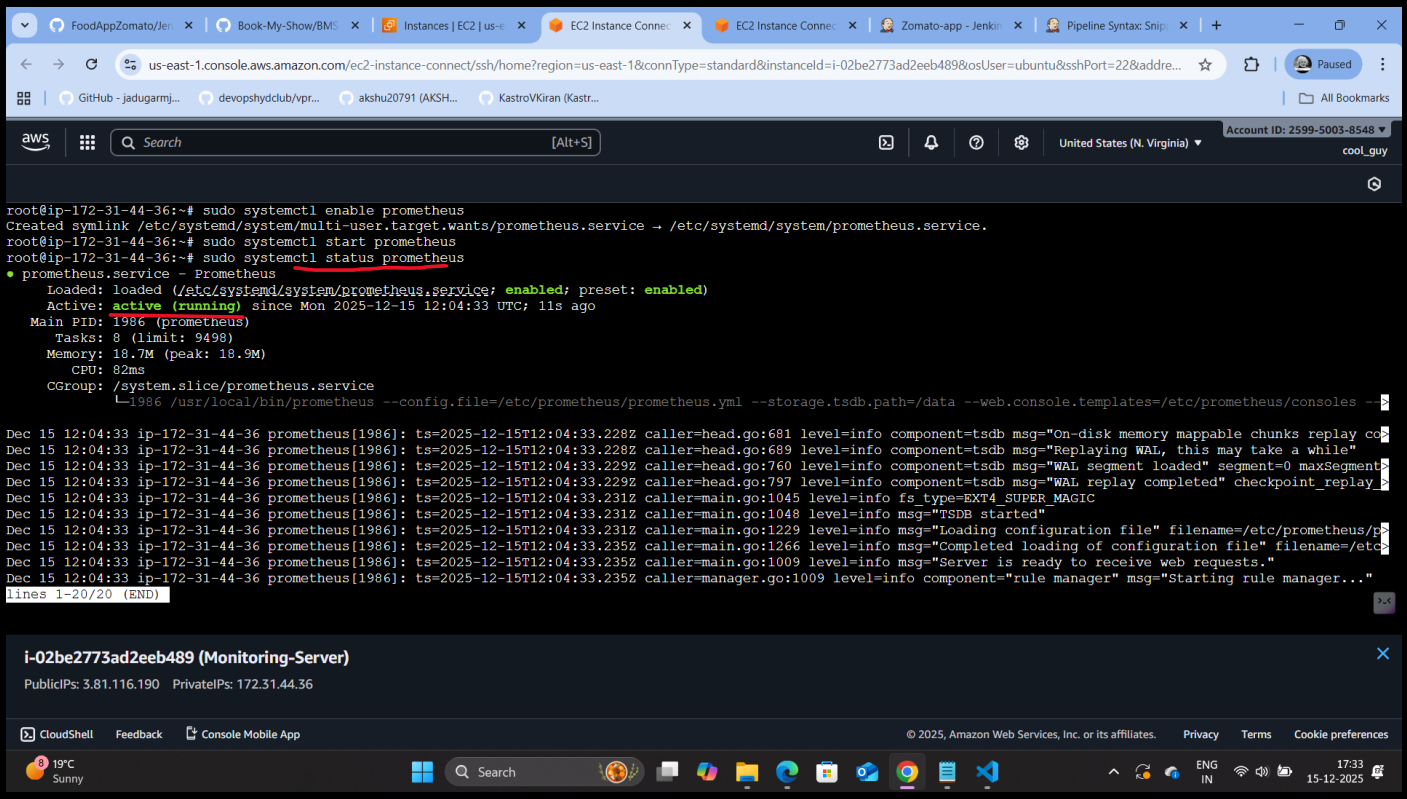
Step-24- Here we installed Prometheus and made some changes in the directories as below.



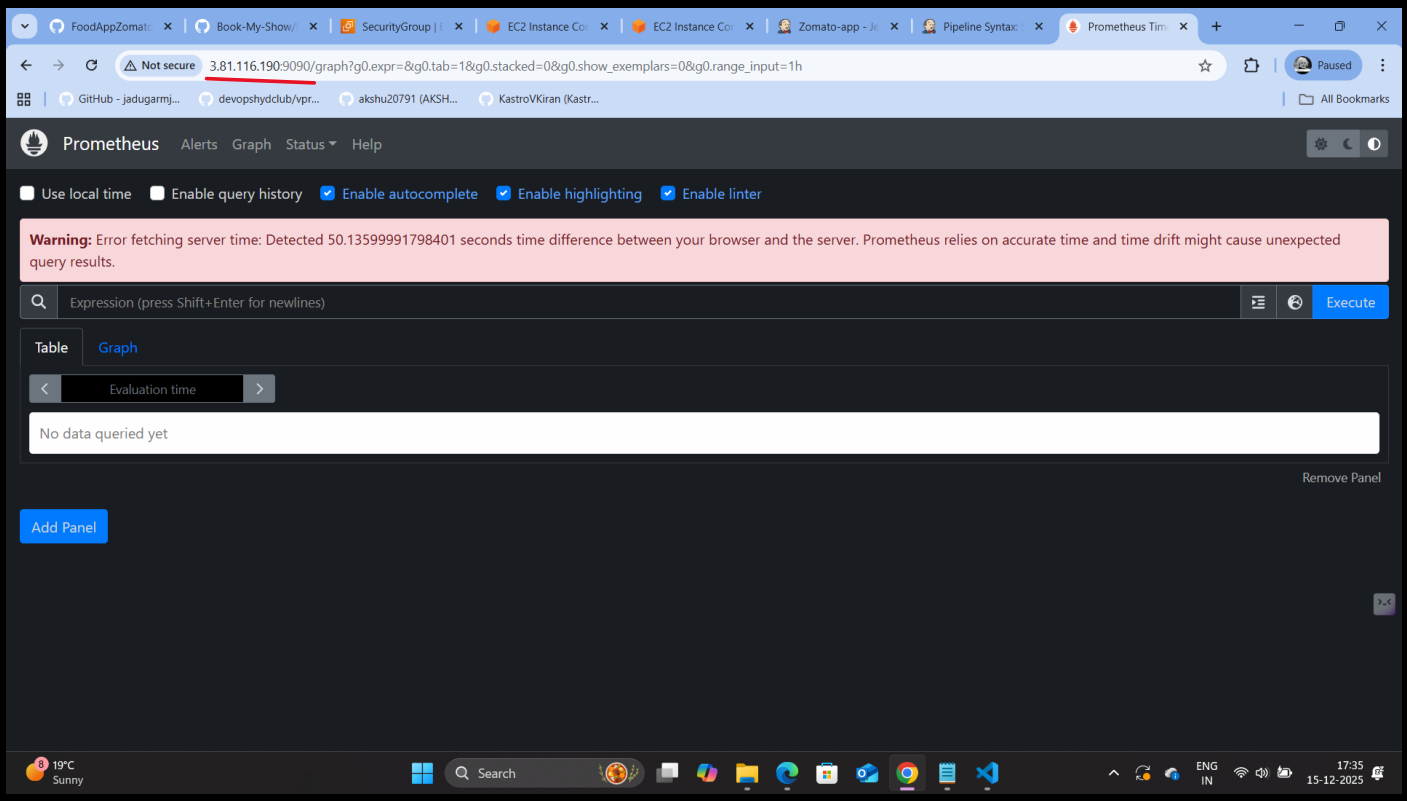
Step-25- We created a systemd unit configuration file.



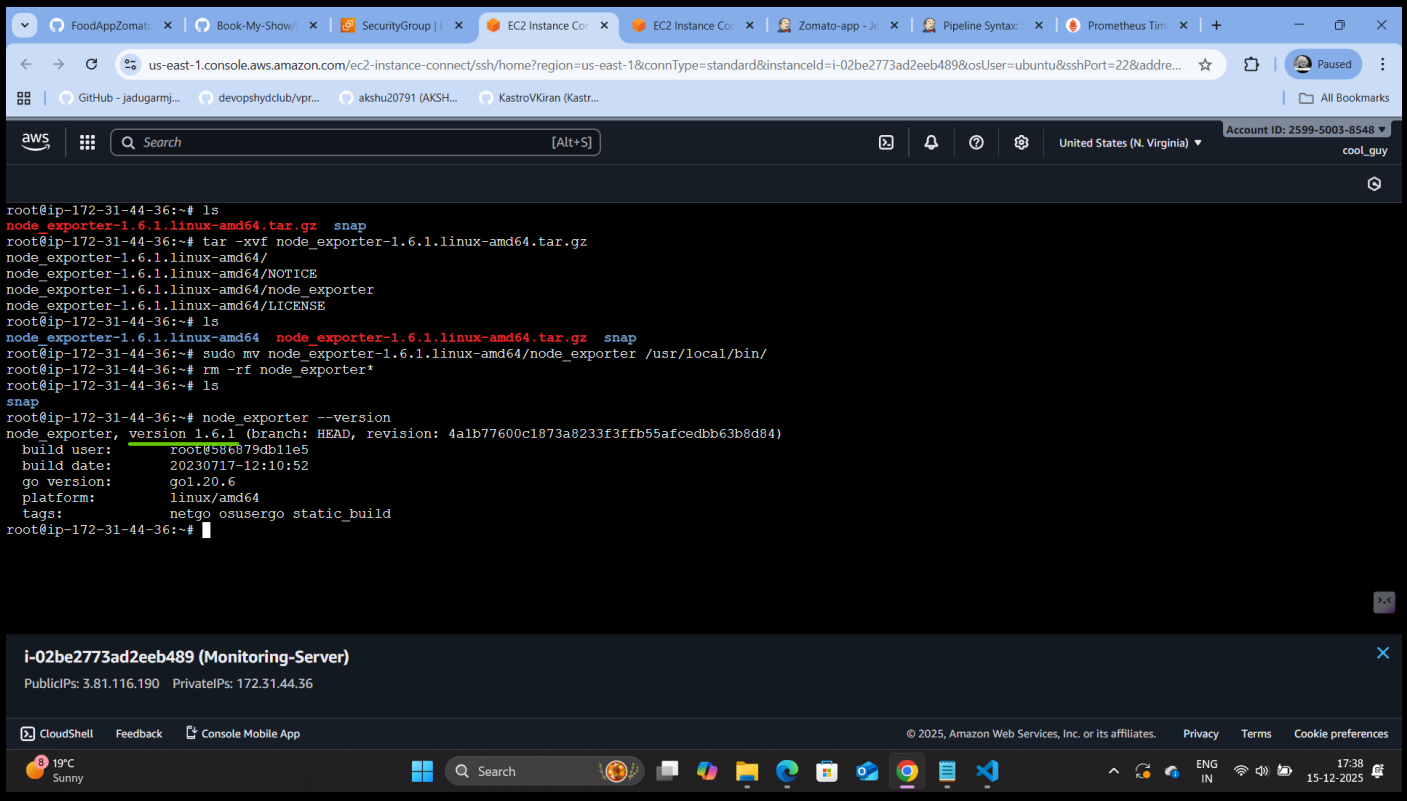
Step-26- We checked the status of Prometheus and its running fine.



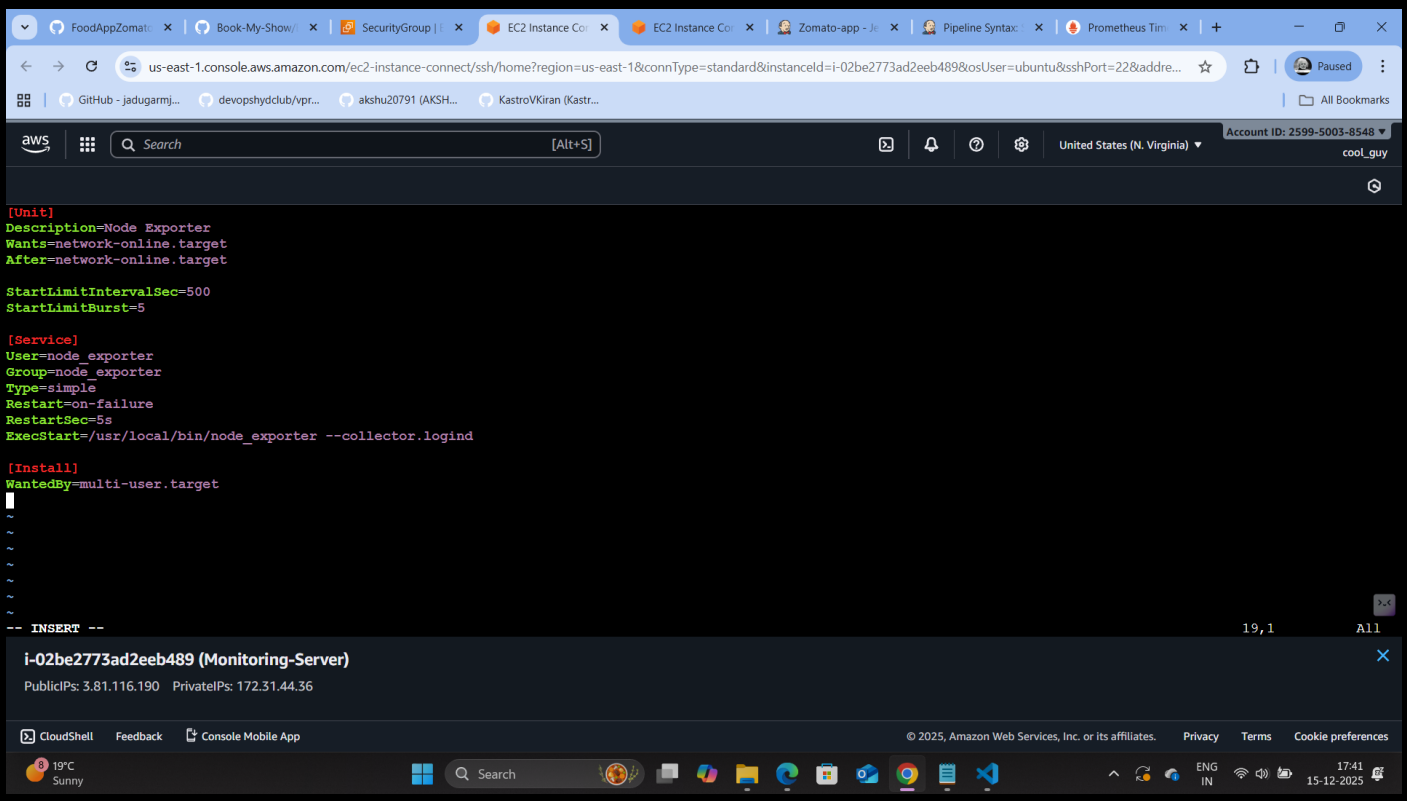
Step-27- we opened Prometheus with <Monitoring Server IP> : 9090



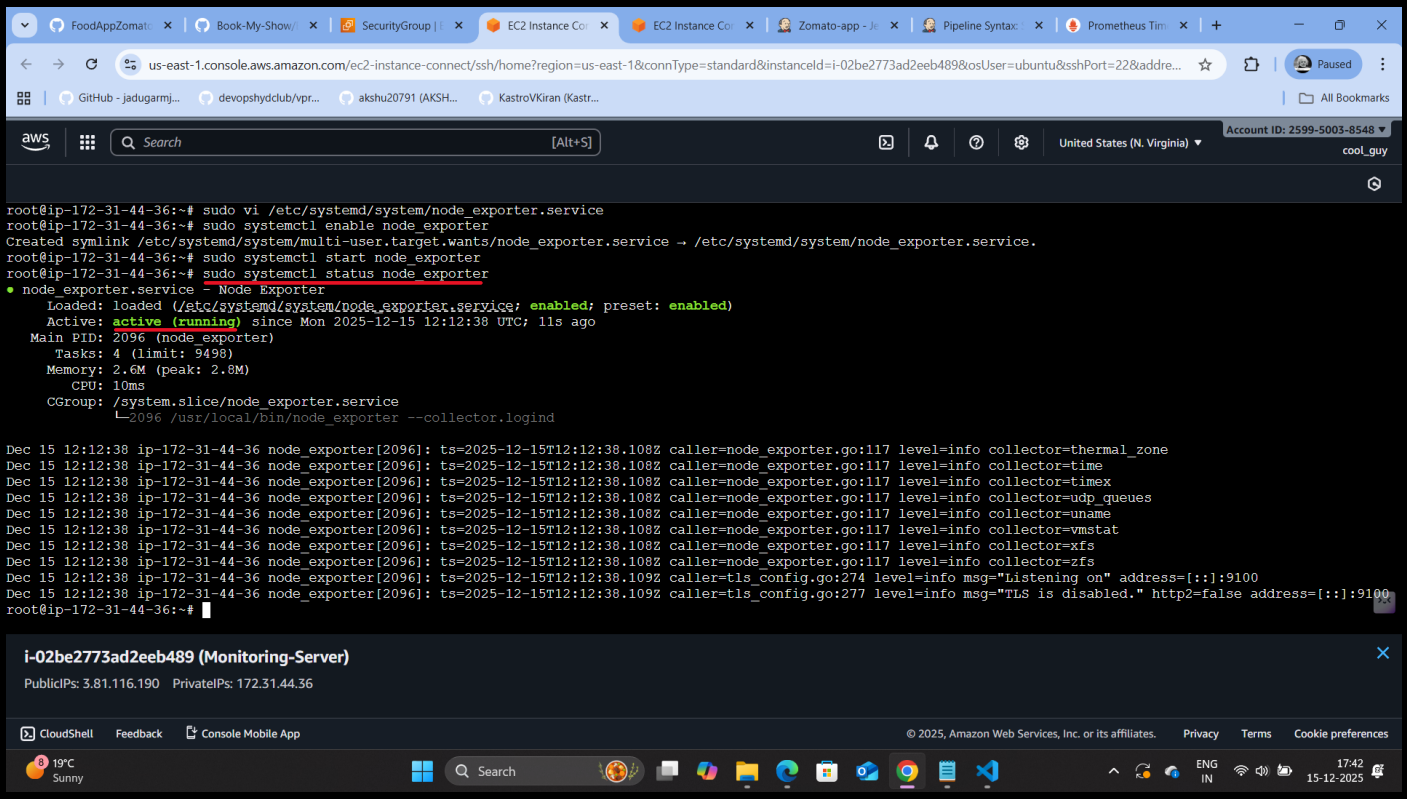
Step-28- Now we installed node exporter and checked its version.



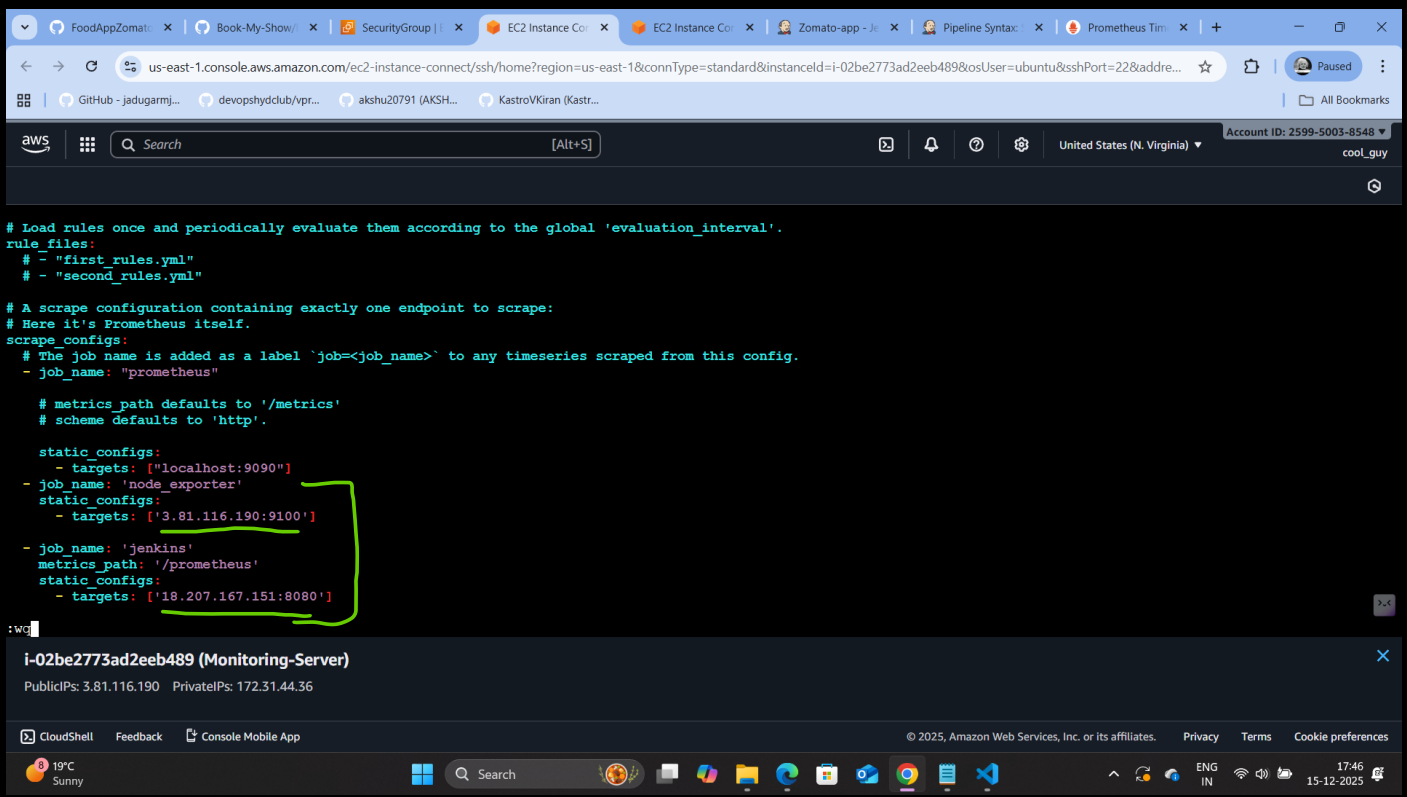
Step-29- We created a systemd unit configuration file for node exporter.



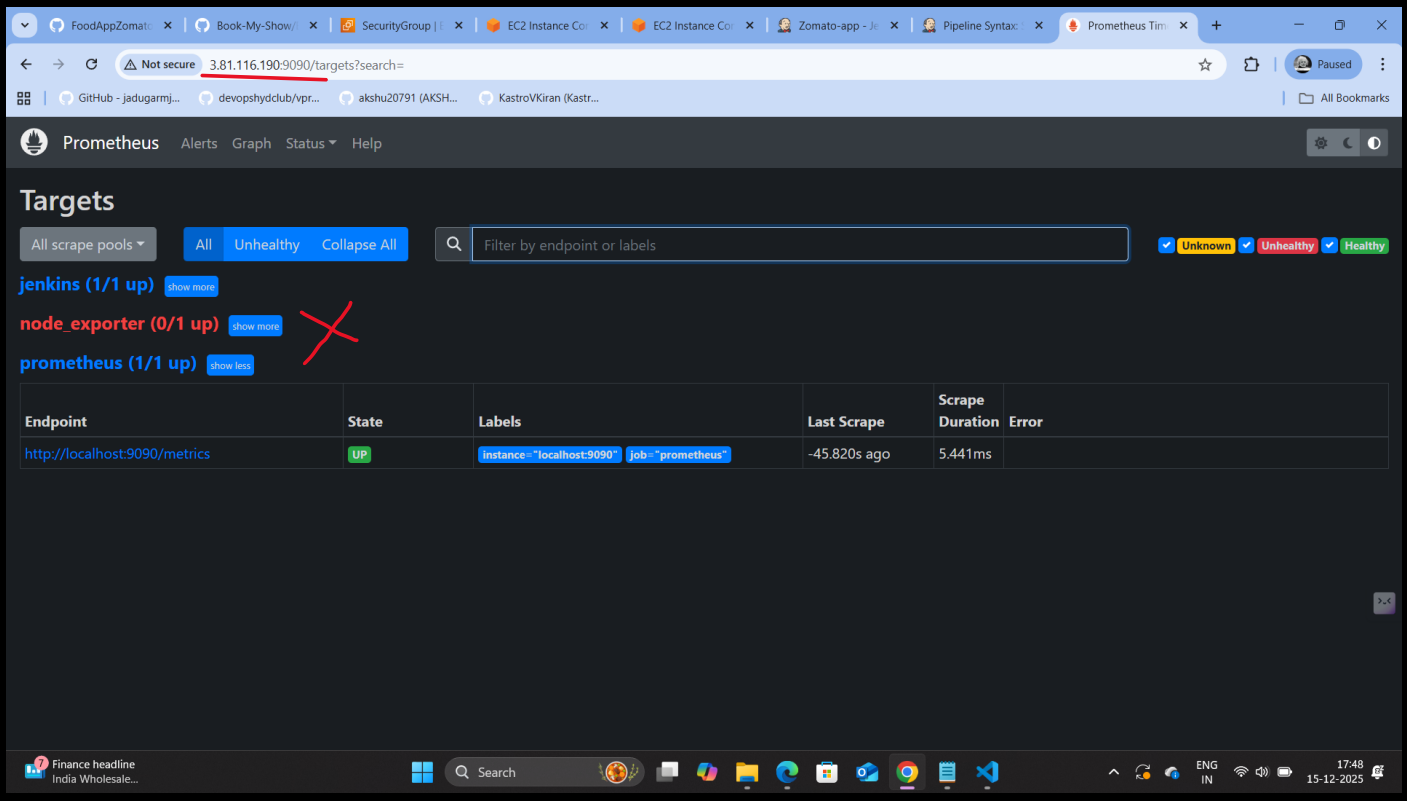
Step-30- Here we checked the status of the node exporter and its running fine.



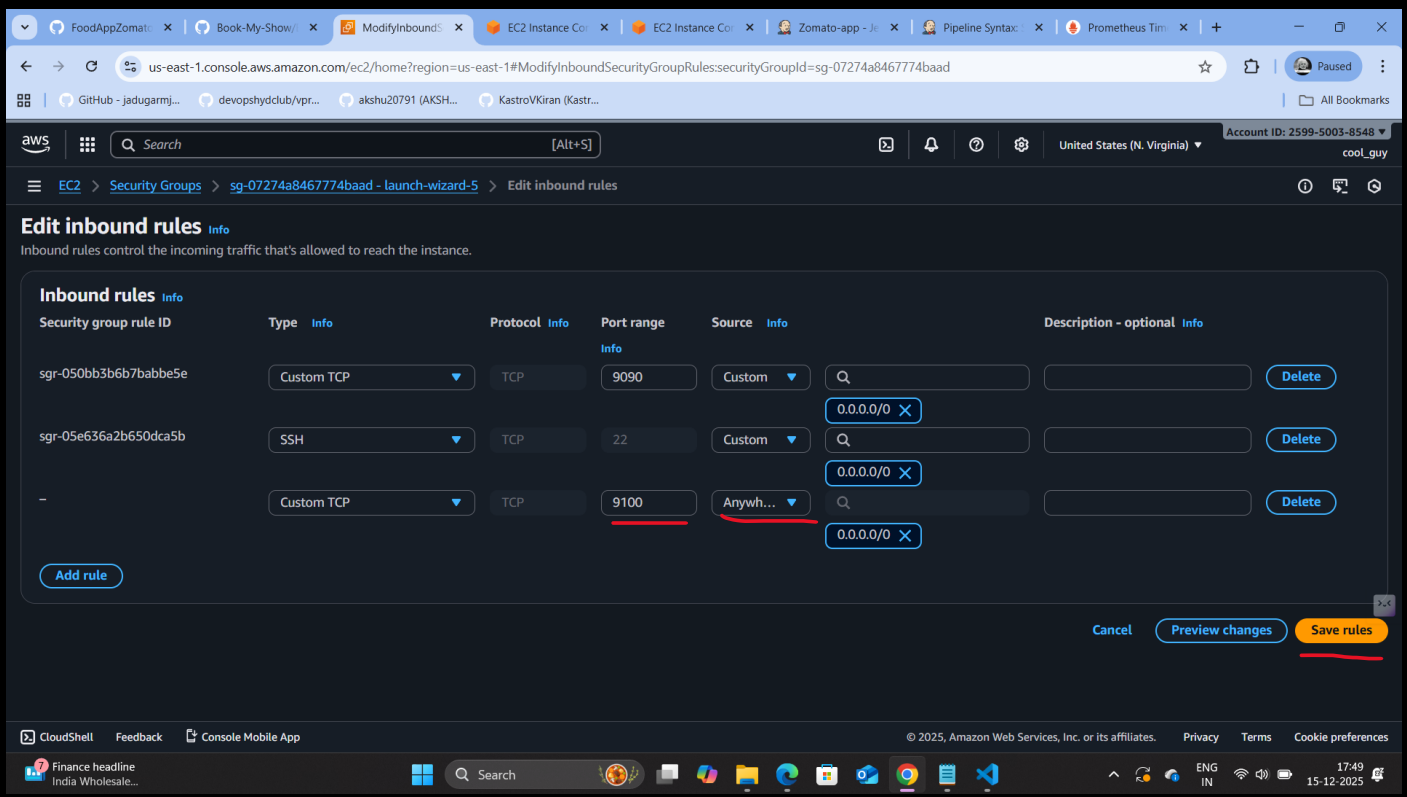
Step-31- We are configuring Prometheus.yml file to scrape the metrics from Jenkins and node exporter.



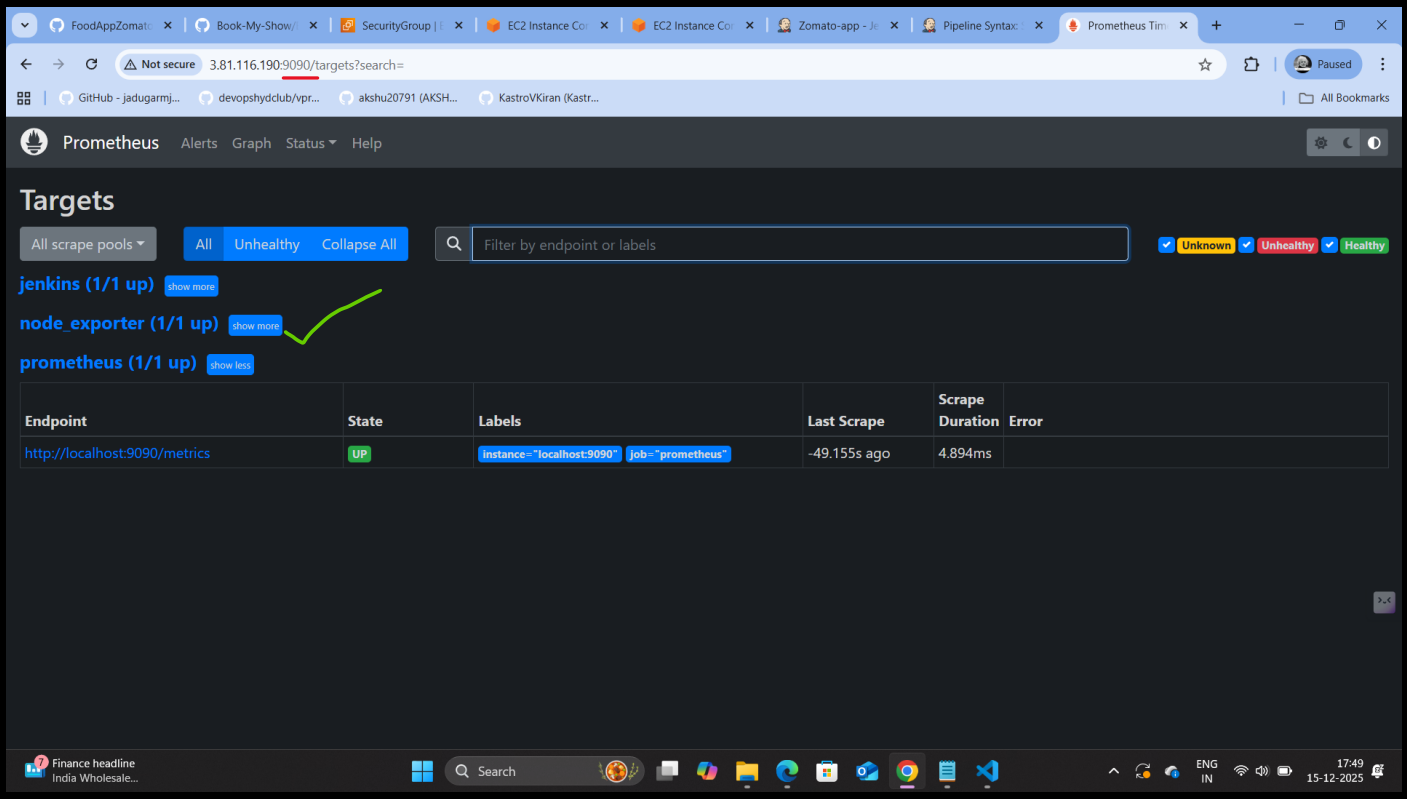
Step-32- We can see the node exporter is not running her as the port no. 9100 is not enabled in the security group.



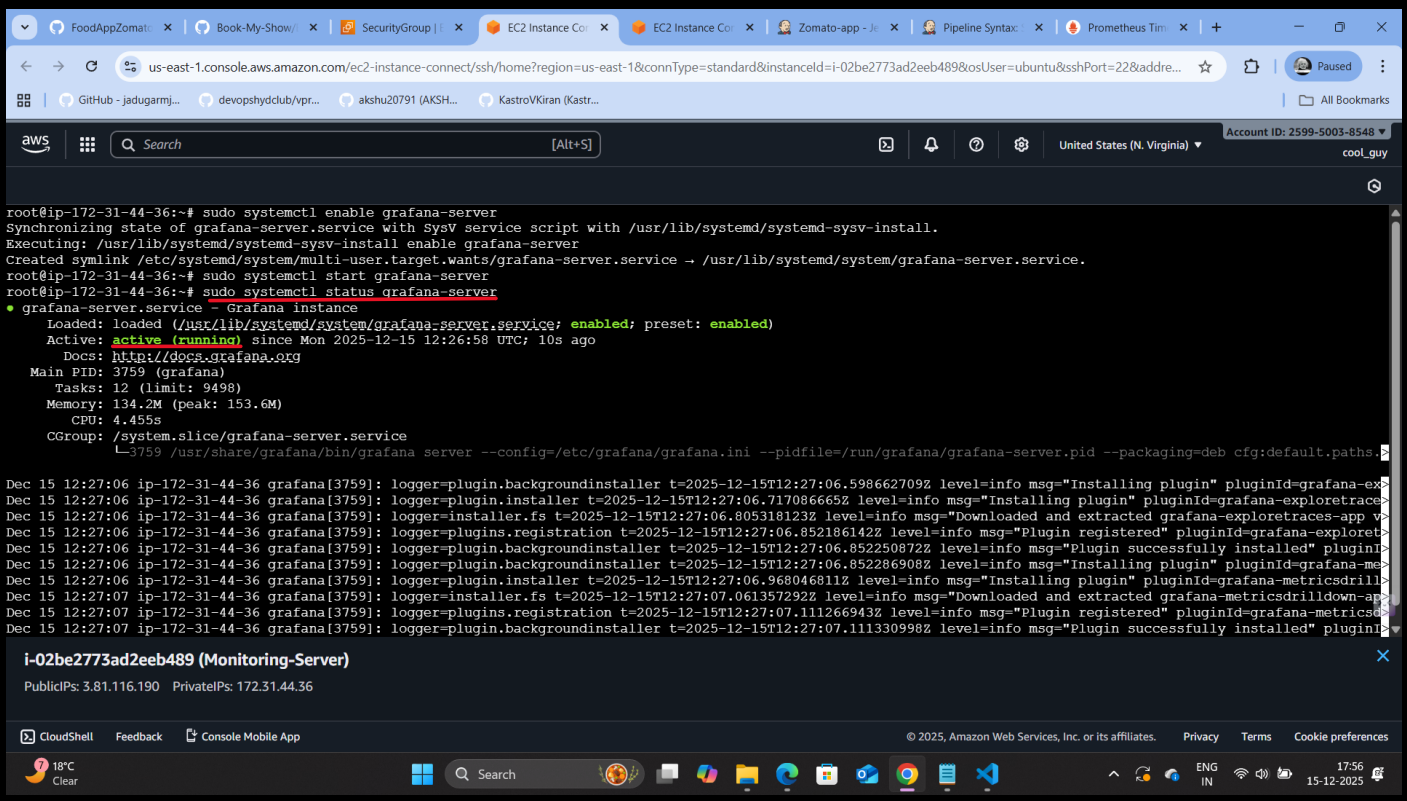
Step-33- Here we enabled the port no. 9100 in the security group.



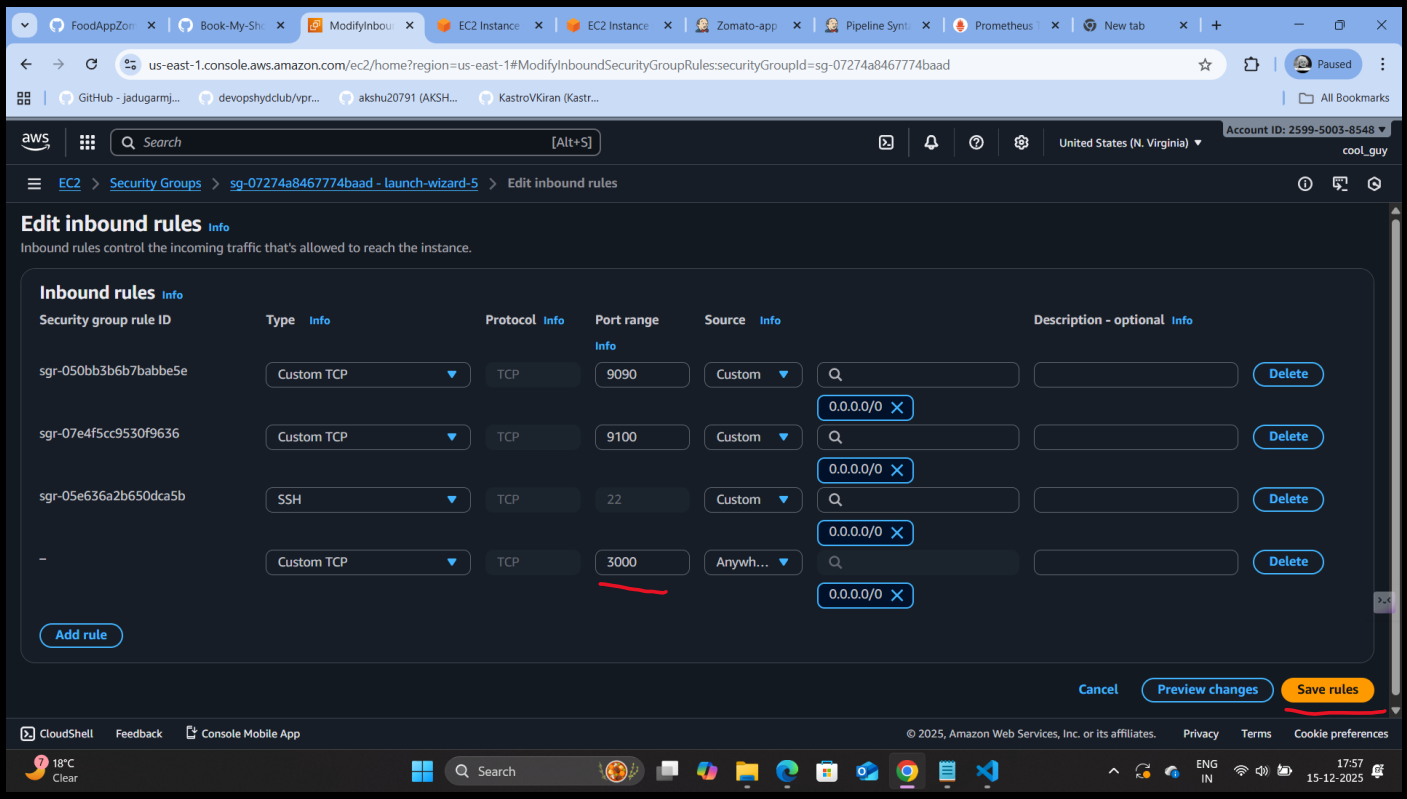
Step-34- Here node exporter is running now.



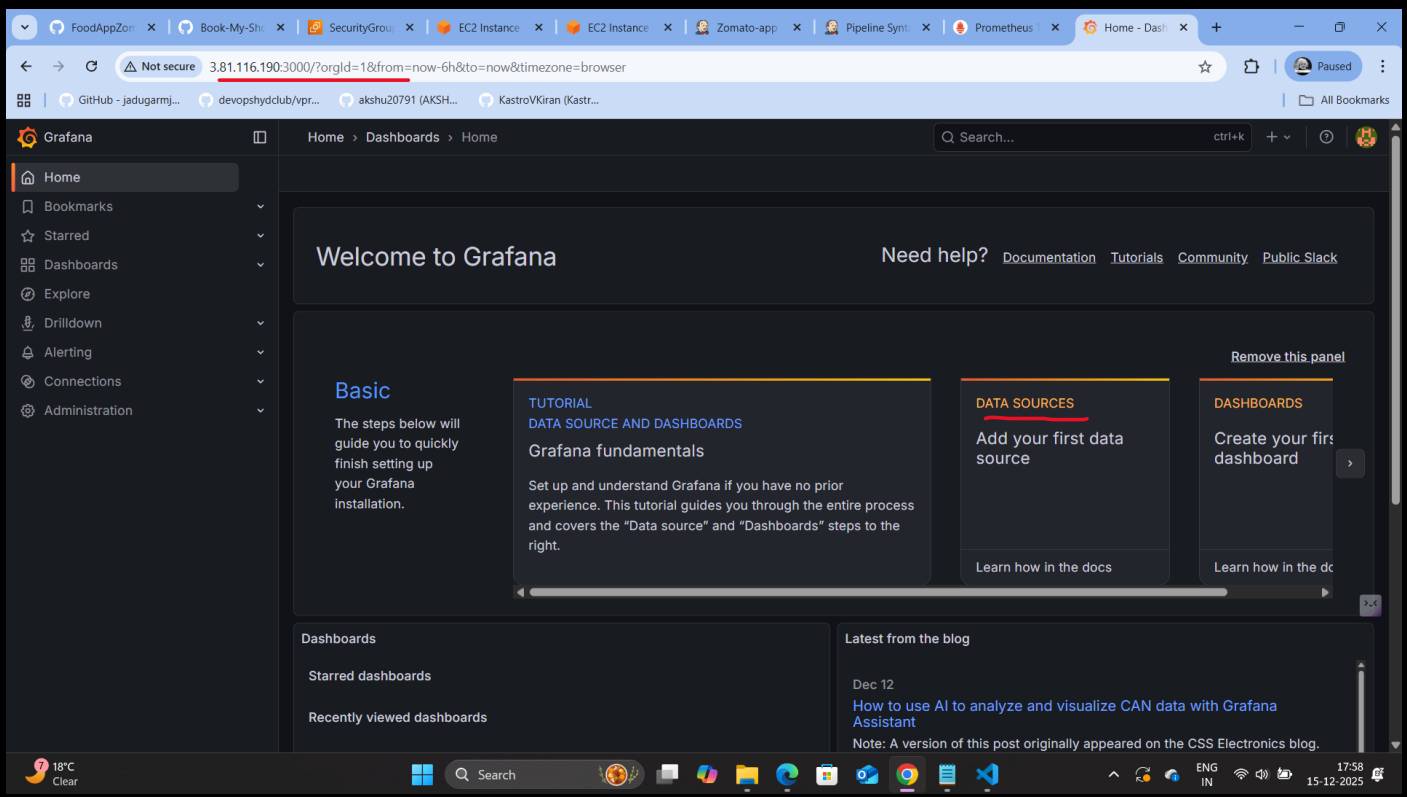
Step-35- Now we installed the Grafana and checked its status.



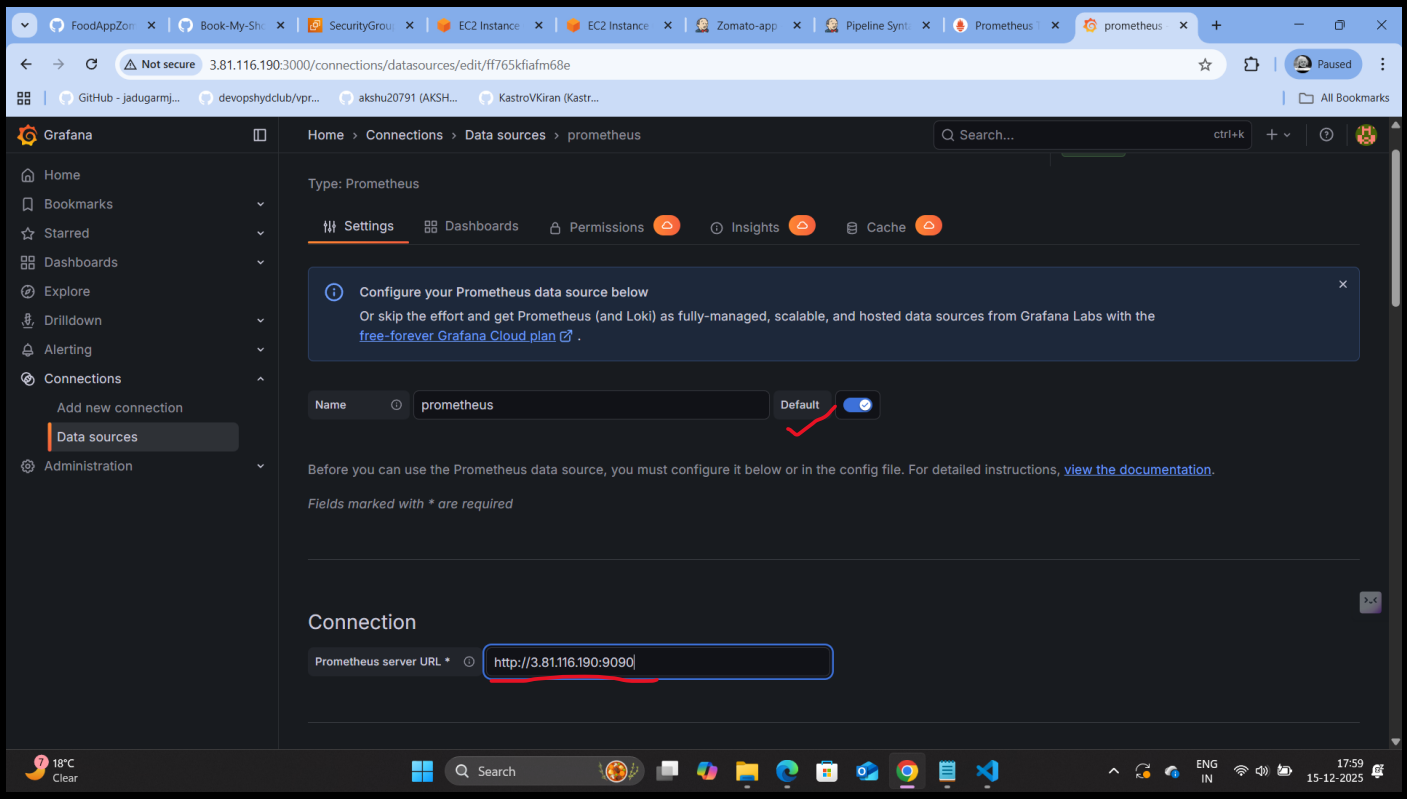
Step-36- We enabled the port no. 3000 for Grafana in the security group.



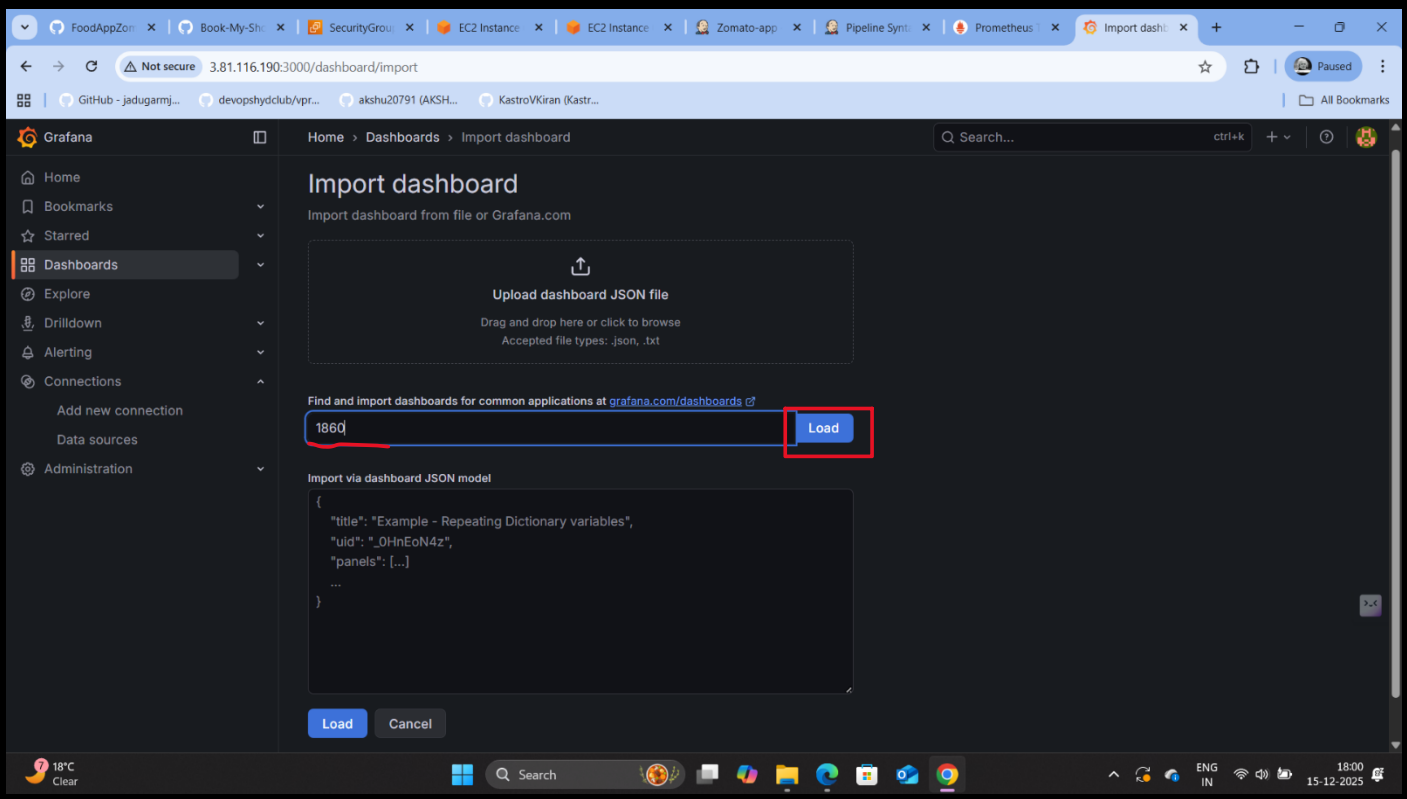
Step-37- Here we add the data source from which we want to fetch the data to make the dashboards.



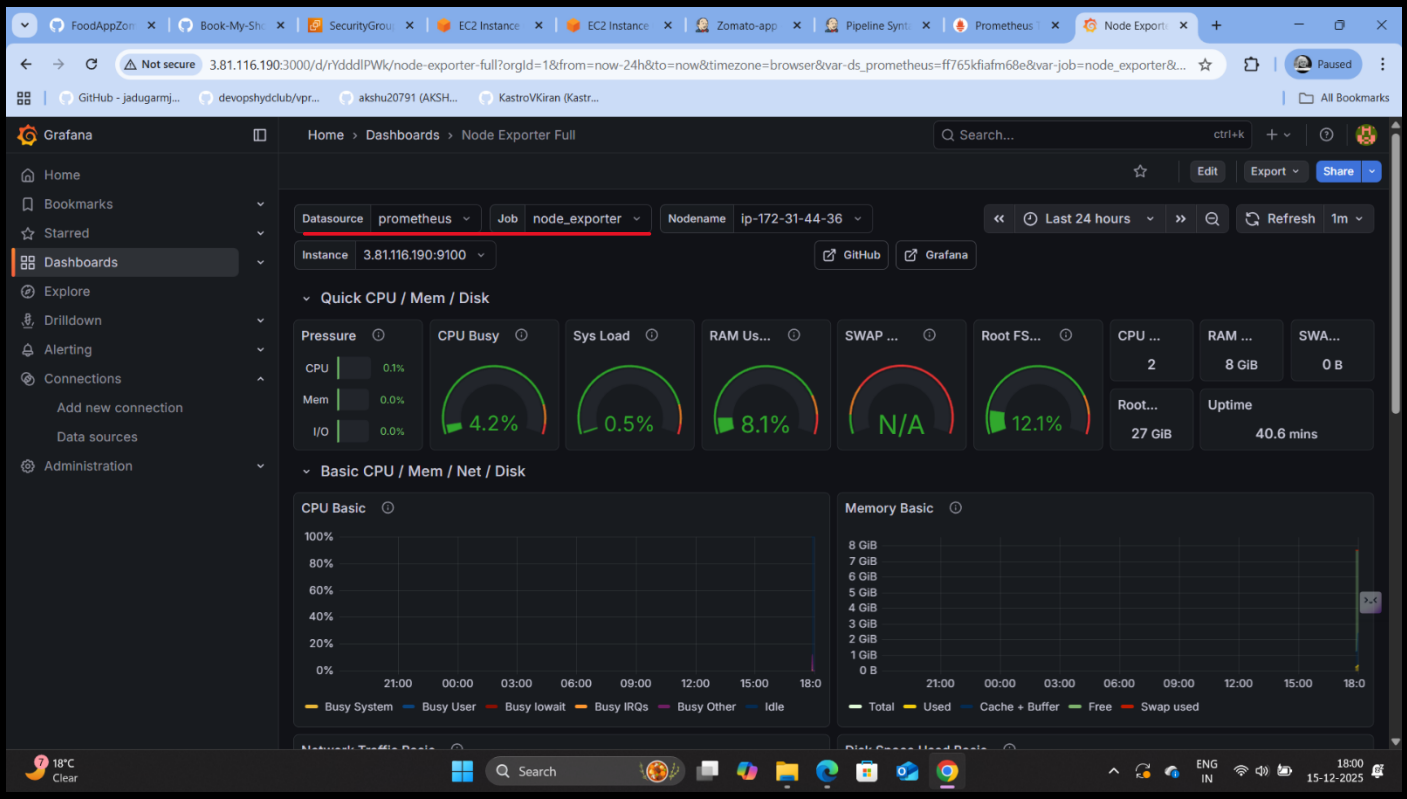
Step-38- We had given the URL of the Prometheus.



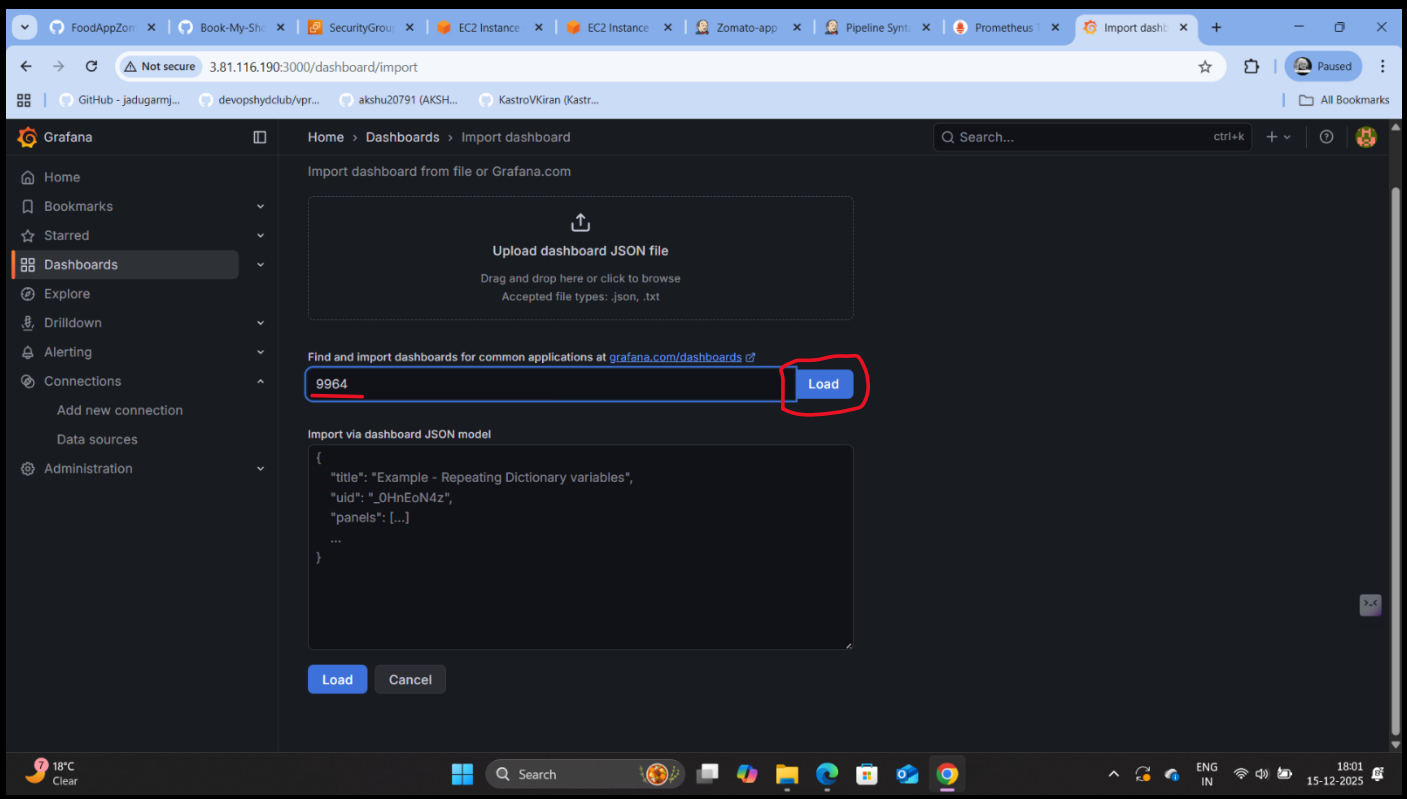
Step-39 Now we will import the dashboard , 1860 is for the node exporter.



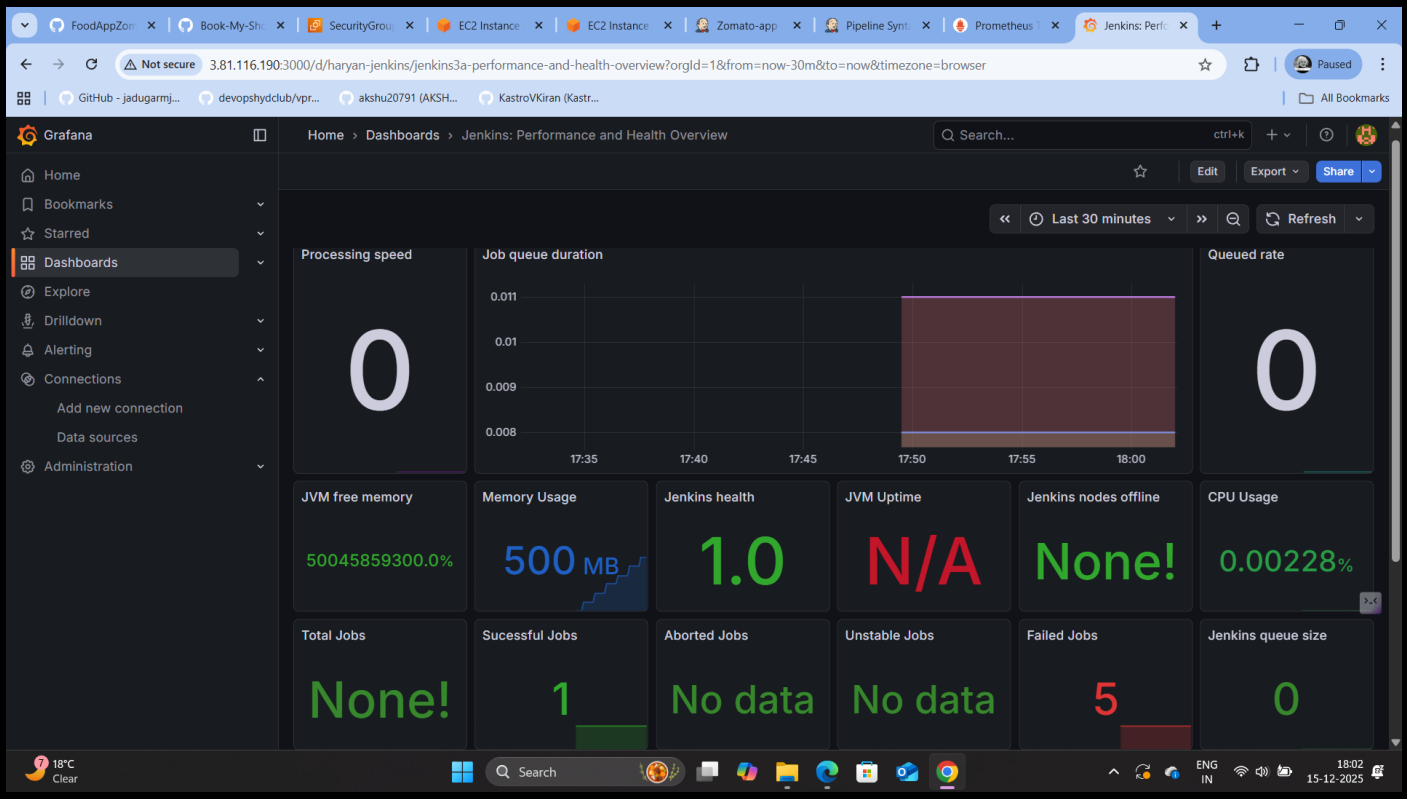
Step-40- We can see the dashboard



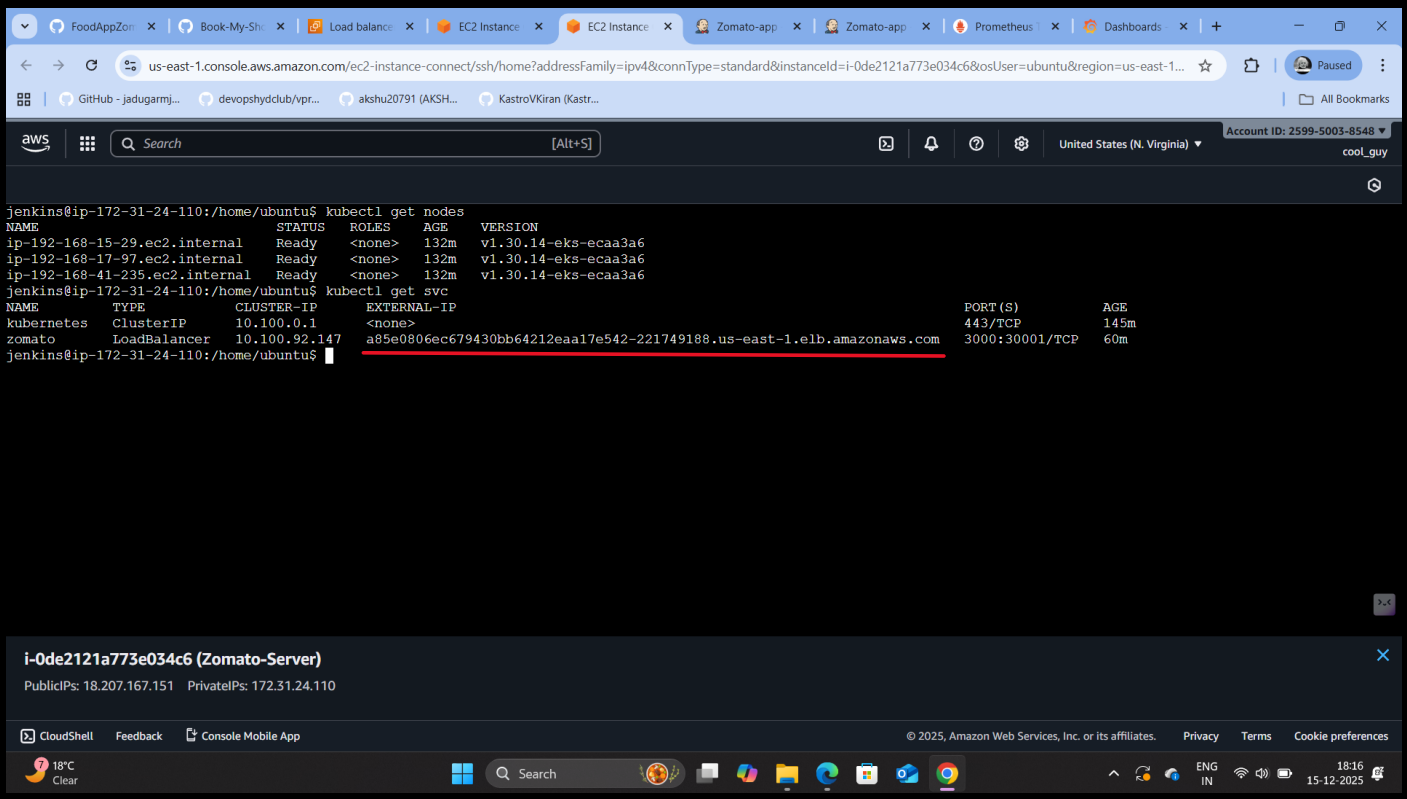
Step-41- 9964 is the dashboard for Jenkins , paste it and load it.



Step-42- Here is our Jenkins dashboard



Step-43- Now we will get the URL of the Load Balancer to run our application



Step-44- Paste the URL and we can see that our application is deployed successfully.