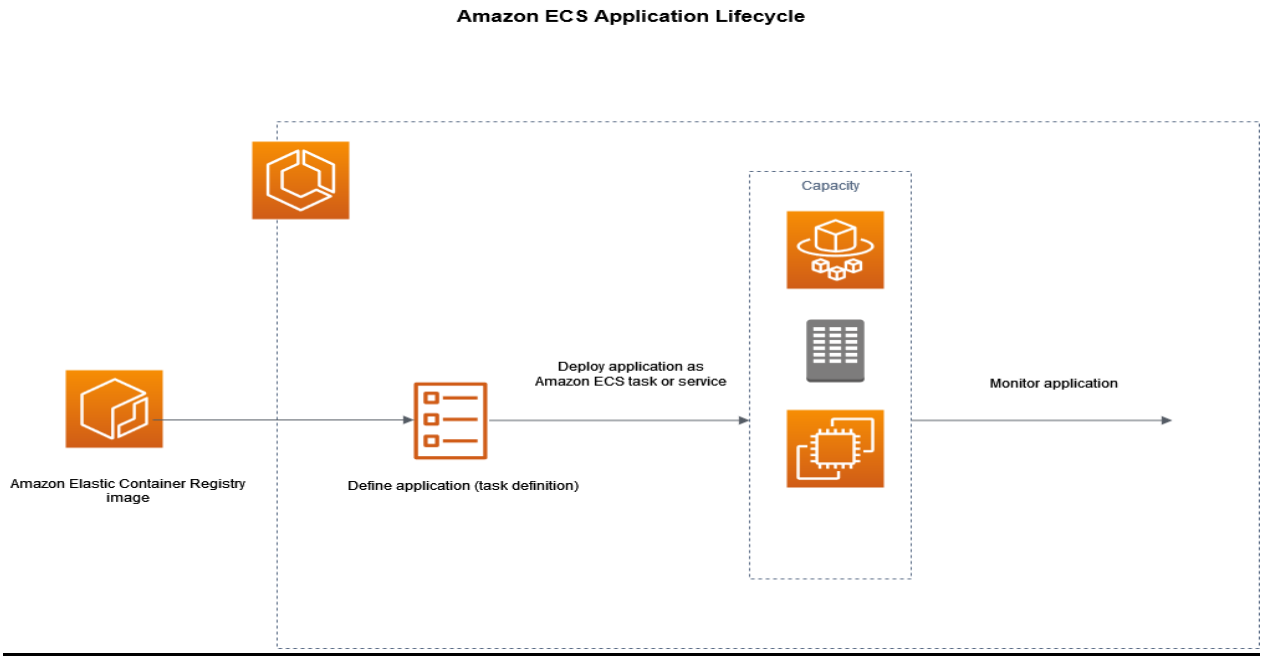
## Deploy Serverless Applications

### ECS-[Elastic Container Service]

Amazon Elastic Container Service (Amazon ECS) is a fully managed container orchestration service that helps you easily deploy, manage, and scale containerized applications.

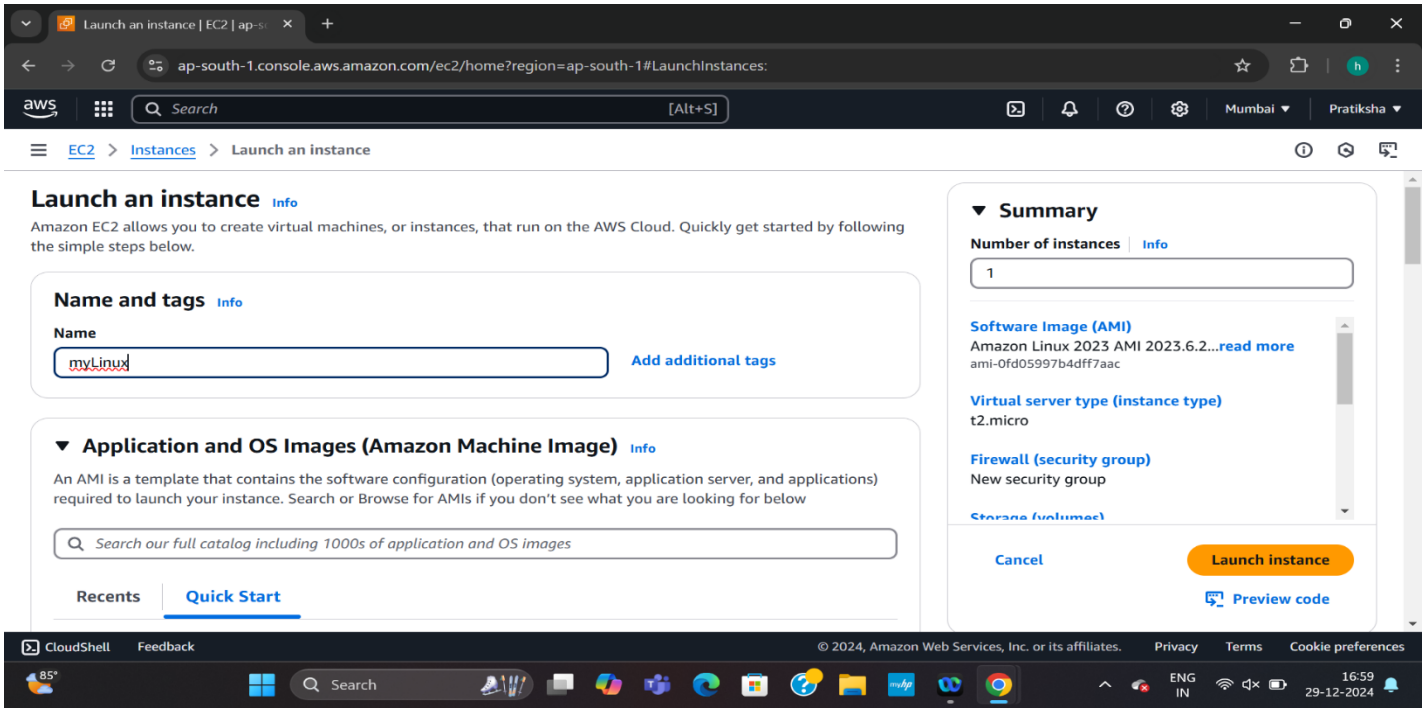
## Application lifecycle

The following diagram shows the application lifecycle and how it works with the Amazon ECS components.




Steps:

- Launch a Linux instance

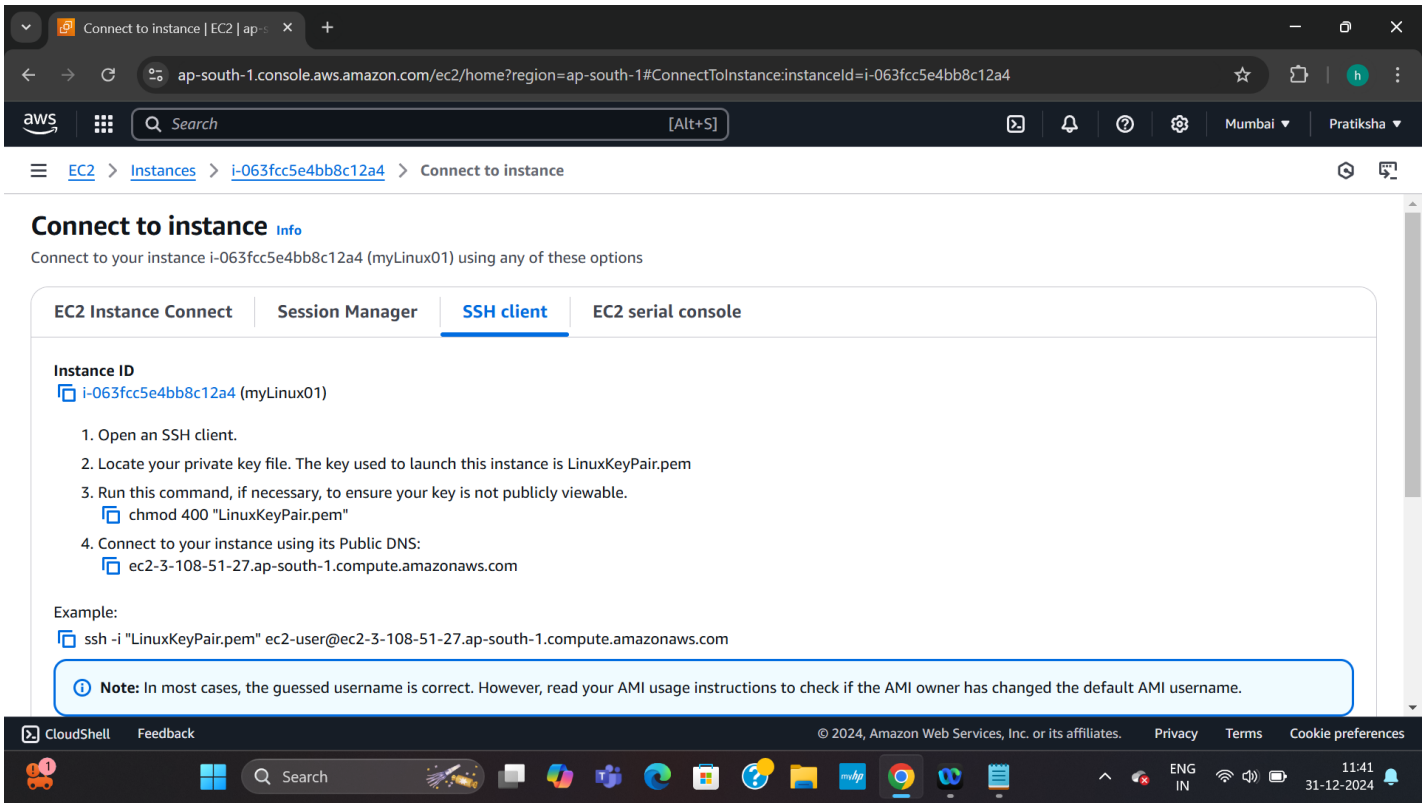



-

-

- Connect with instance

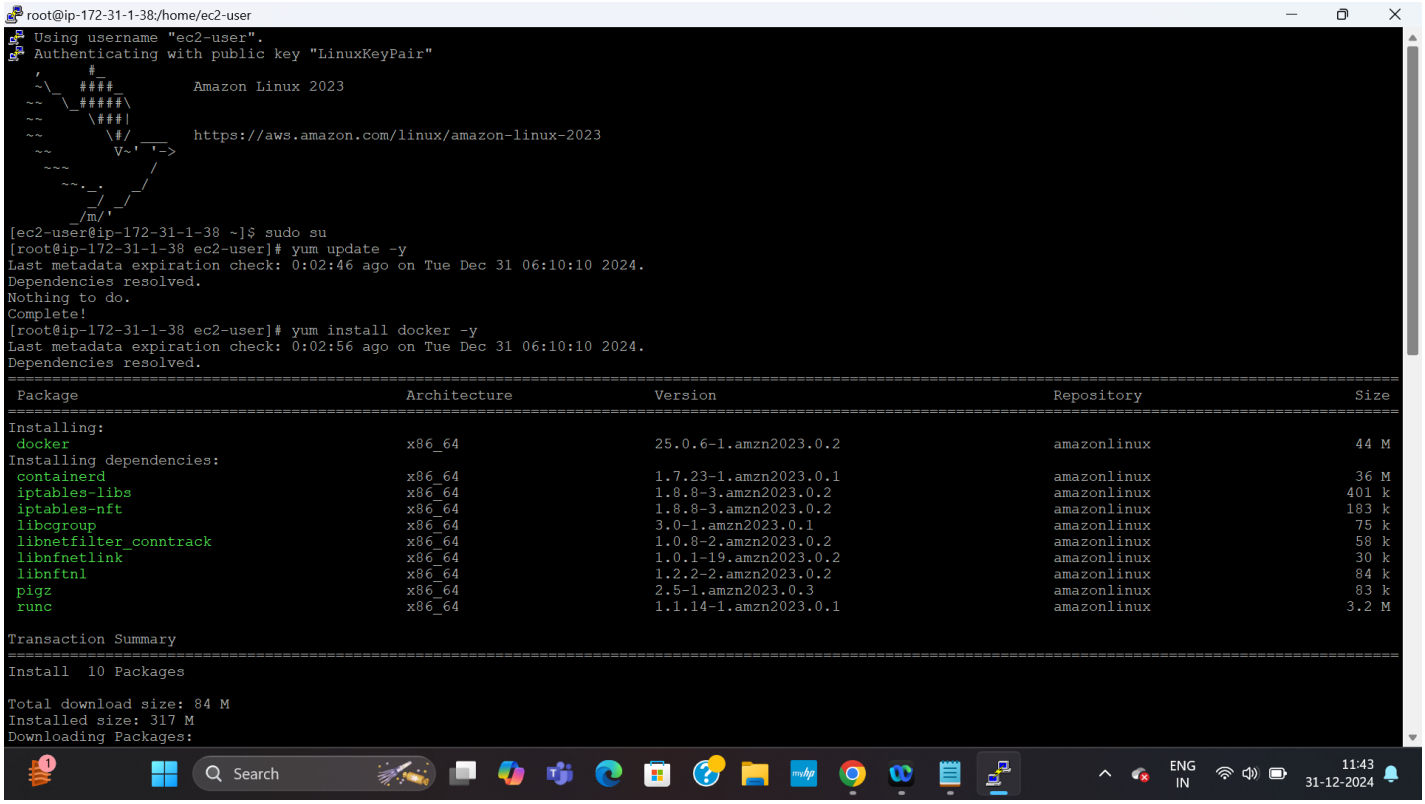
-



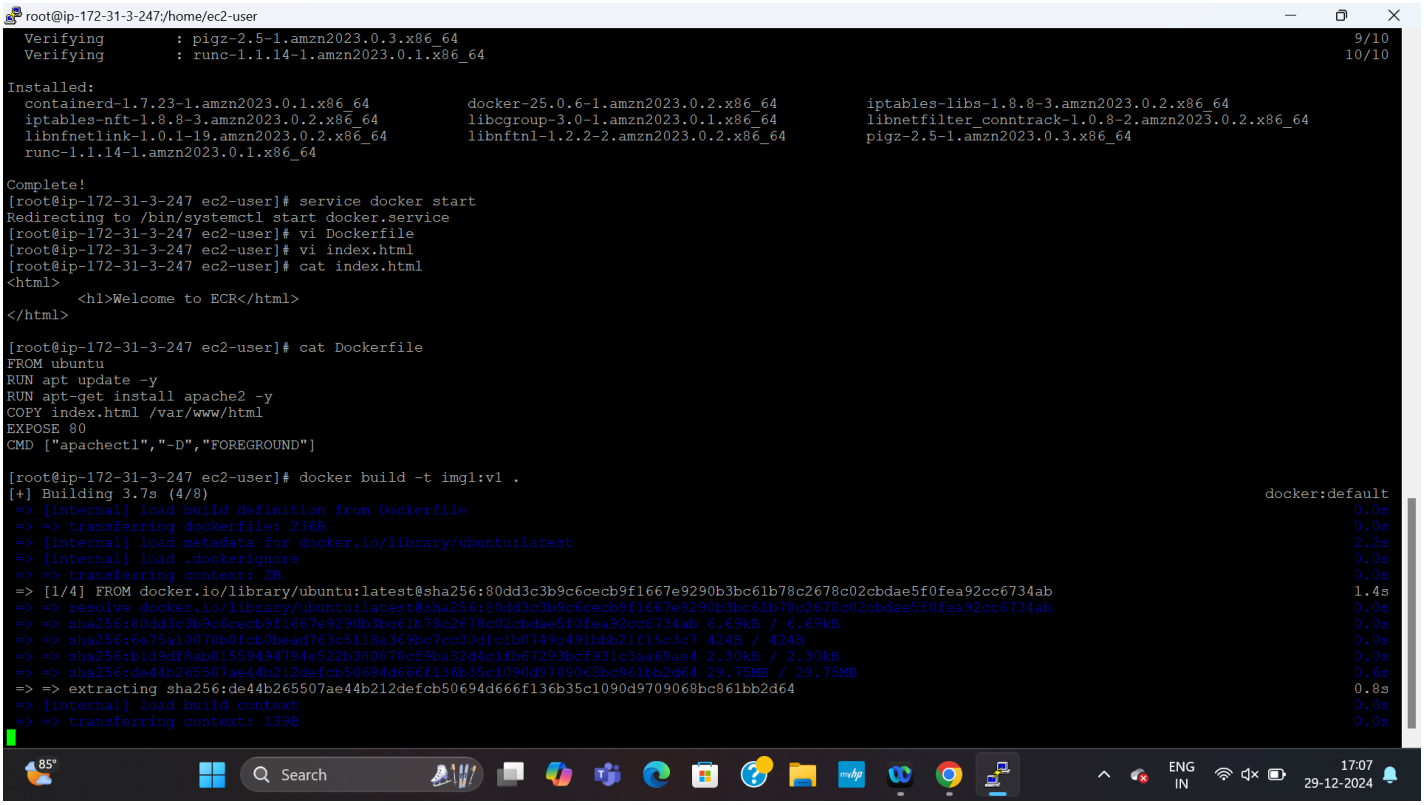

-

-

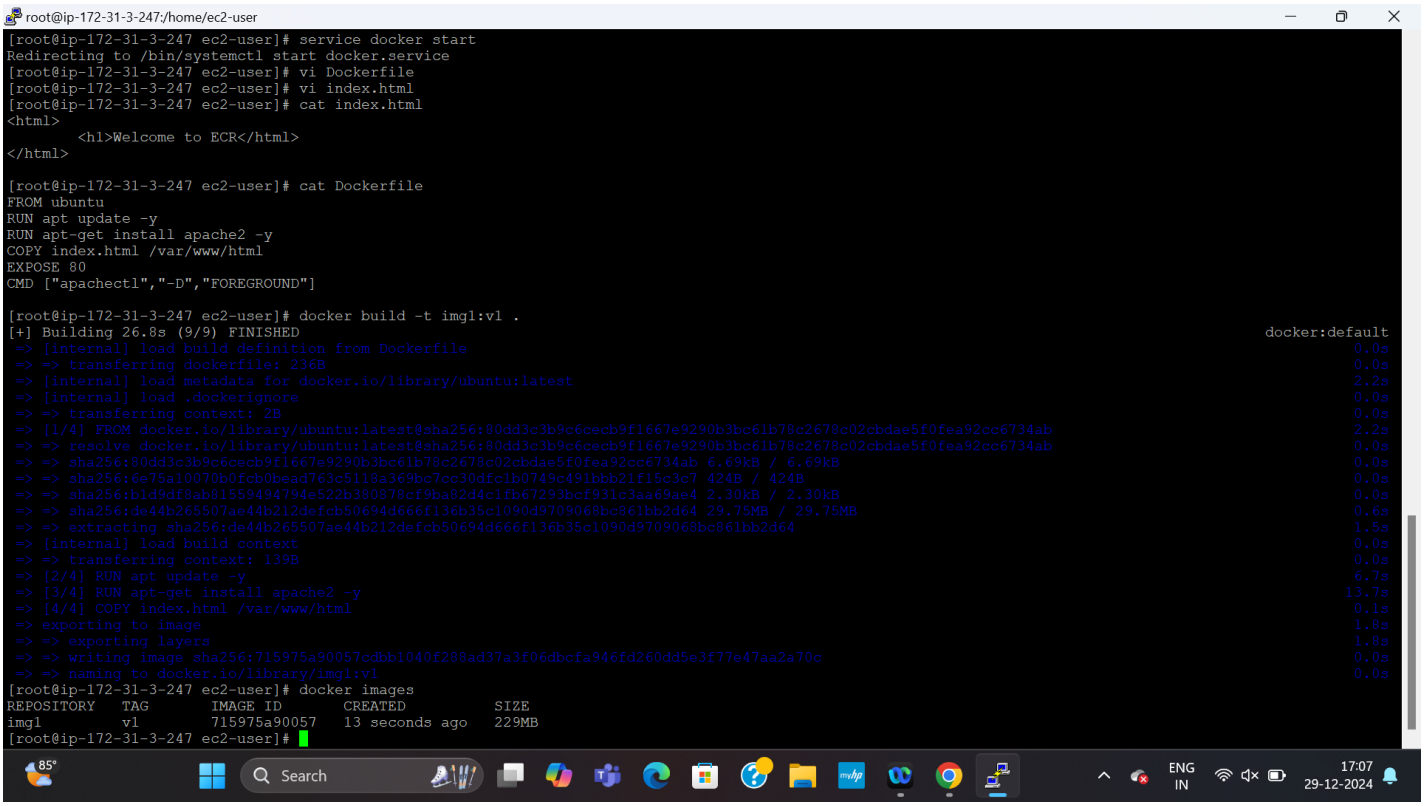
- --> Install Docker



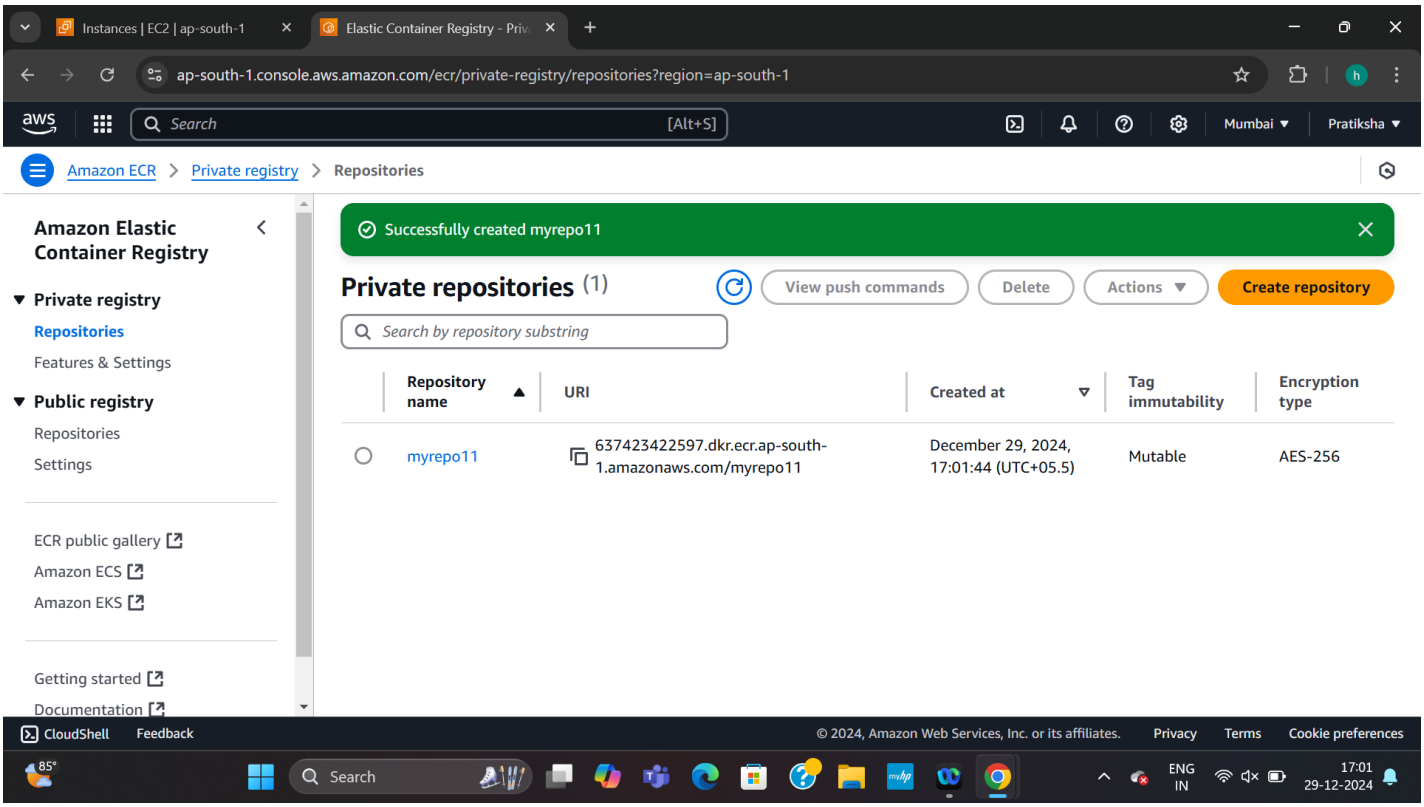

-



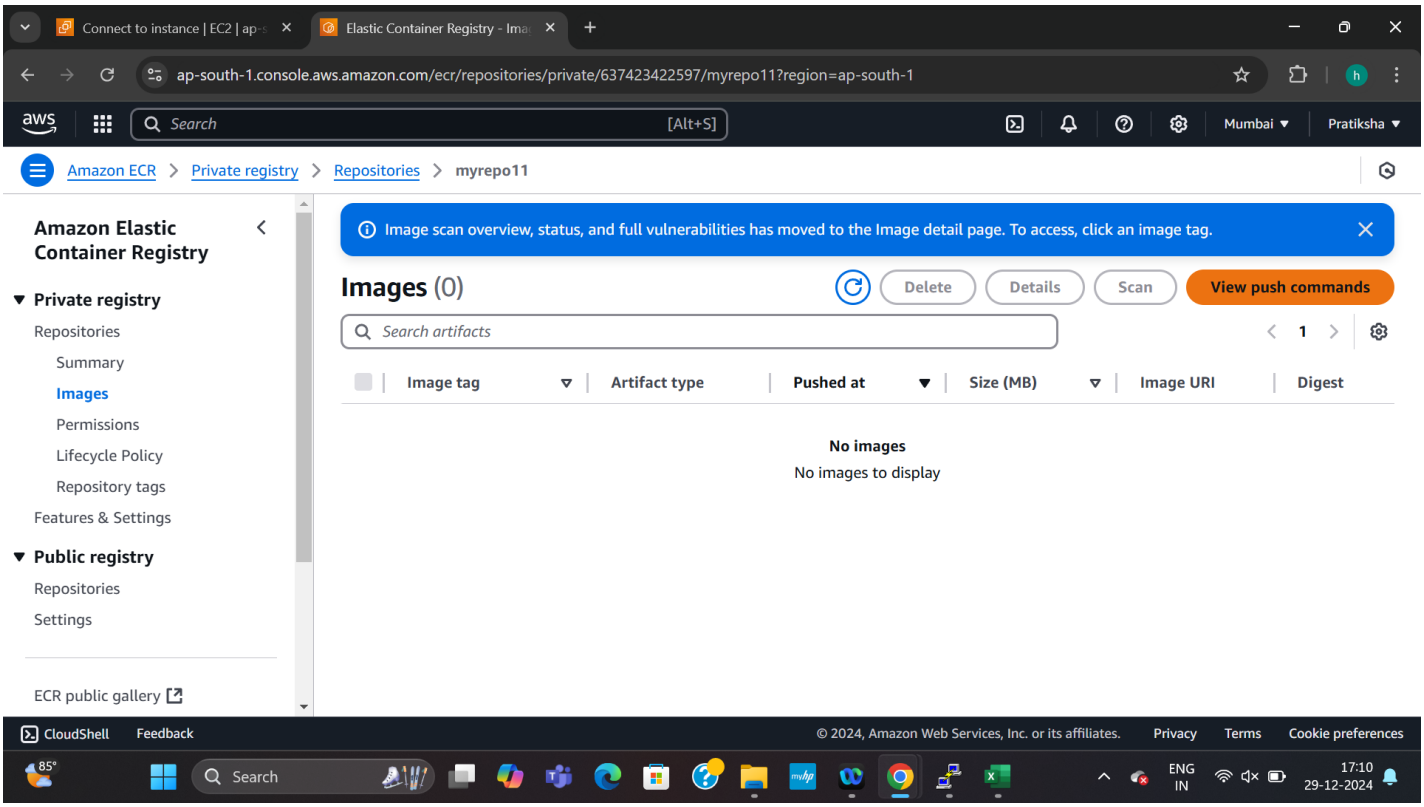

--> And create Image

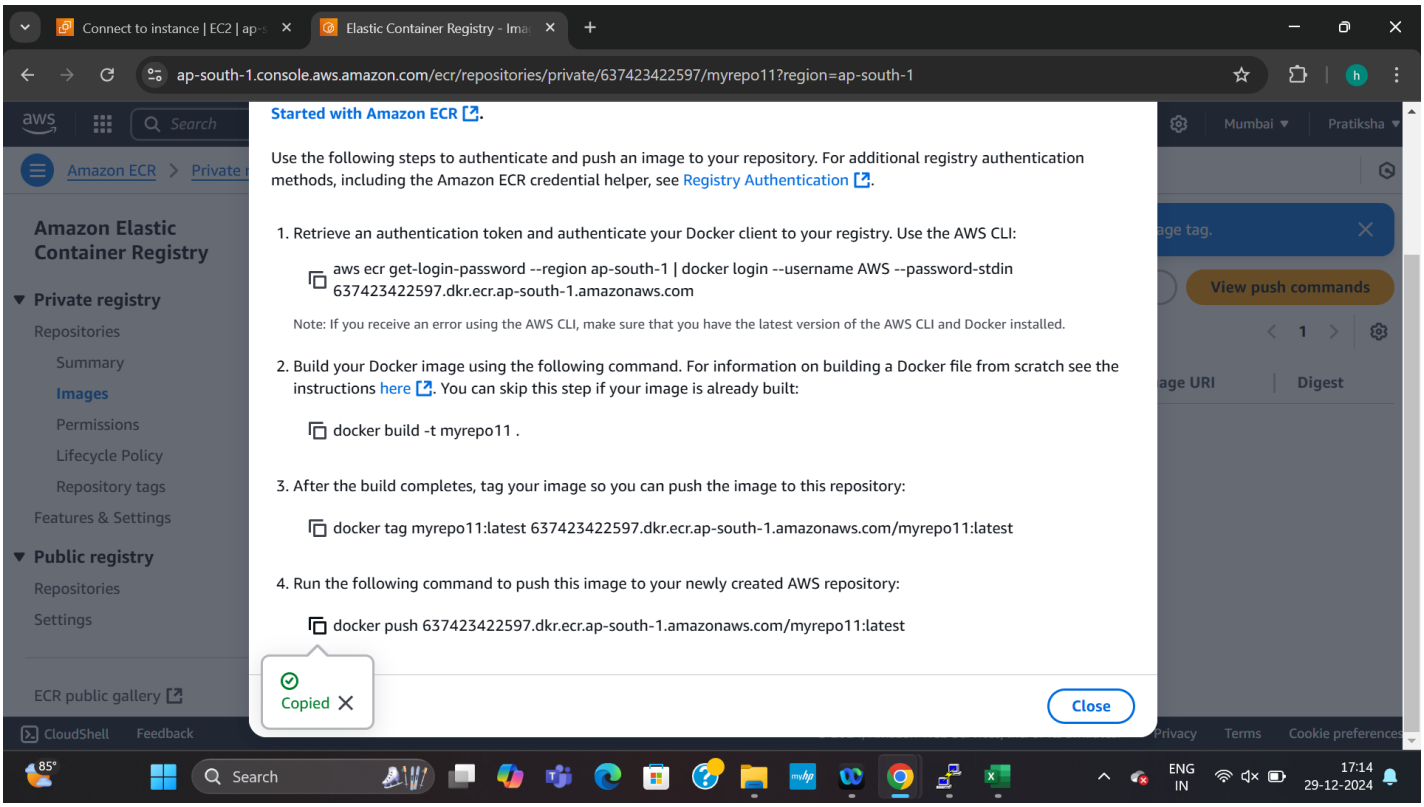


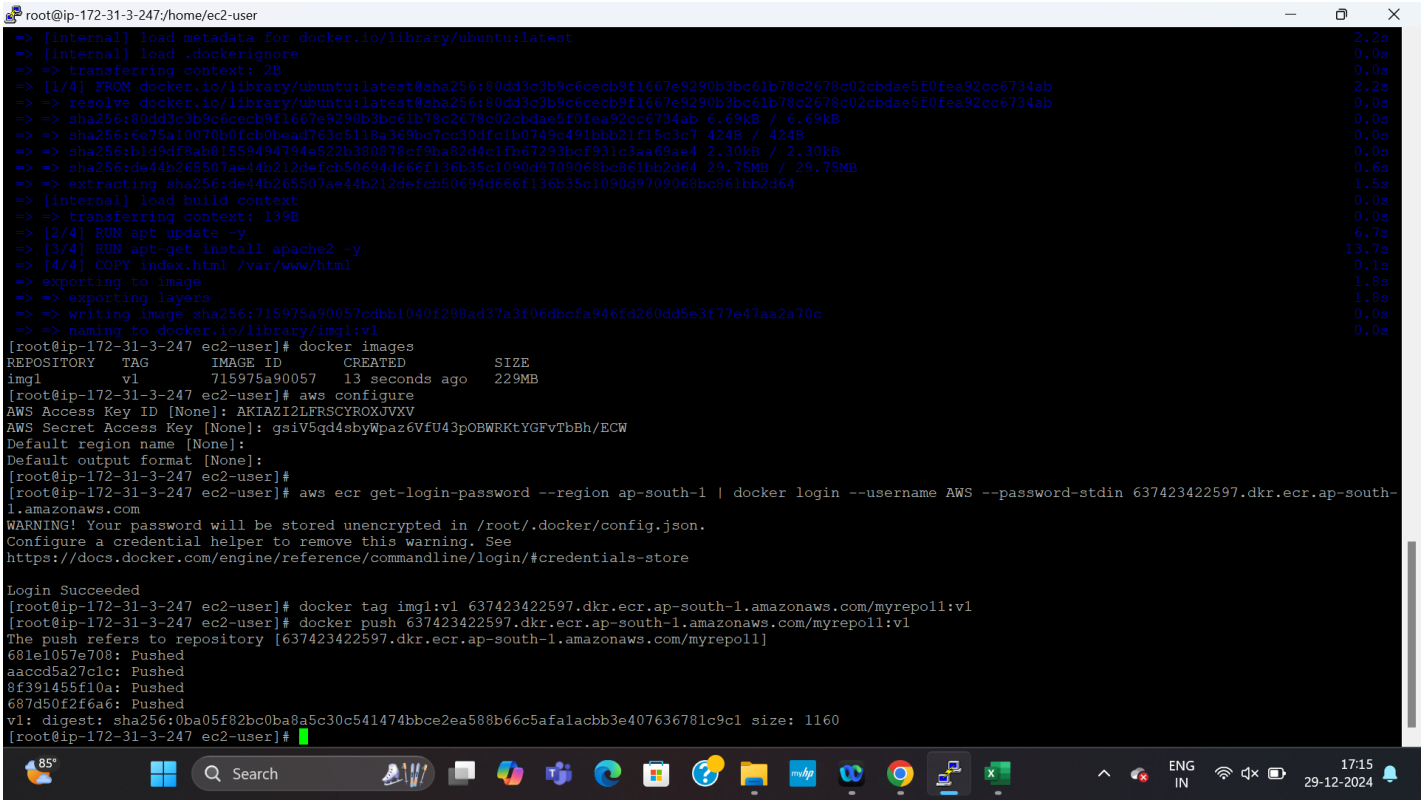

--> create repository in ECR--> provide name

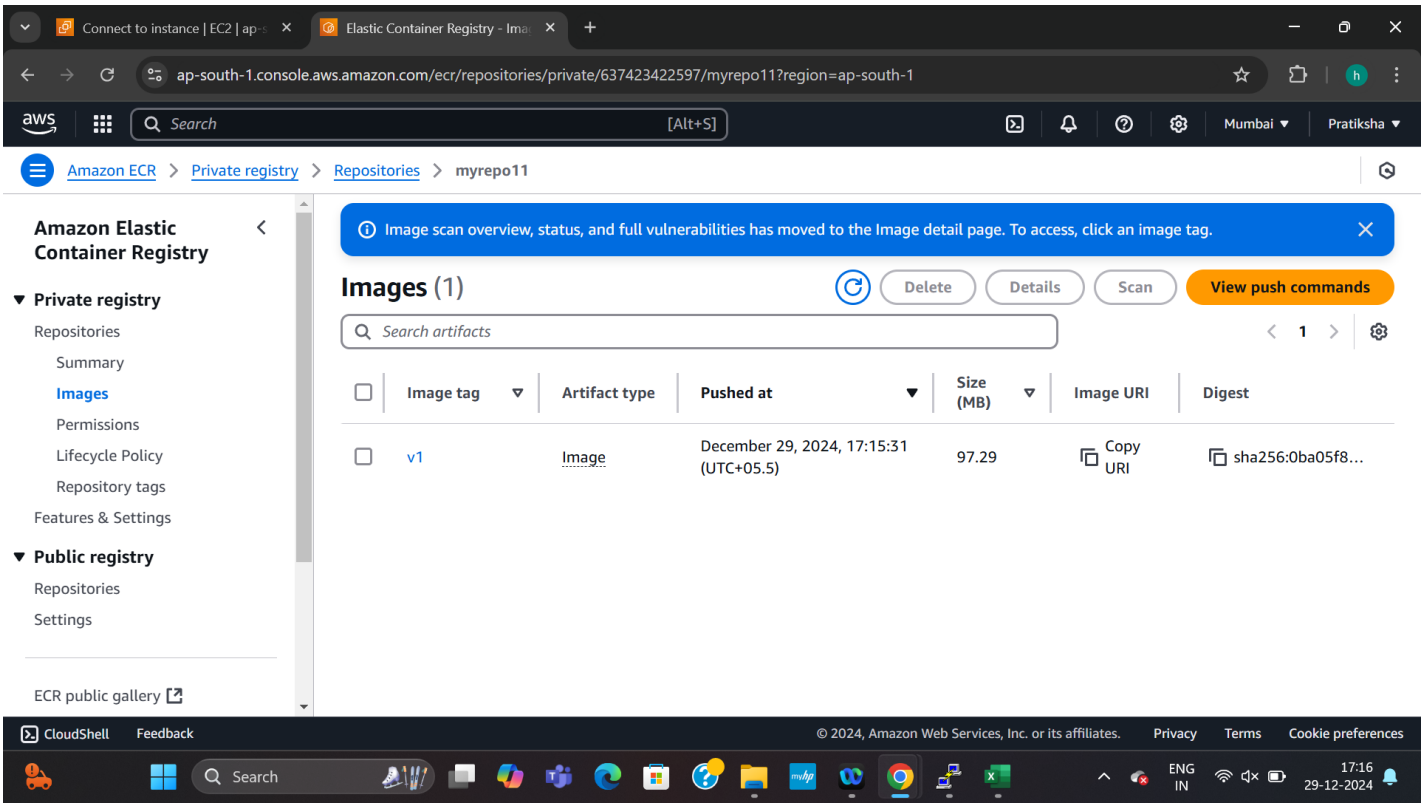



--> goin repo name --> select viewpush command nd copy paste the commands inside putty after configuring AWS



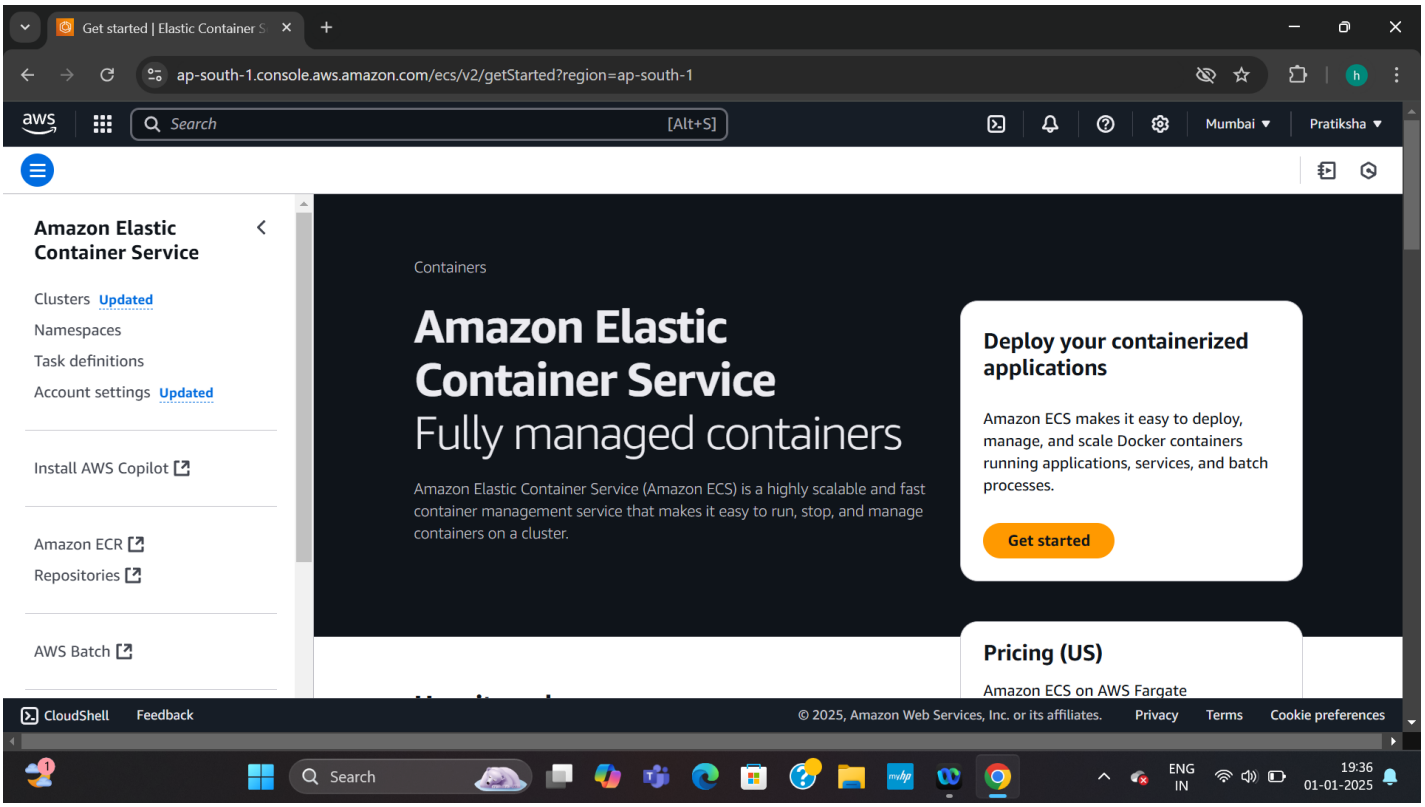

Connect to instance | EC2 | ap-s
Elastic Container Registry - Ima
ap-south-1.console.aws.amazon.com/ecr/repositories/private/637423422597/myrepo11?region=ap-south-1
Search
Mumbai
Pratiksha
Amazon ECR
Private r
Amazon Elastic Container Registry
Private registry
Repositories
Summary
Images
Permissions
Lifecycle Policy
Repository tags
Features & Settings
Public registry
Repositories
Settings
ECR public gallery
CloudShell
Feedback
Started with Amazon ECR.
Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see Registry Authentication.
1. Retrieve an authentication token and authenticate your Docker client to your registry. Use the AWS CLI:
aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 637423422597.dkr.ecr.ap-south-1.amazonaws.com
Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.
2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions here. You can skip this step if your image is already built:
docker build -t myrepo11 .
3. After the build completes, tag your image so you can push the image to this repository:
docker tag myrepo11:latest 637423422597.dkr.ecr.ap-south-1.amazonaws.com/myrepo11:latest
4. Run the following command to push this image to your newly created AWS repository:
docker push 637423422597.dkr.ecr.ap-south-1.amazonaws.com/myrepo11:latest
Copied
Close
View push commands
Digest
Privacy
Terms
Cookie preferences
ENG IN
17:14
29-12-2024


root@ip-172-31-3-247:/home/ec2-user
=> [internal] load metadata for docker.io/library/ubuntu:latest 2.2s
=> [internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [1/4] FROM docker.io/library/ubuntu:latest@sha256:80dd3c3b9c6cecb9f1667e9290b3bc61b78c2678c02cbdae5f0fea92cc6734ab 2.2s
=> => resolve docker.io/library/ubuntu:latest@sha256:80dd3c3b9c6cecb9f1667e9290b3bc61b78c2678c02cbdae5f0fea92cc6734ab 0.0s
=> => sha256:80dd3c3b9c6cecb9f1667e9290b3bc61b78c2678c02cbdae5f0fea92cc6734ab 6.69kB / 6.69kB 0.0s
=> => sha256:6e75a10070b0fcb0bead763c5118a369bc7cc30dfc1b0749c491bbb21f15c3c7 424B / 424B 0.0s
=> => sha256:b1d9df8ab81559494794e522b380878cf9ba82d4c1fb67293bcf931c3aa69ae4 2.30kB / 2.30kB 0.0s
=> => sha256:de44b265507ae44b212defcb50694d666f136b35c1090d9709068bc861bb2d64 29.75MB / 29.75MB 0.6s
=> => extracting sha256:de44b265507ae44b212defcb50694d666f136b35c1090d9709068bc861bb2d64 1.5s
=> [internal] load build context 0.0s
=> => transferring context: 139B 0.0s
=> [2/4] RUN apt update -y 6.7s
=> [3/4] RUN apt-get install apache2 -y 13.7s
=> [4/4] COPY index.html /var/www/html 0.1s
=> exporting to image 1.8s
=> => exporting layers 1.8s
=> => writing image sha256:715975a90057cdbb1040f288ad37a3f06dbcfa946fd260dd5e3f77e47aa2a70c 0.0s
=> => naming to docker.io/library/img1:v1 0.0s
[root@ip-172-31-3-247 ec2-user]# docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
img1 v1 715975a90057 13 seconds ago 229MB
[root@ip-172-31-3-247 ec2-user]# aws configure
AWS Access Key ID [None]: AKIAZI2LFRSCYROXJVXV
AWS Secret Access Key [None]: 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
Default region name [None]:
Default output format [None]:
[root@ip-172-31-3-247 ec2-user]#
[root@ip-172-31-3-247 ec2-user]# aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 637423422597.dkr.ecr.ap-south-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
[root@ip-172-31-3-247 ec2-user]# docker tag img1:v1 637423422597.dkr.ecr.ap-south-1.amazonaws.com/myrepo11:v1
[root@ip-172-31-3-247 ec2-user]# docker push 637423422597.dkr.ecr.ap-south-1.amazonaws.com/myrepo11:v1
The push refers to repository [637423422597.dkr.ecr.ap-south-1.amazonaws.com/myrepo11]
681e1057e708: Pushed
aaccd5a27c1c: Pushed
8f391455f10a: Pushed
687d50f2f6a6: Pushed
v1: digest: sha256:0ba05f82bc0ba8a5c30c541474bbce2ea588b66c5afa1acbb3e407636781c9c1 size: 1160
[root@ip-172-31-3-247 ec2-user]#
Search
ENG IN
17:15
29-12-2024

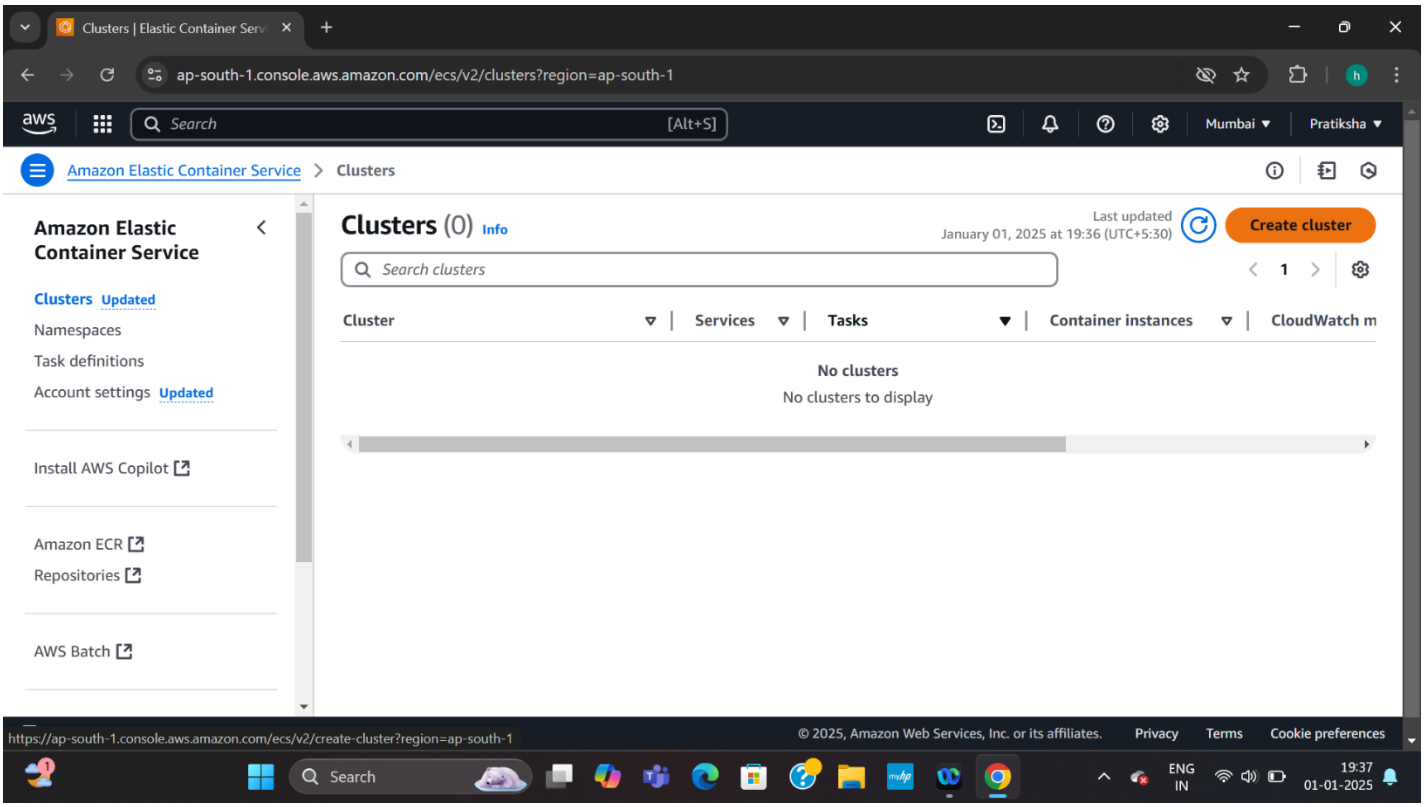



close putty -->

--> Search ECS

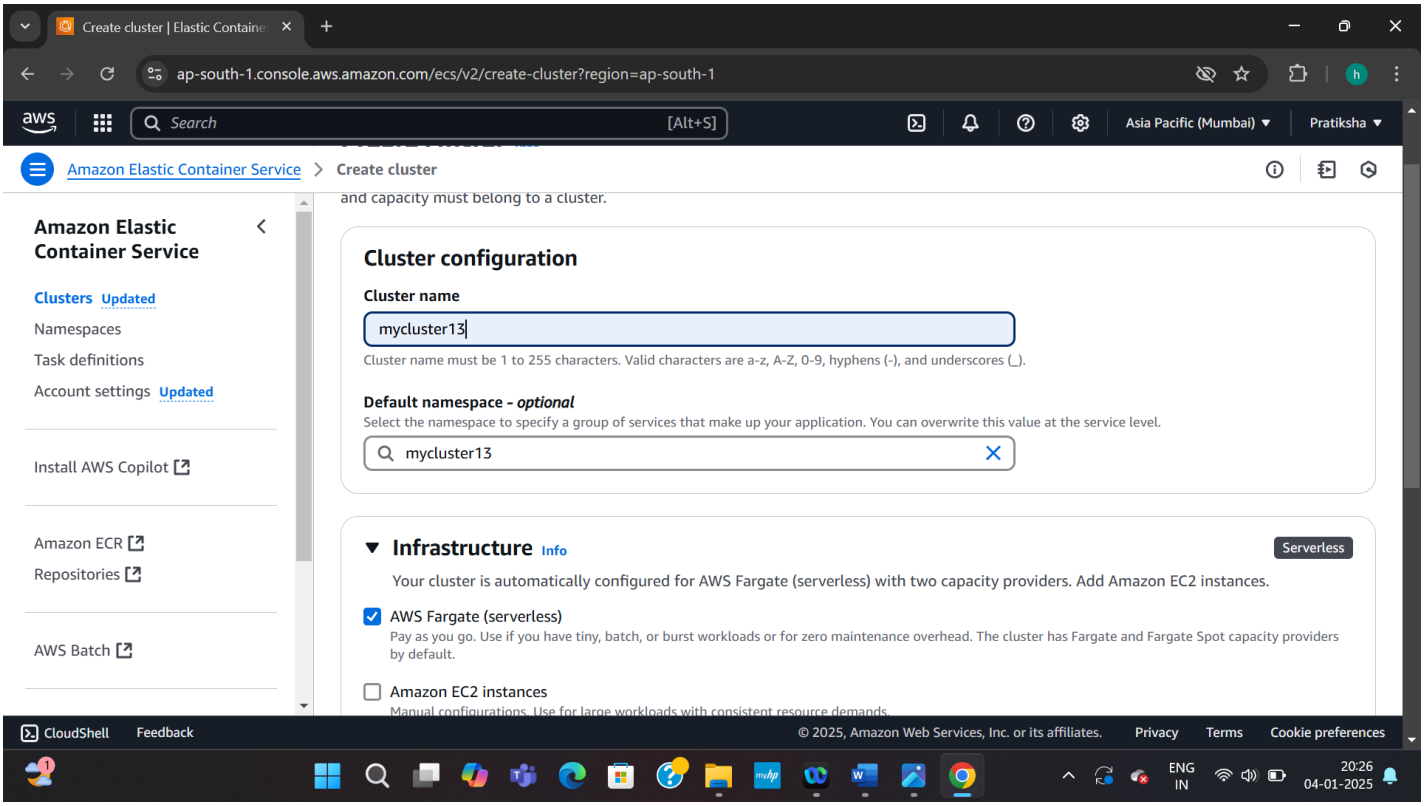



--> Create Cluster

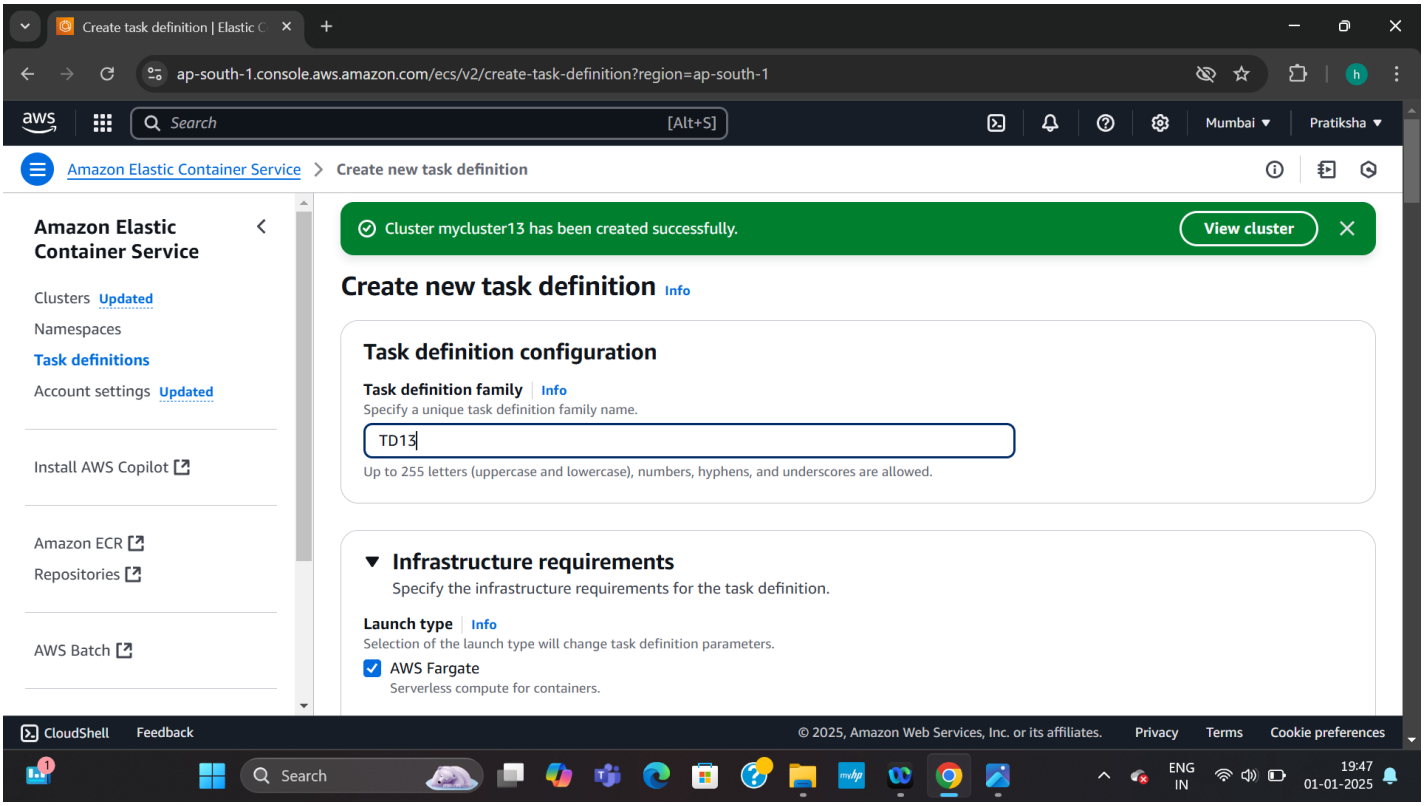



-->provide name

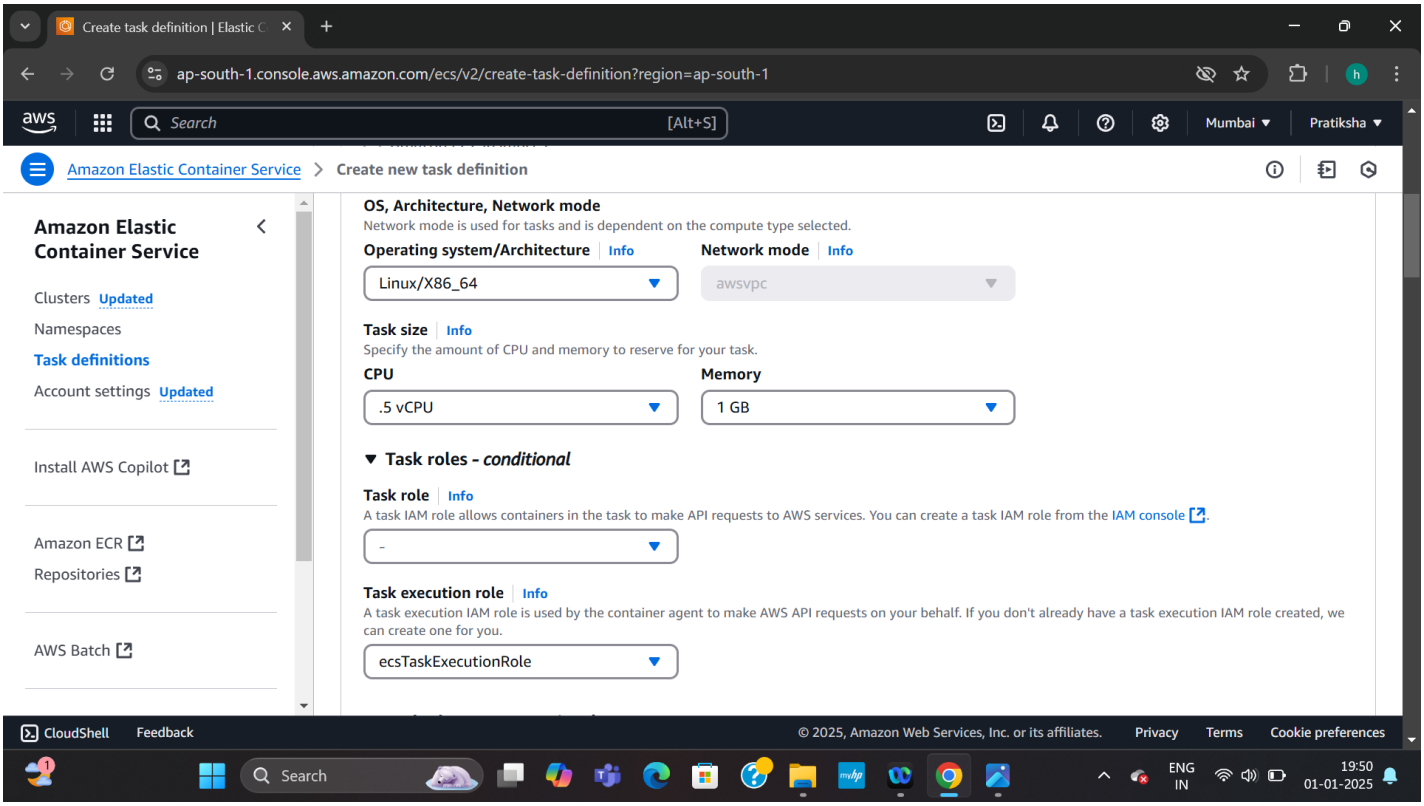
-->under Infrastucture select AWS fargate which is serverless



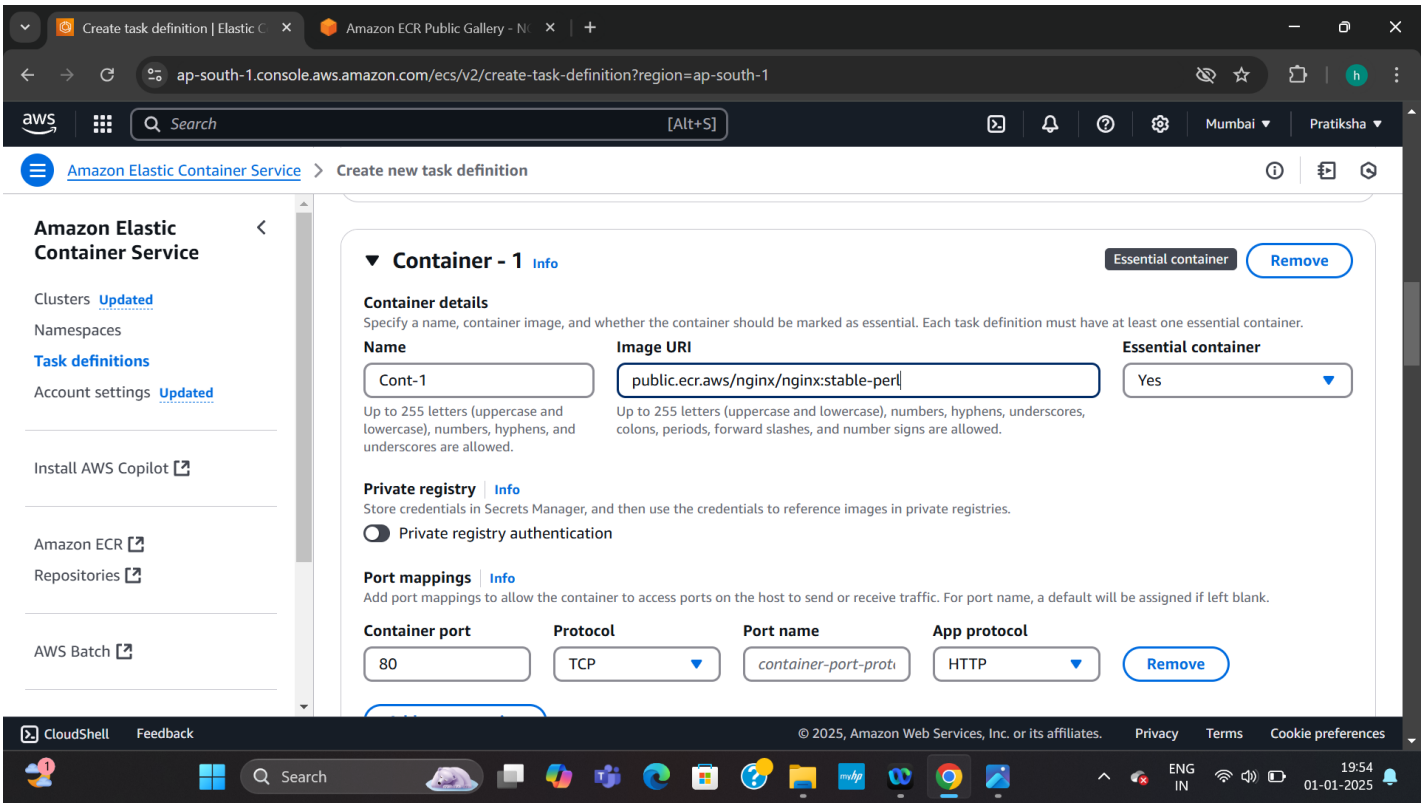

--> Select Create--> Go in Task defination--> crate new task defn 2--> give family name--> choose launch type as AWS farget



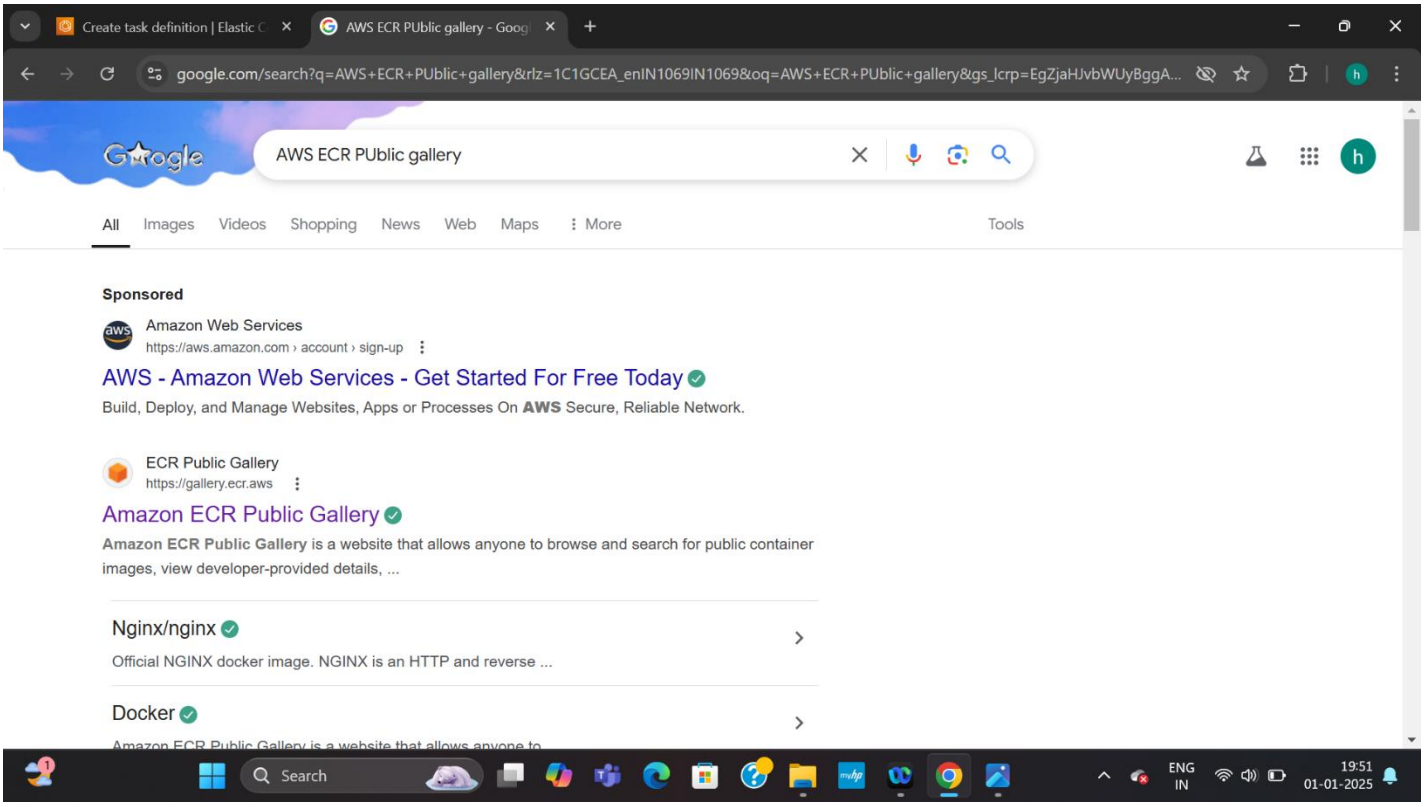

--> Inside Task size select 0.5CPU and 1 gb memory-->



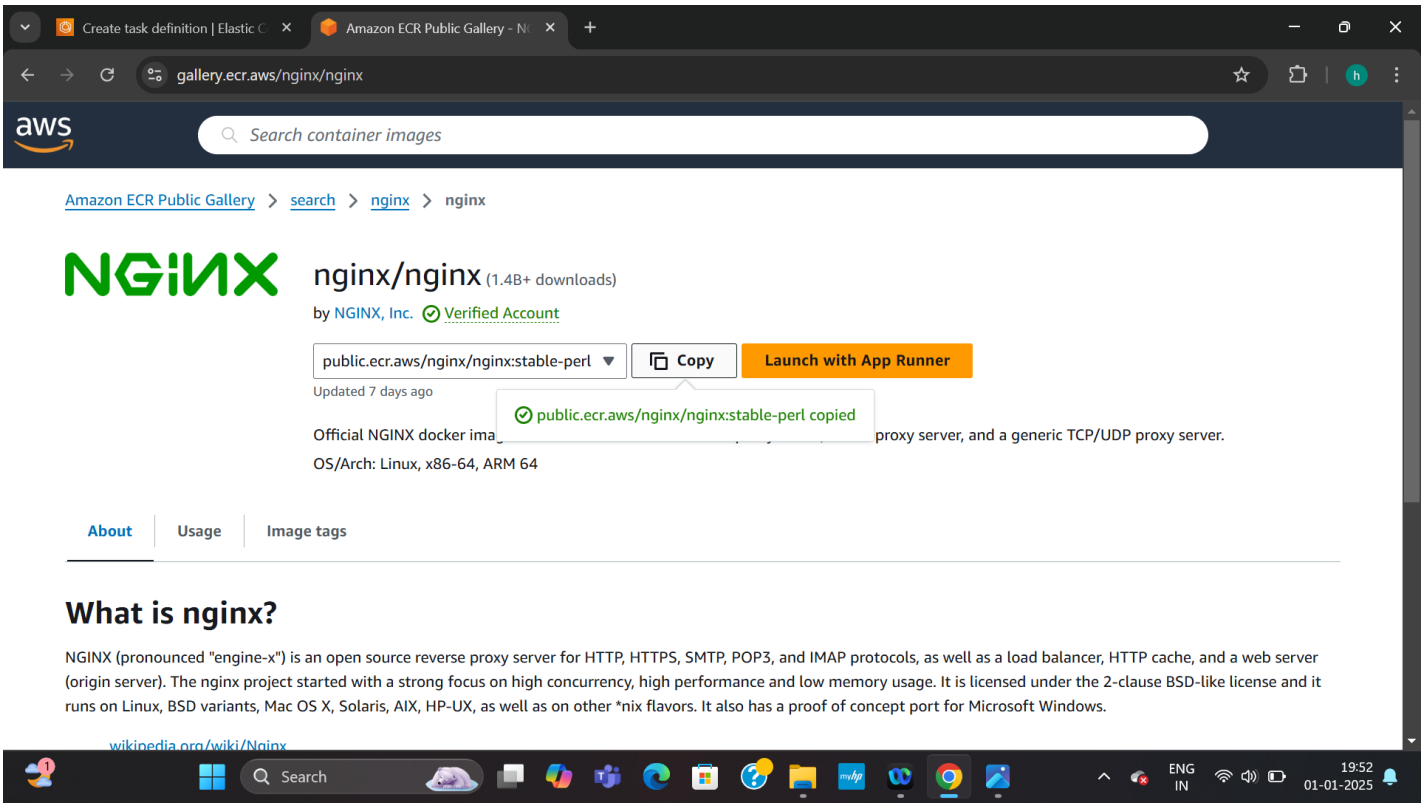

--> Inside Container-1 provide Container name



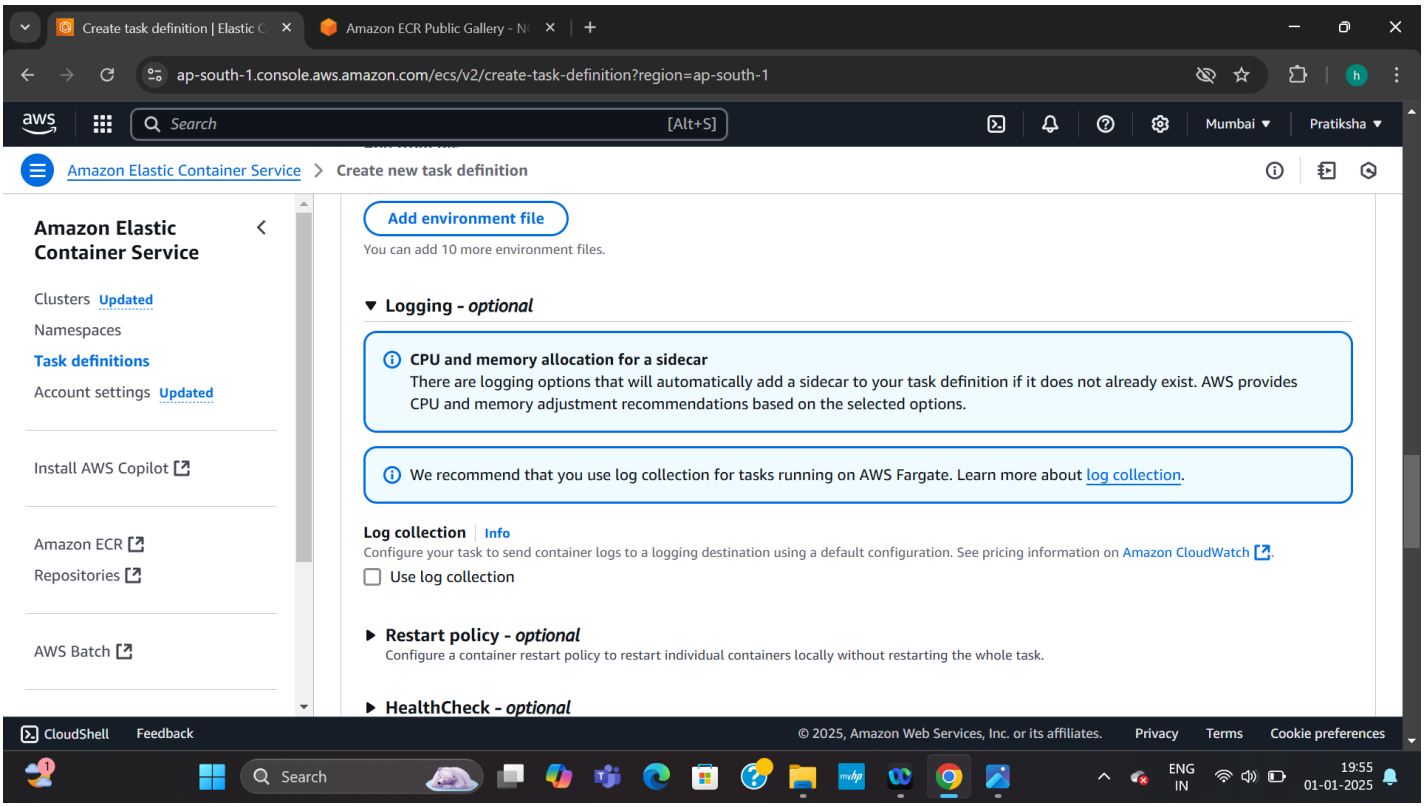

--> Search AWS ECR public Gallery



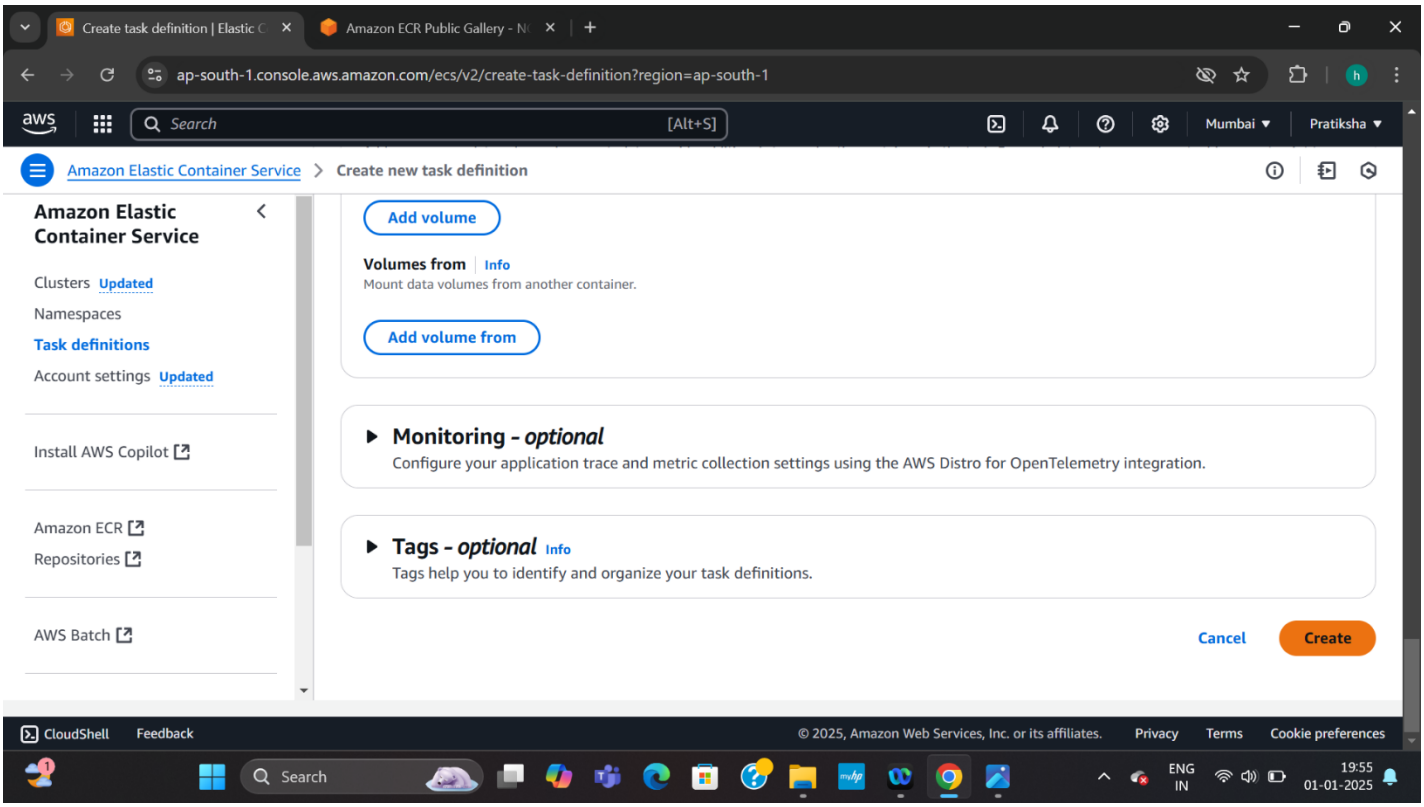

--> Select NGiNX--> select the first link--> COpy the URI--> and paste it in Image URI



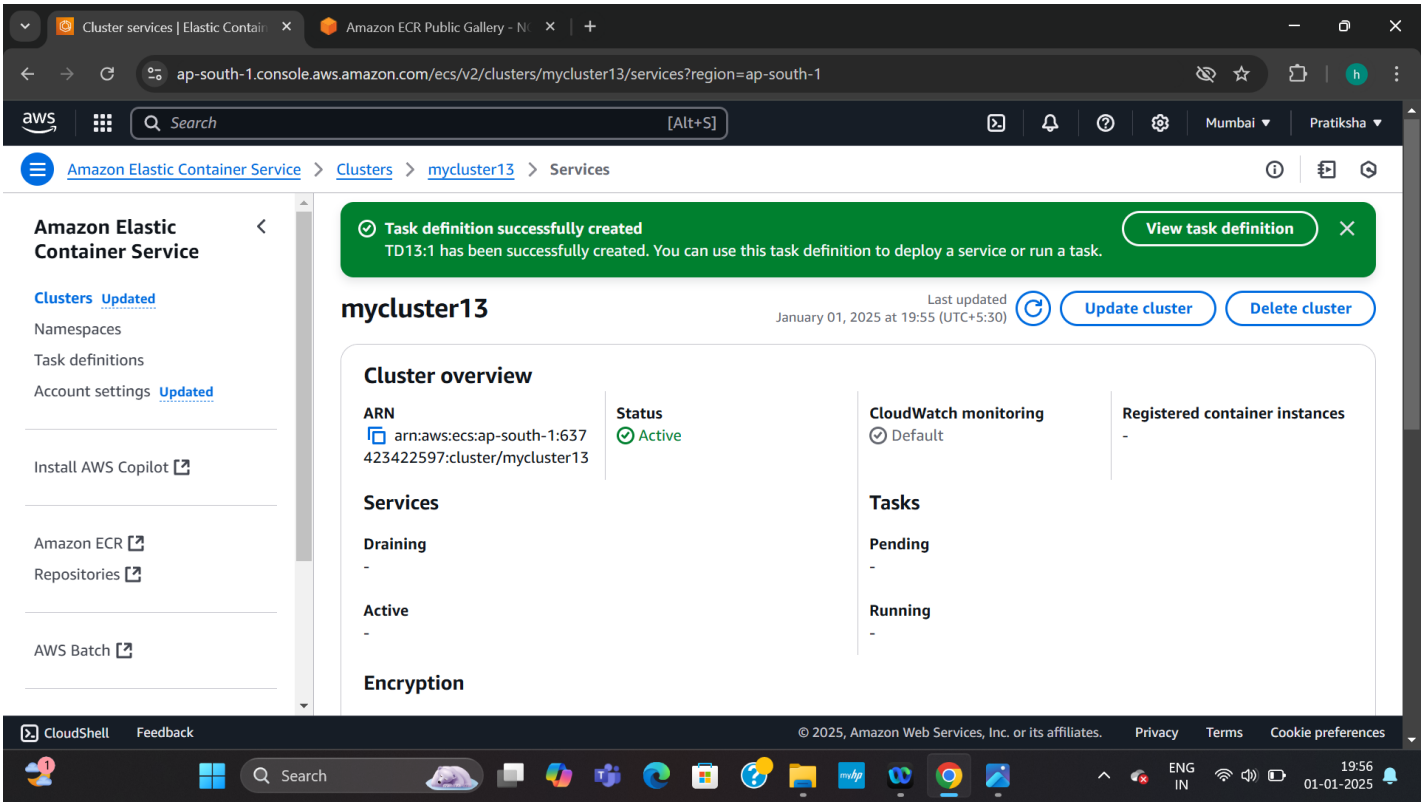

--> Under the logging untick use log collection



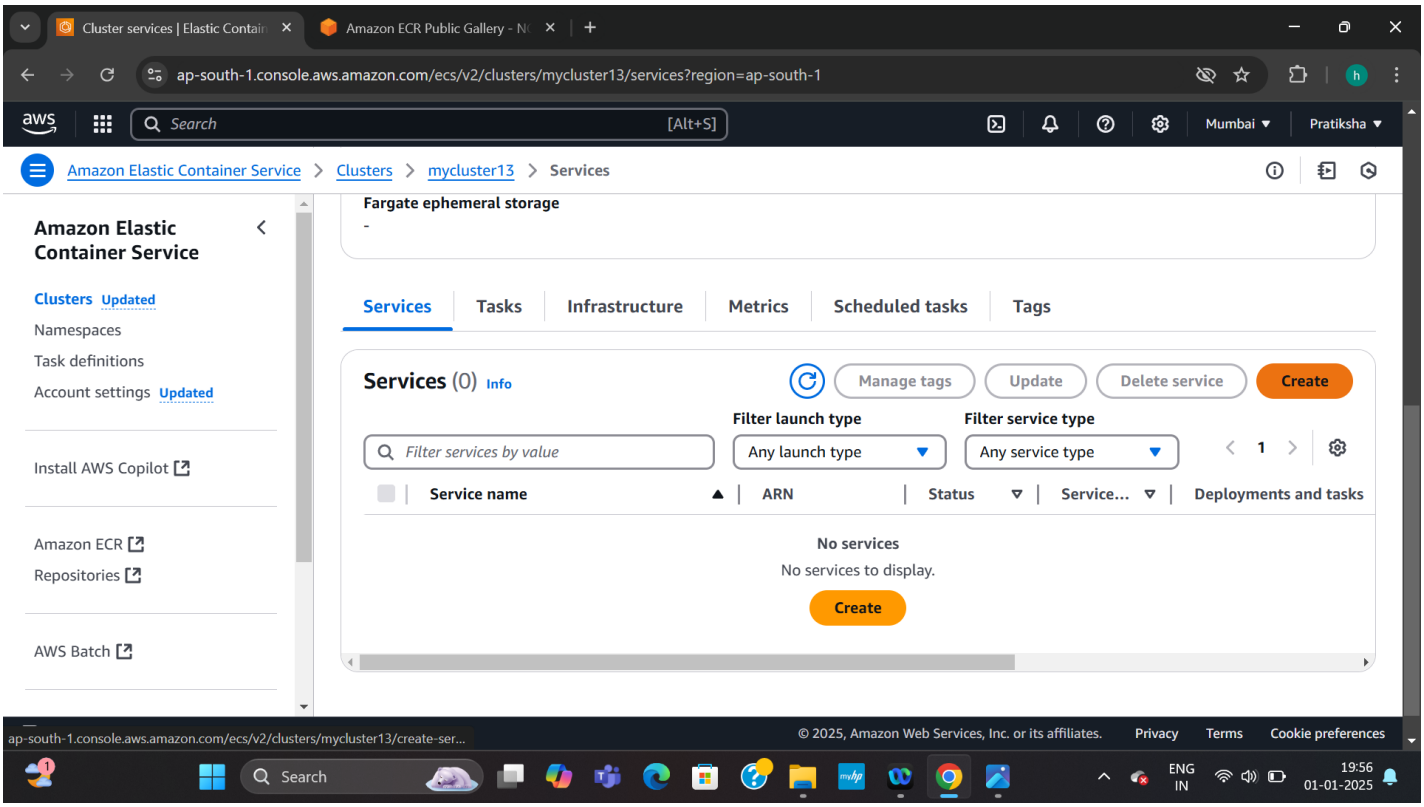

--> Create.



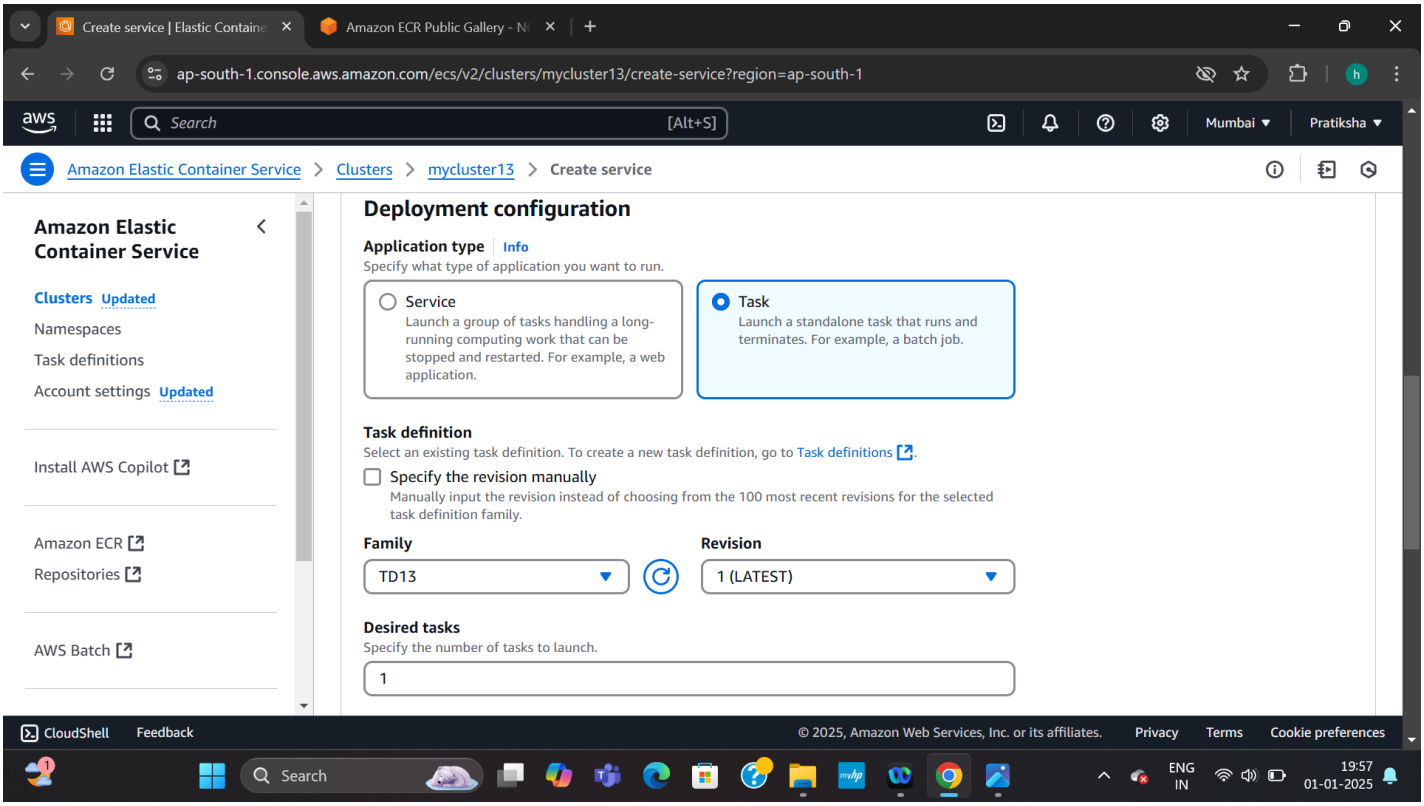

--> Go inside the Clustor name



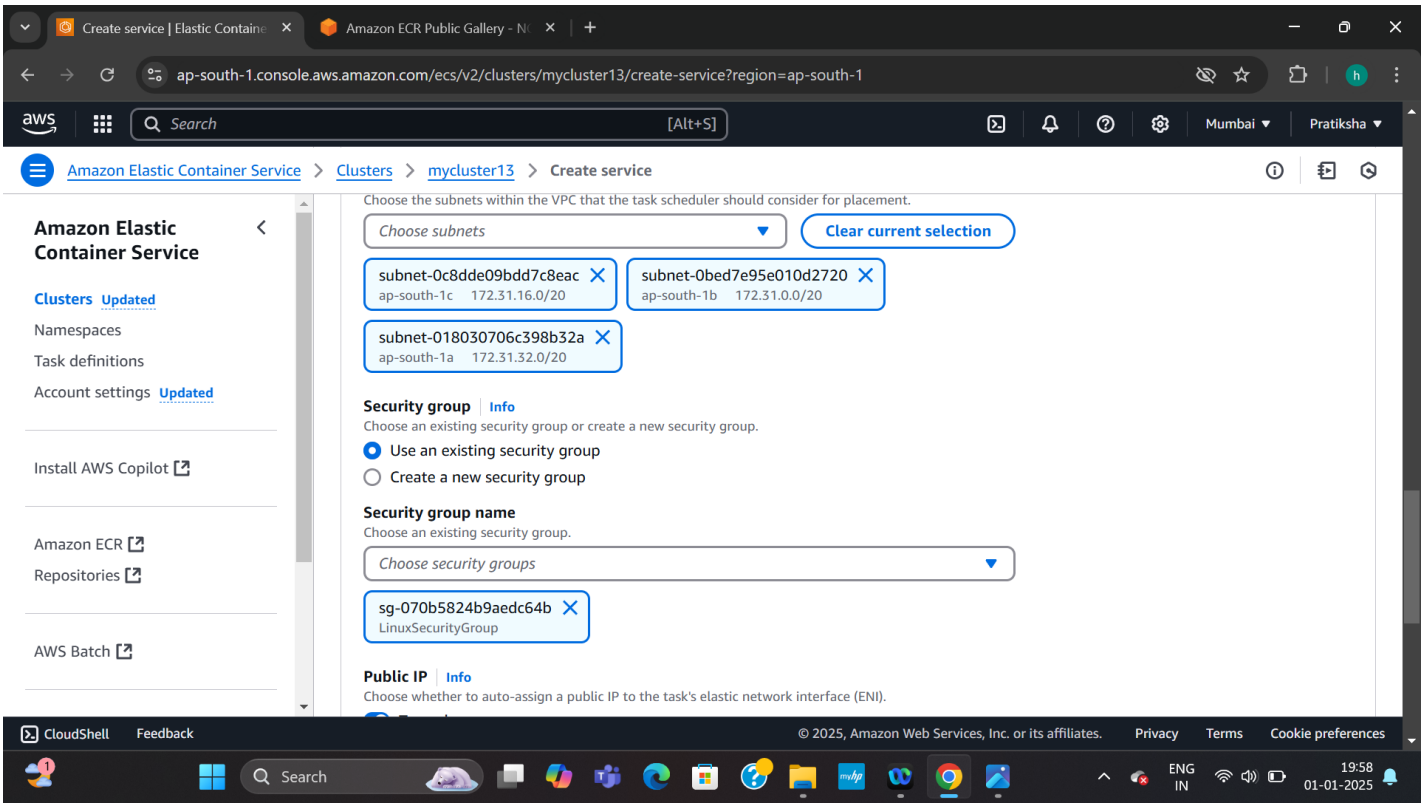

-> scroll below and select Create service



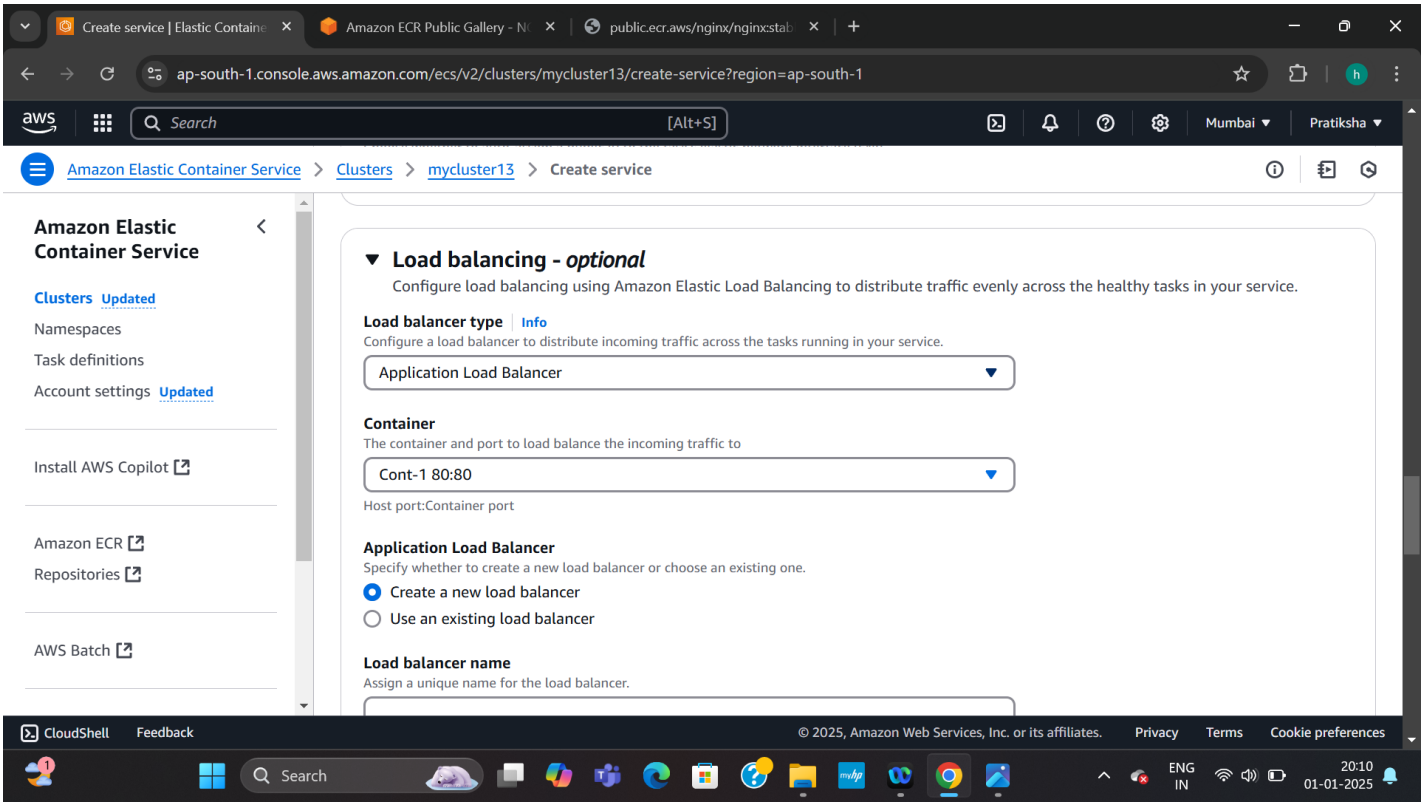

--> under deployment configuration select our task family



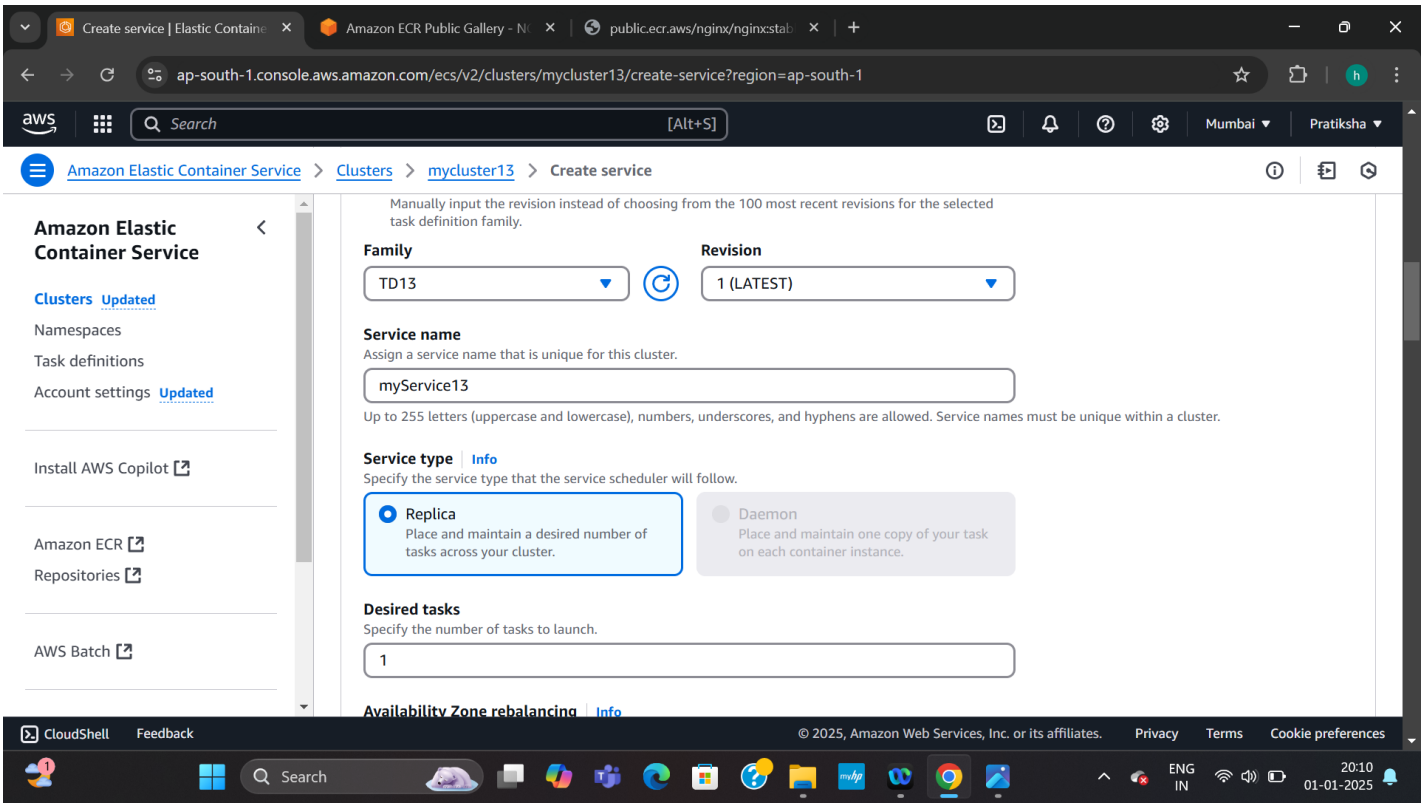

--> Provie Service Name--> Under Networking Select Linux Security Group



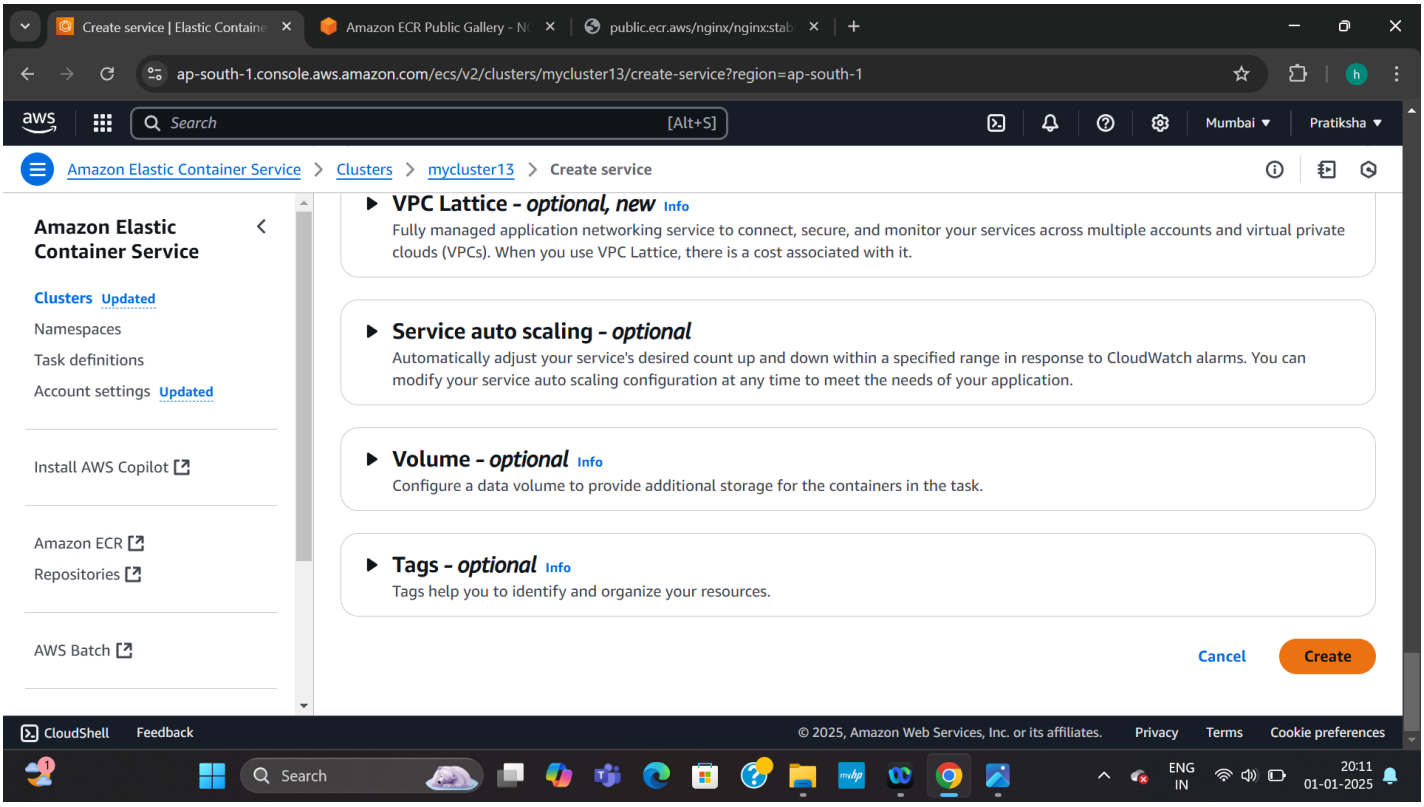

--> Expand Load balancer--> select Application Load balancer--> Provide Load balancer Name



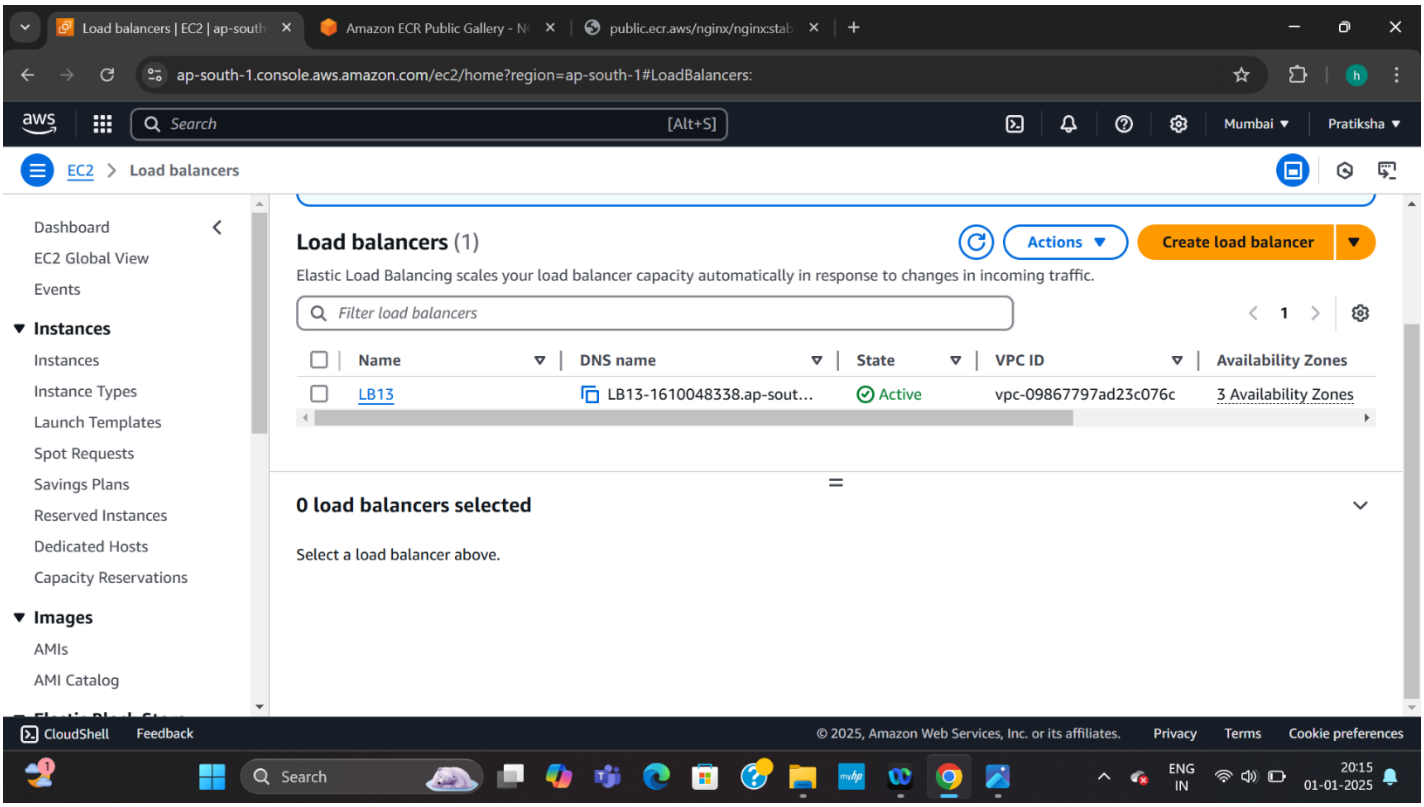

--> Provide Target Group name



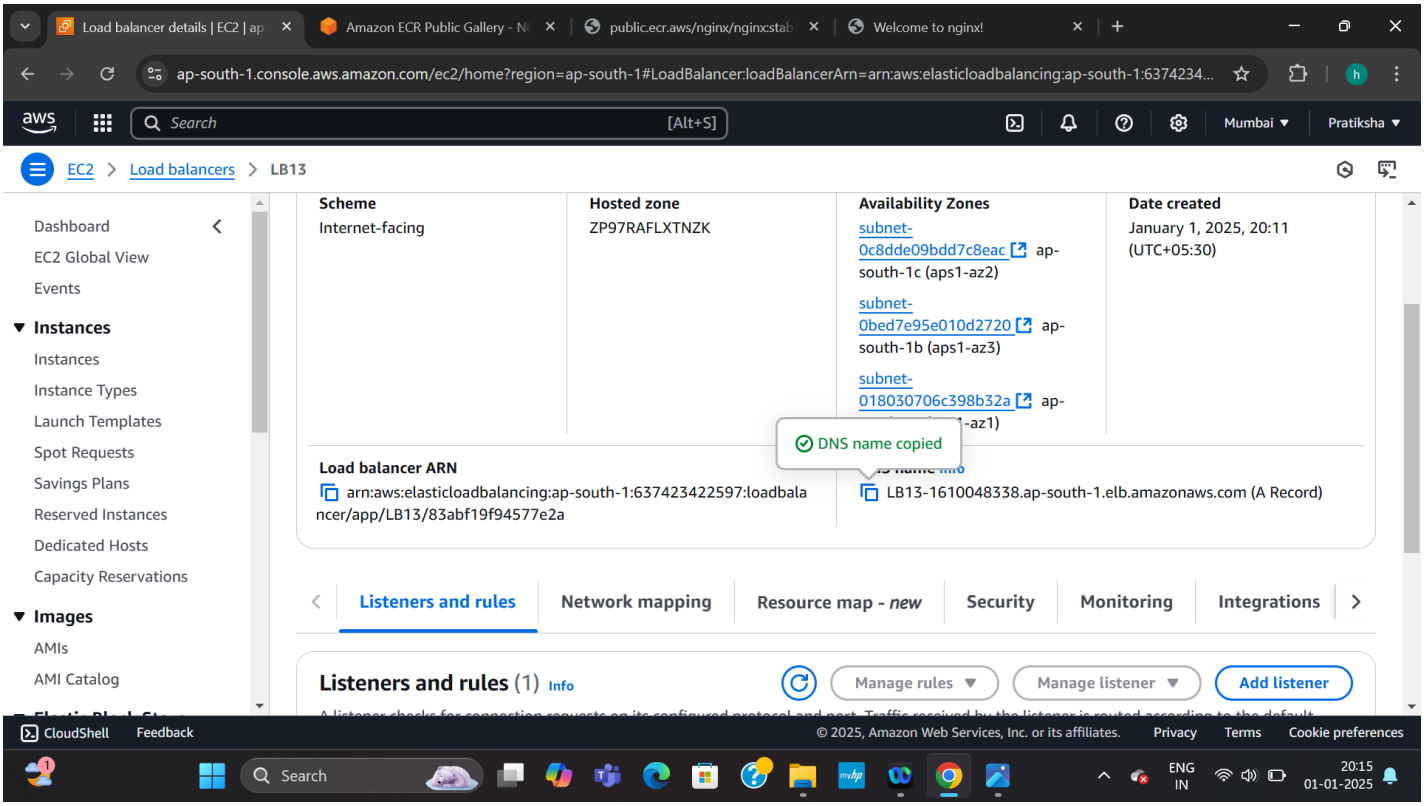

--> and create



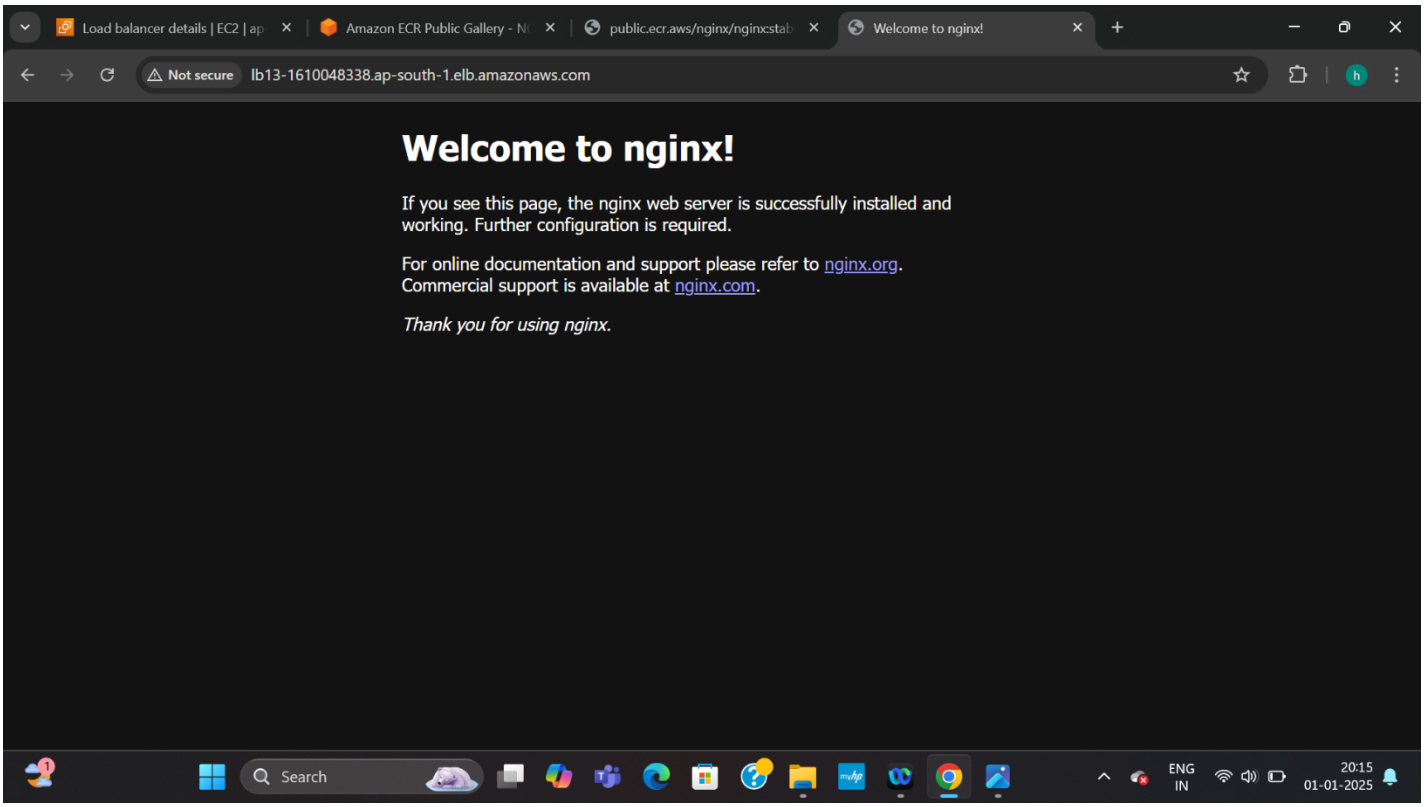

--> After Service is Created go in load balancer Name Inside EC2 Service




--> Copy the DNS name and paste it in New Tab




--> Here u will see the Welcome Page of nginix


Load balancer details | EC2 | ap
Amazon ECR Public Gallery - N
public.ecr.aws/nginx/nginx:stab
Welcome to nginx!
Not secure
lb13-1610048338.ap-south-1.elb.amazonaws.com
Welcome to nginx!
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.
Thank you for using nginx.
Search
ENG
IN
20:15
01-01-2025