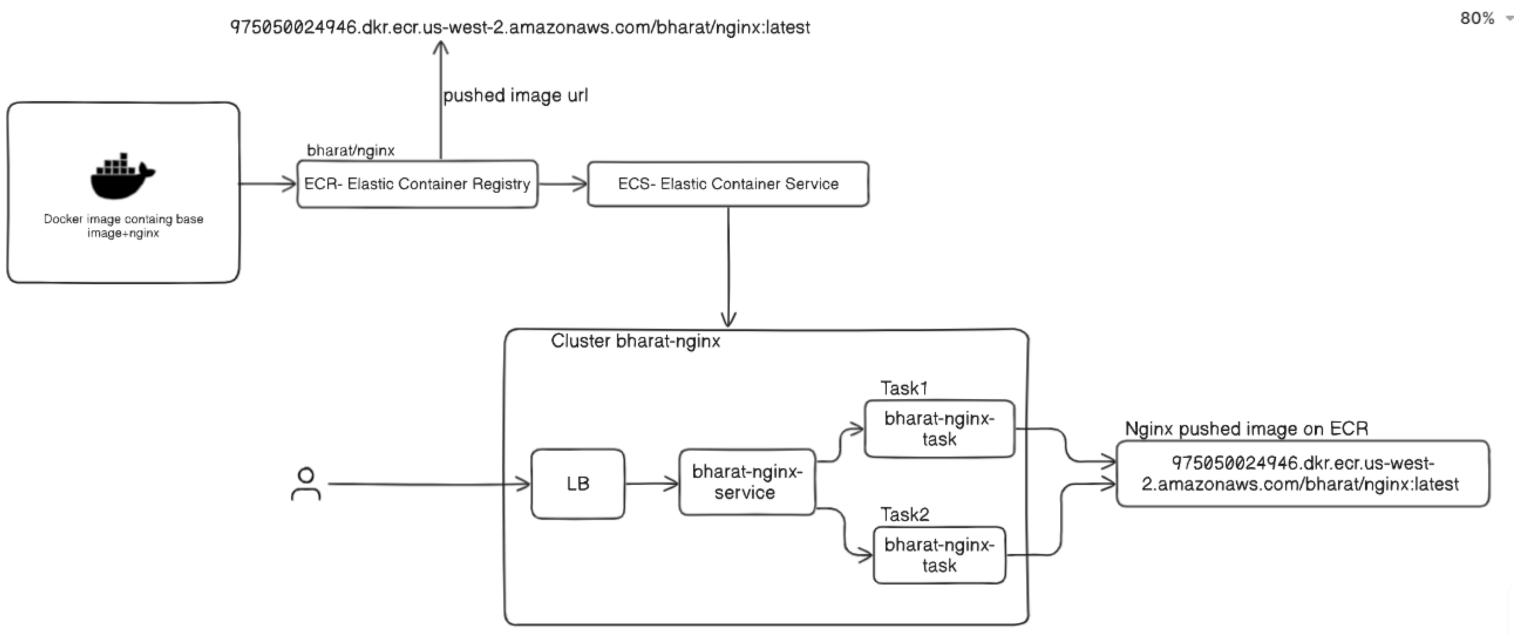
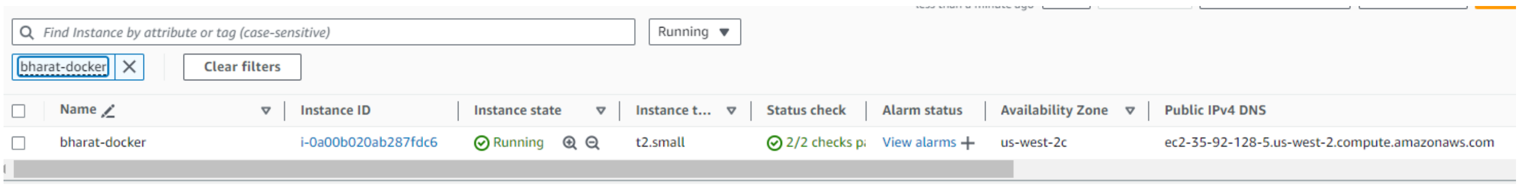
GitHub: https://github.com/devops-bharat05/Dockerizing-HTML

Diagram-Flow: https://app.eraser.io/workspace/H2FgUXbU9GDuCklien6E?origin=share



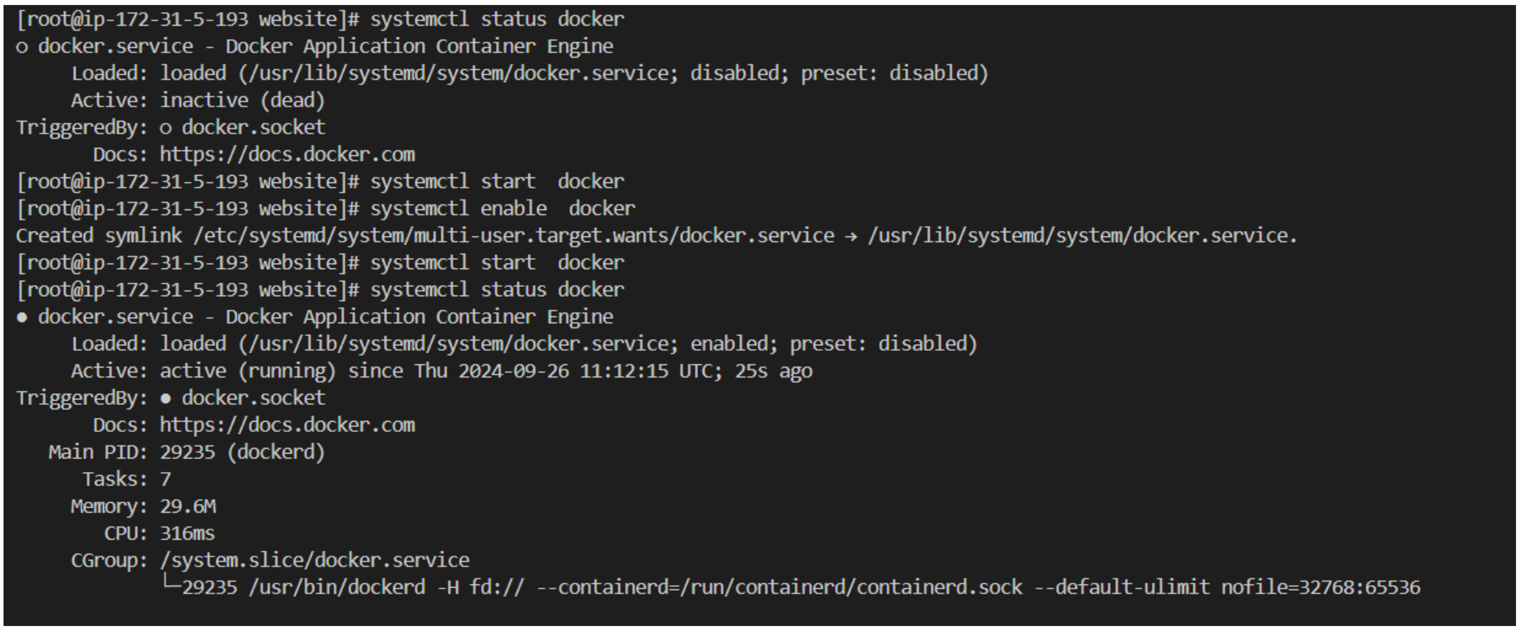
80%
975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx:latest
pushed image url
Docker image containg base image+nginx
bharat/nginx
ECR- Elastic Container Registry
ECS- Elastic Container Service
Cluster bharat-nginx
LB
bharat-nginx-service
Task1
bharat-nginx-task
Task2
bharat-nginx-task
Nginx pushed image on ECR
975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx:latest

- Create ec2 instance and install docker and aws cli
  - # sudo yum install docker –y
  - # curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  - # unzip awscliv2.zip
  - # sudo ./aws/install
  - # aws –version
  - # aws configure ## configure aws cli by creating token using IAM



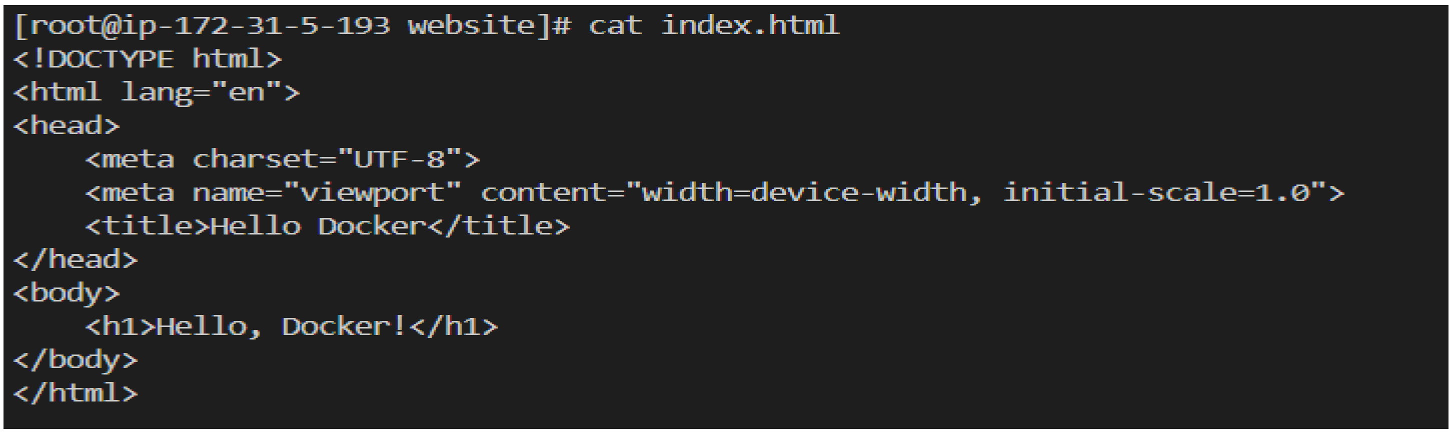
Name	Instance ID	Instance state	Instance t...	Status check	Alarm status	Availability Zone	Public IPv4 DNS
bharat-docker	i-0a00b020ab287fdc6	Running	t2.small	2/2 checks pa	View alarms +	us-west-2c	ec2-35-92-128-5.us-west-2.compute.amazonaws.com

- Make sure docker is running



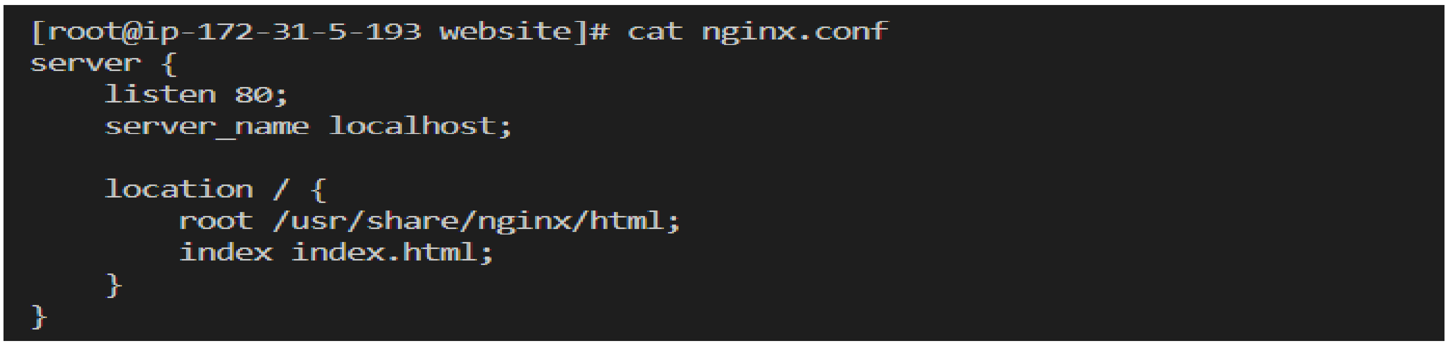
```
[root@ip-172-31-5-193 website]# systemctl status docker
○ docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
     Active: inactive (dead)
TriggeredBy: ○ docker.socket
       Docs: https://docs.docker.com
[root@ip-172-31-5-193 website]# systemctl start  docker
[root@ip-172-31-5-193 website]# systemctl enable  docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[root@ip-172-31-5-193 website]# systemctl start  docker
[root@ip-172-31-5-193 website]# systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: disabled)
     Active: active (running) since Thu 2024-09-26 11:12:15 UTC; 25s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 29235 (dockerd)
      Tasks: 7
     Memory: 29.6M
        CPU: 316ms
     CGroup: /system.slice/docker.service
             └─29235 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536
```

- Creating Basic HTML Page `index.html`



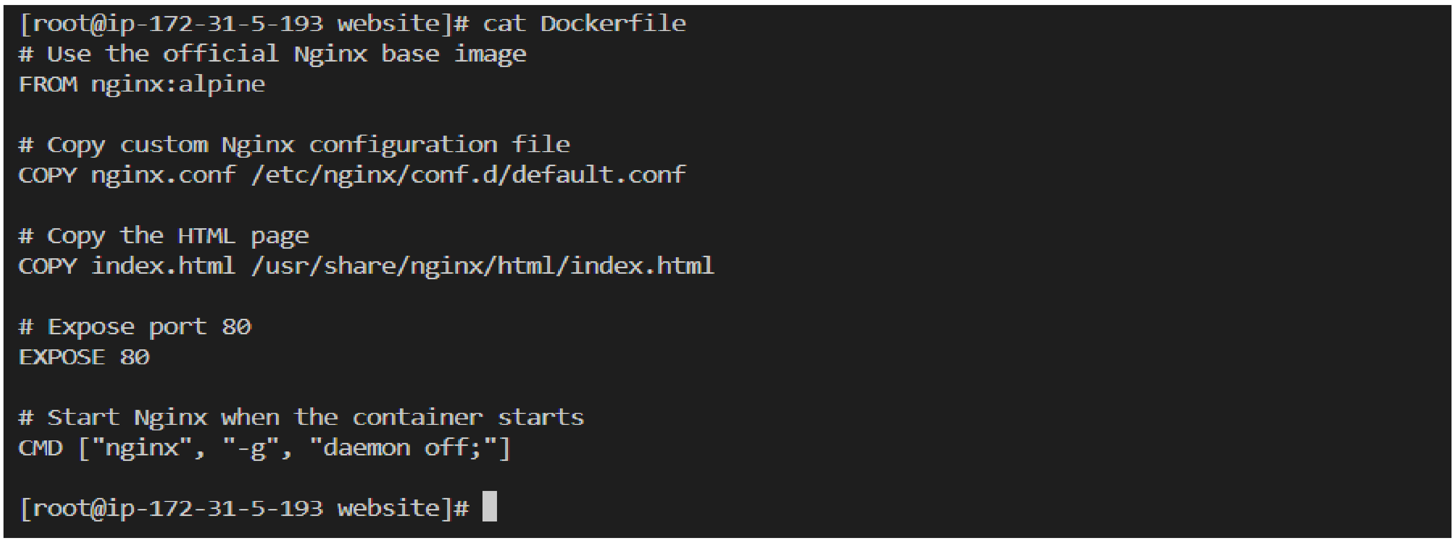
```
[root@ip-172-31-5-193 website]# cat index.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Hello Docker</title>
</head>
<body>
    <h1>Hello, Docker!</h1>
</body>
</html>
```

- Create the Nginx configuration file `nginx.conf` to serve the HTML page



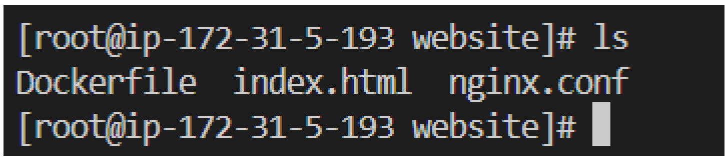
```
[root@ip-172-31-5-193 website]# cat nginx.conf
server {
    listen 80;
    server_name localhost;

    location / {
        root /usr/share/nginx/html;
        index index.html;
    }
}
```

- Create the `Dockerfile` to build the Docker image using the official Nginx base image

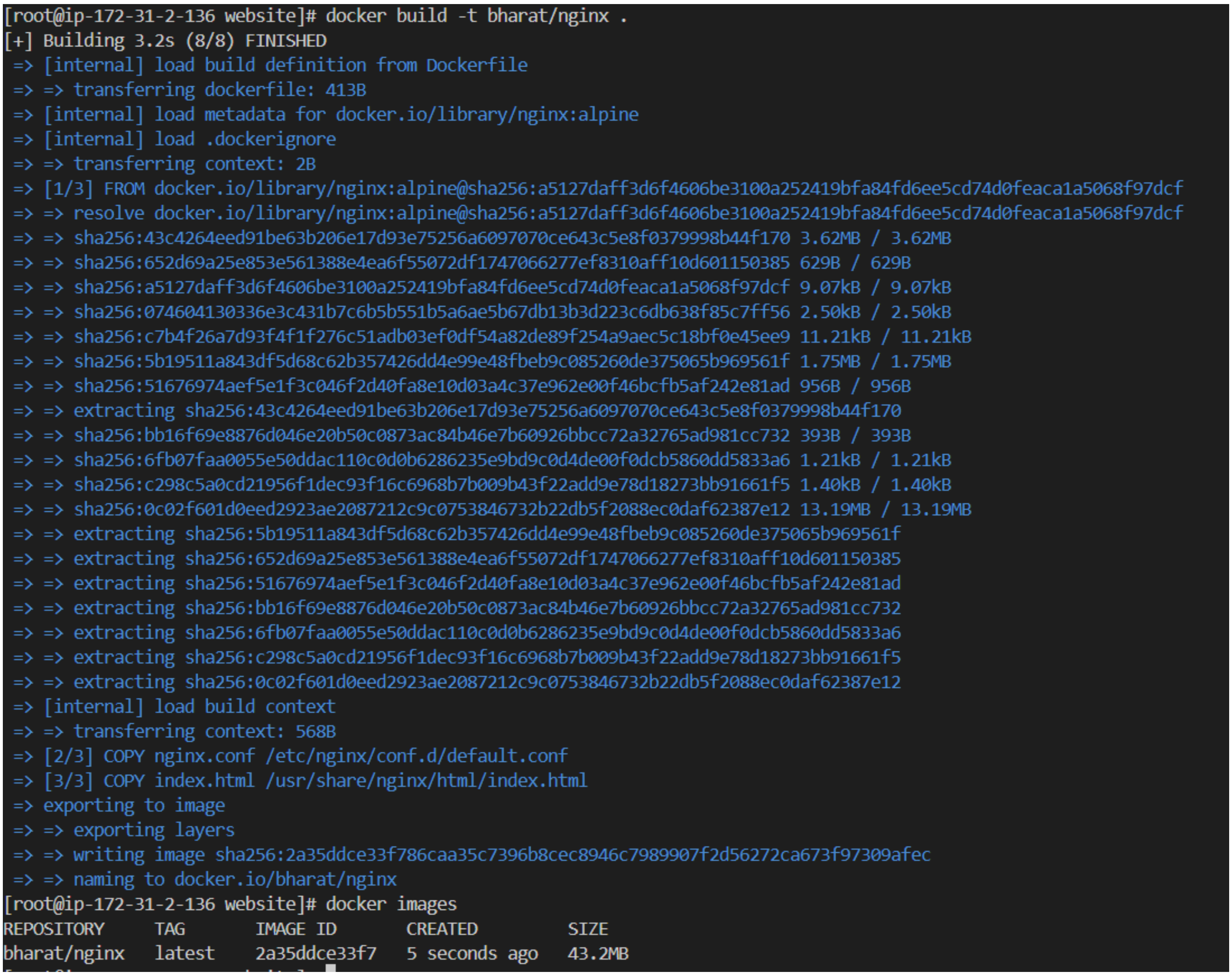


```
[root@ip-172-31-5-193 website]# cat Dockerfile
# Use the official Nginx base image
FROM nginx:alpine

# Copy custom Nginx configuration file
COPY nginx.conf /etc/nginx/conf.d/default.conf

# Copy the HTML page
COPY index.html /usr/share/nginx/html/index.html

# Expose port 80
EXPOSE 80

# Start Nginx when the container starts
CMD ["nginx", "-g", "daemon off;"]

[root@ip-172-31-5-193 website]#
```

- Build the Docker Image
    - # docker build -t my-nginx-app .

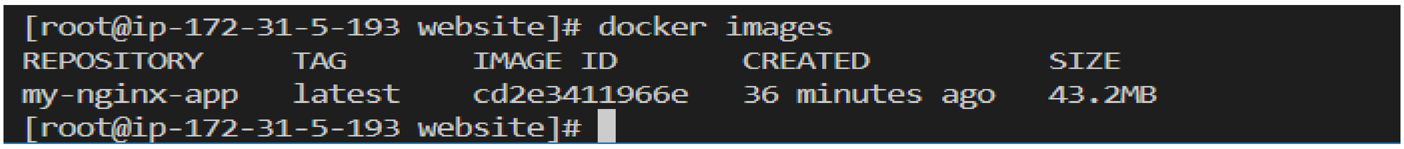


```
[root@ip-172-31-5-193 website]# ls
Dockerfile  index.html  nginx.conf
[root@ip-172-31-5-193 website]#
```



```
[root@ip-172-31-2-136 website]# docker build -t bharat/nginx .
[+] Building 3.2s (8/8) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 413B
 => [internal] load metadata for docker.io/library/nginx:alpine
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/3] FROM docker.io/library/nginx:alpine@sha256:a5127daff3d6f4606be3100a252419bfa84fd6ee5cd74d0feaca1a5068f97dcf
 => => resolve docker.io/library/nginx:alpine@sha256:a5127daff3d6f4606be3100a252419bfa84fd6ee5cd74d0feaca1a5068f97dcf
 => => sha256:43c4264eed91be63b206e17d93e75256a6097070ce643c5e8f0379998b44f170 3.62MB / 3.62MB
 => => sha256:652d69a25e853e561388e4ea6f55072df1747066277ef8310aff10d601150385 629B / 629B
 => => sha256:a5127daff3d6f4606be3100a252419bfa84fd6ee5cd74d0feaca1a5068f97dcf 9.07kB / 9.07kB
 => => sha256:074604130336e3c431b7c6b5b551b5a6ae5b67db13b3d223c6db638f85c7ff56 2.50kB / 2.50kB
 => => sha256:c7b4f26a7d93f4f1f276c51adb03ef0df54a82de89f254a9aec5c18bf0e45ee9 11.21kB / 11.21kB
 => => sha256:5b19511a843df5d68c62b357426dd4e99e48fbeb9c085260de375065b969561f 1.75MB / 1.75MB
 => => sha256:51676974aef5e1f3c046f2d40fa8e10d03a4c37e962e00f46bcfb5af242e81ad 956B / 956B
 => => extracting sha256:43c4264eed91be63b206e17d93e75256a6097070ce643c5e8f0379998b44f170
 => => sha256:bb16f69e8876d046e20b50c0873ac84b46e7b60926bbcc72a32765ad981cc732 393B / 393B
 => => sha256:6fb07faa0055e50ddac110c0d0b6286235e9bd9c0d4de00f0dcb5860dd5833a6 1.21kB / 1.21kB
 => => sha256:c298c5a0cd21956f1dec93f16c6968b7b009b43f22add9e78d18273bb91661f5 1.40kB / 1.40kB
 => => sha256:0c02f601d0eed2923ae2087212c9c0753846732b22db5f2088ec0daf62387e12 13.19MB / 13.19MB
 => => extracting sha256:5b19511a843df5d68c62b357426dd4e99e48fbeb9c085260de375065b969561f
 => => extracting sha256:652d69a25e853e561388e4ea6f55072df1747066277ef8310aff10d601150385
 => => extracting sha256:51676974aef5e1f3c046f2d40fa8e10d03a4c37e962e00f46bcfb5af242e81ad
 => => extracting sha256:bb16f69e8876d046e20b50c0873ac84b46e7b60926bbcc72a32765ad981cc732
 => => extracting sha256:6fb07faa0055e50ddac110c0d0b6286235e9bd9c0d4de00f0dcb5860dd5833a6
 => => extracting sha256:c298c5a0cd21956f1dec93f16c6968b7b009b43f22add9e78d18273bb91661f5
 => => extracting sha256:0c02f601d0eed2923ae2087212c9c0753846732b22db5f2088ec0daf62387e12
 => [internal] load build context
 => => transferring context: 568B
 => [2/3] COPY nginx.conf /etc/nginx/conf.d/default.conf
 => [3/3] COPY index.html /usr/share/nginx/html/index.html
 => exporting to image
 => => exporting layers
 => => writing image sha256:2a35ddce33f786caa35c7396b8cec8946c7989907f2d56272ca673f97309afec
 => => naming to docker.io/bharat/nginx
[root@ip-172-31-2-136 website]# docker images
REPOSITORY     TAG       IMAGE ID       CREATED         SIZE
bharat/nginx   latest    2a35ddce33f7   5 seconds ago   43.2MB
```

- Check for image in your local



```
[root@ip-172-31-5-193 website]# docker images
REPOSITORY     TAG       IMAGE ID       CREATED          SIZE
my-nginx-app   latest    cd2e3411966e   36 minutes ago   43.2MB
[root@ip-172-31-5-193 website]#
```

- Now go the AWS ECR and
  - create your repo
  - click on view push commands to get the steps to link to ec2 instances

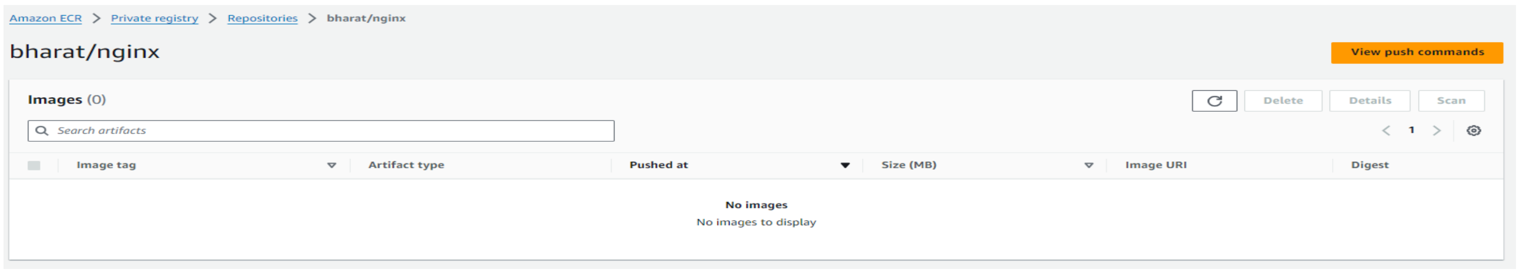
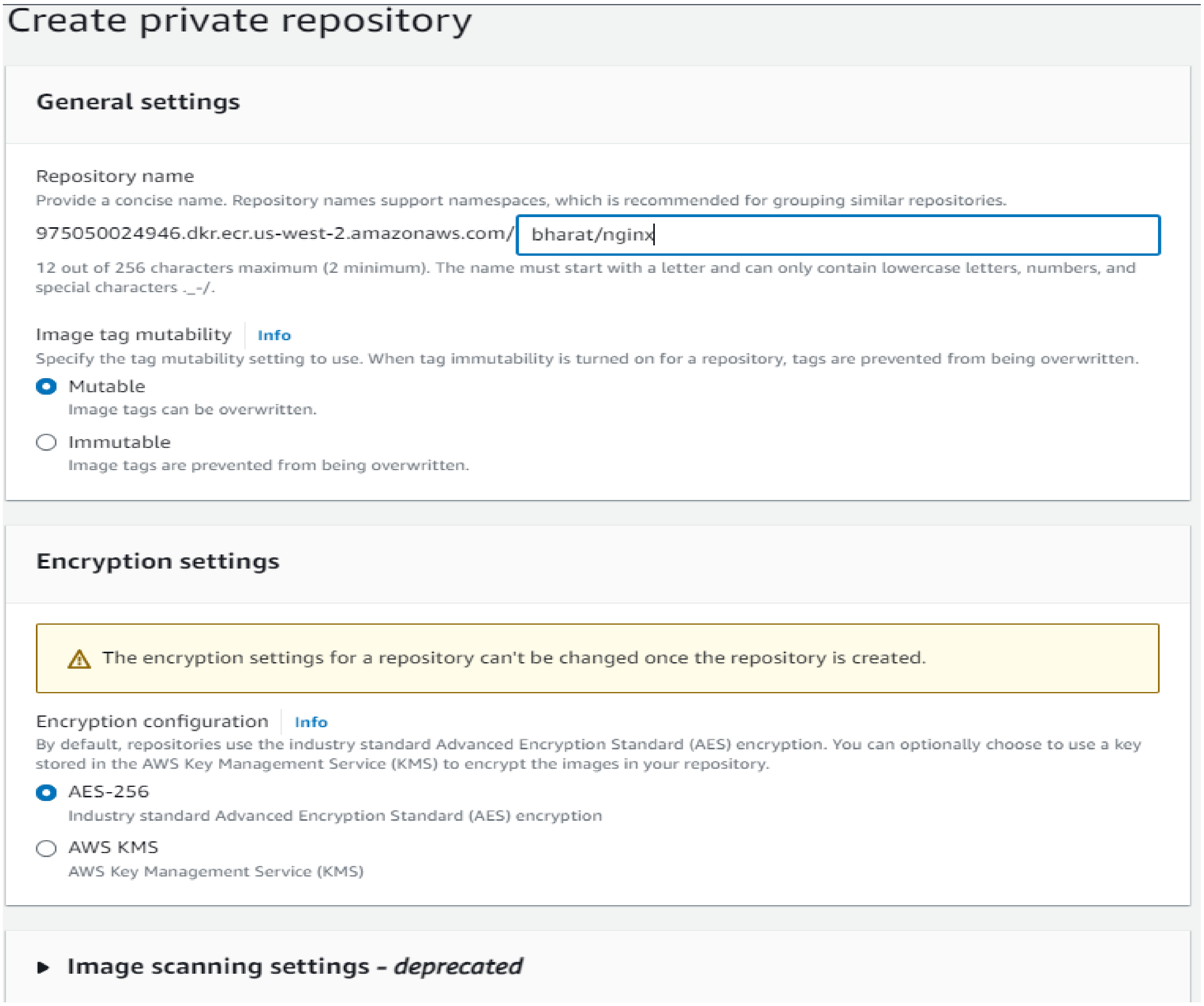


Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest

No images
No images to display

  - # aws ecr get-login-password --region us-west-2 | docker login --username AWS --password-stdin 975050024946.dkr.ecr.us-west-2.amazonaws.com
  - # docker build -t bharat/nginx .
  - # docker tag bharat/nginx:latest 975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx:latest
  - # docker push 975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx:latest



Create private repository

General settings

Repository name

Provide a concise name. Repository names support namespaces, which is recommended for grouping similar repositories.

975050024946.dkr.ecr.us-west-2.amazonaws.com/ bharat/nginx

12 out of 256 characters maximum (2 minimum). The name must start with a letter and can only contain lowercase letters, numbers, and special characters ._-/.

Image tag mutability | Info

Specify the tag mutability setting to use. When tag immutability is turned on for a repository, tags are prevented from being overwritten.

[✓ Mutable]
Image tags can be overwritten.

Immutable
Image tags are prevented from being overwritten.

Encryption settings

⚠ The encryption settings for a repository can't be changed once the repository is created.

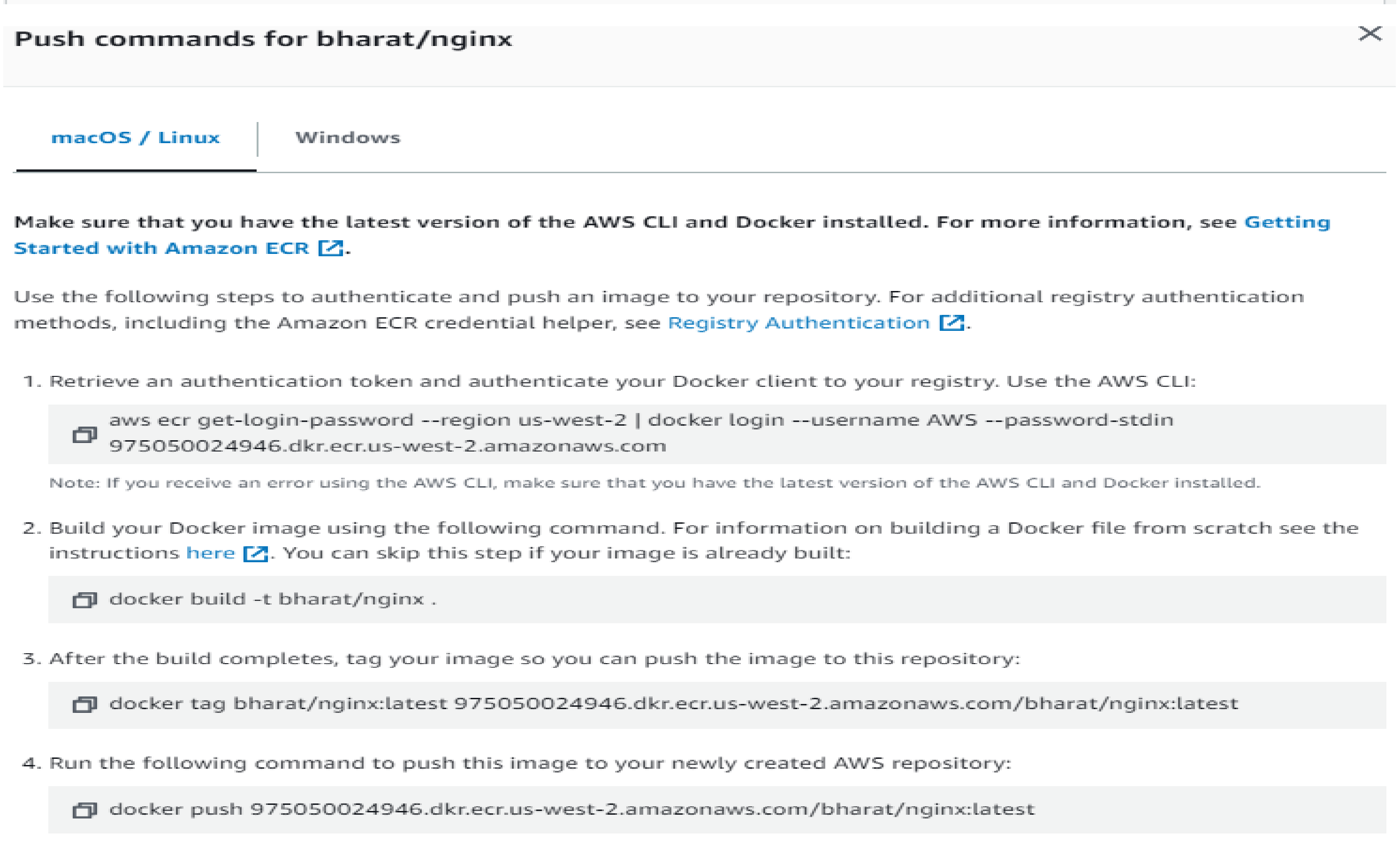
Encryption configuration | Info

By default, repositories use the industry standard Advanced Encryption Standard (AES) encryption. You can optionally choose to use a key stored in the AWS Key Management Service (KMS) to encrypt the images in your repository.

[✓ AES-256]
Industry standard Advanced Encryption Standard (AES) encryption

AWS KMS
AWS Key Management Service (KMS)

▶ **Image scanning settings -** ***deprecated***



Push commands for bharat/nginx

Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see Getting 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 us-west-2 | docker login --username AWS --password-stdin
975050024946.dkr.ecr.us-west-2.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:

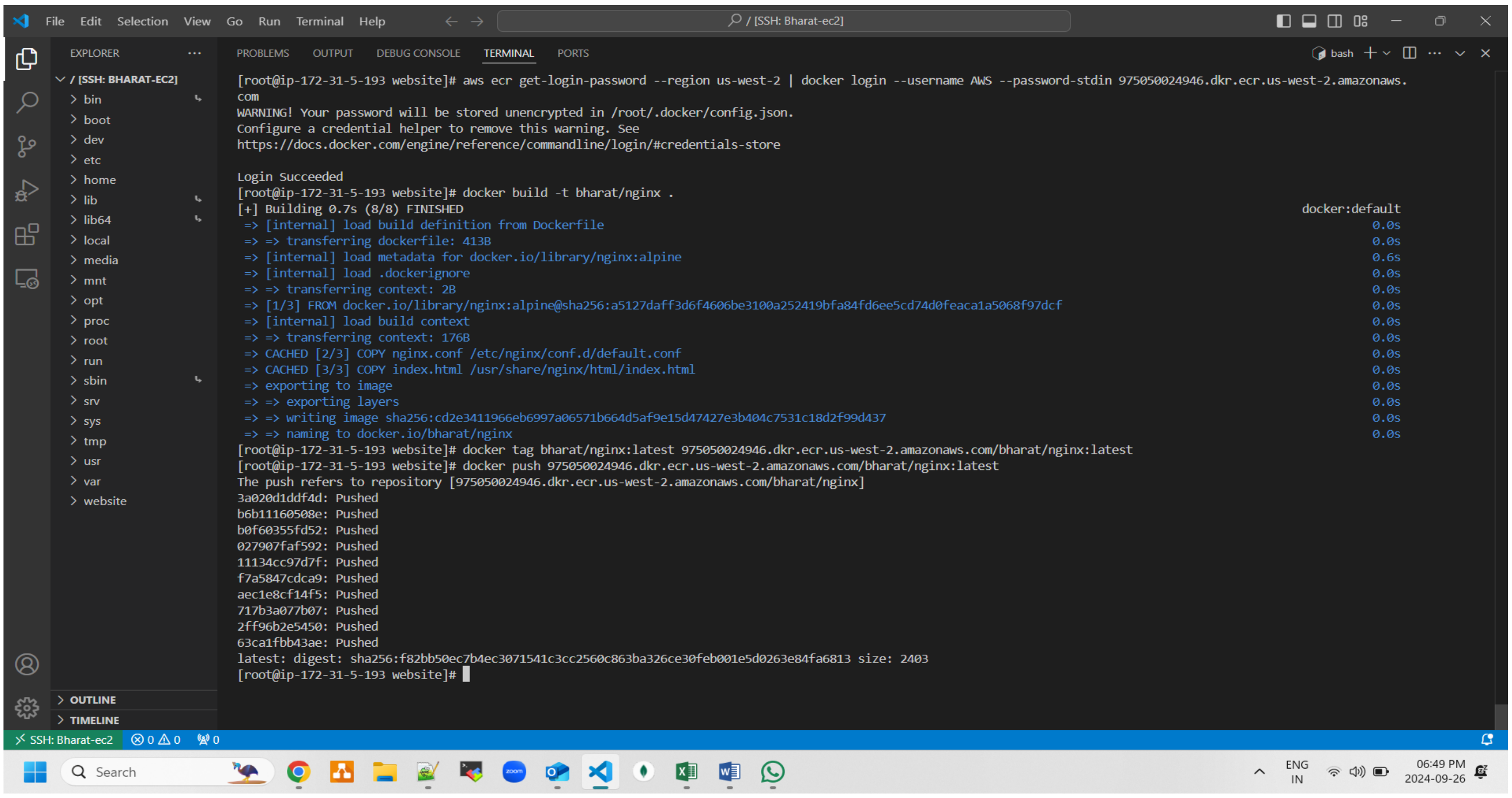
```
docker build -t bharat/nginx .
```

3. After the build completes, tag your image so you can push the image to this repository:

```
docker tag bharat/nginx:latest 975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx:latest
```

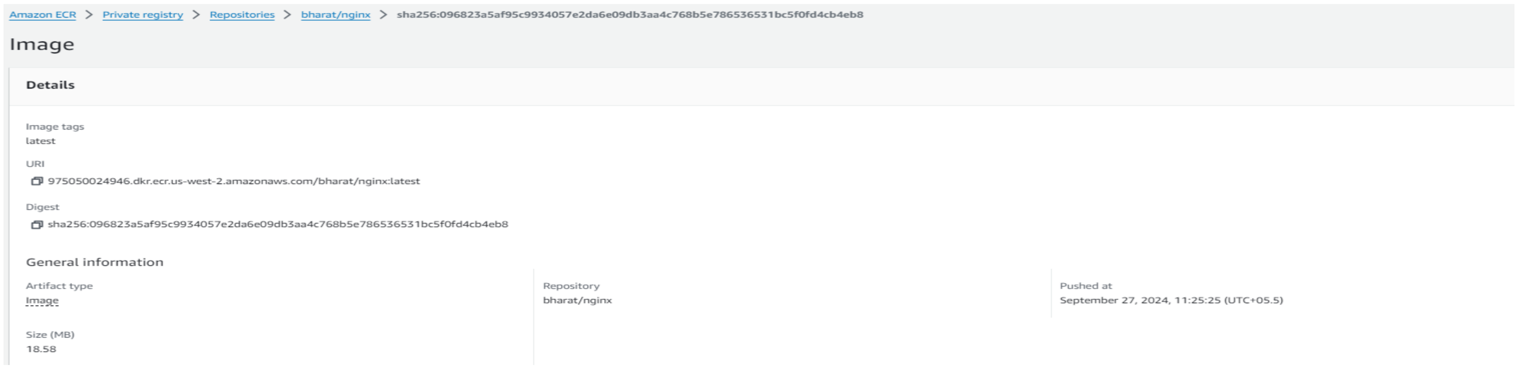
4. Run the following command to push this image to your newly created AWS repository:

```
docker push 975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx:latest
```



```
[root@ip-172-31-5-193 website]# aws ecr get-login-password --region us-west-2 | docker login --username AWS --password-stdin 975050024946.dkr.ecr.us-west-2.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-172-31-5-193 website]# docker build -t bharat/nginx .
[+] Building 0.7s (8/8) FINISHED                                                                          docker:default
 => [internal] load build definition from Dockerfile                                                               0.0s
 => => transferring dockerfile: 413B                                                                               0.0s
 => [internal] load metadata for docker.io/library/nginx:alpine                                                    0.6s
 => [internal] load .dockerignore                                                                                  0.0s
 => => transferring context: 2B                                                                                    0.0s
 => [1/3] FROM docker.io/library/nginx:alpine@sha256:a5127daff3d6f4606be3100a252419bfa84fd6ee5cd74d0feaca1a5068f97dcf  0.0s
 => [internal] load build context                                                                                  0.0s
 => => transferring context: 176B                                                                                  0.0s
 => CACHED [2/3] COPY nginx.conf /etc/nginx/conf.d/default.conf                                                    0.0s
 => CACHED [3/3] COPY index.html /usr/share/nginx/html/index.html                                                  0.0s
 => exporting to image                                                                                             0.0s
 => => exporting layers                                                                                            0.0s
 => => writing image sha256:cd2e3411966eb6997a06571b664d5af9e15d47427e3b404c7531c18d2f99d437                       0.0s
 => => naming to docker.io/bharat/nginx                                                                            0.0s
[root@ip-172-31-5-193 website]# docker tag bharat/nginx:latest 975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx:latest
[root@ip-172-31-5-193 website]# docker push 975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx:latest
The push refers to repository [975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx]
3a020d1ddf4d: Pushed
b6b11160508e: Pushed
b0f60355fd52: Pushed
027907faf592: Pushed
11134cc97d7f: Pushed
f7a5847cdca9: Pushed
aec1e8cf14f5: Pushed
717b3a077b07: Pushed
2ff96b2e5450: Pushed
63ca1fbb43ae: Pushed
latest: digest: sha256:f82bb50ec7b4ec3071541c3cc2560c863ba326ce30feb001e5d0263e84fa6813 size: 2403
[root@ip-172-31-5-193 website]# 
```

➢ After pushing image to ECR



Amazon ECR > Private registry > Repositories > bharat/nginx > sha256:096823a5af95c9934057e2da6e09db3aa4c768b5e786536531bc5f0fd4cb4eb8

Image

Details

Image tags
latest

URI
975050024946.dkr.ecr.us-west-2.amazonaws.com/bharat/nginx:latest

Digest
sha256:096823a5af95c9934057e2da6e09db3aa4c768b5e786536531bc5f0fd4cb4eb8

General information

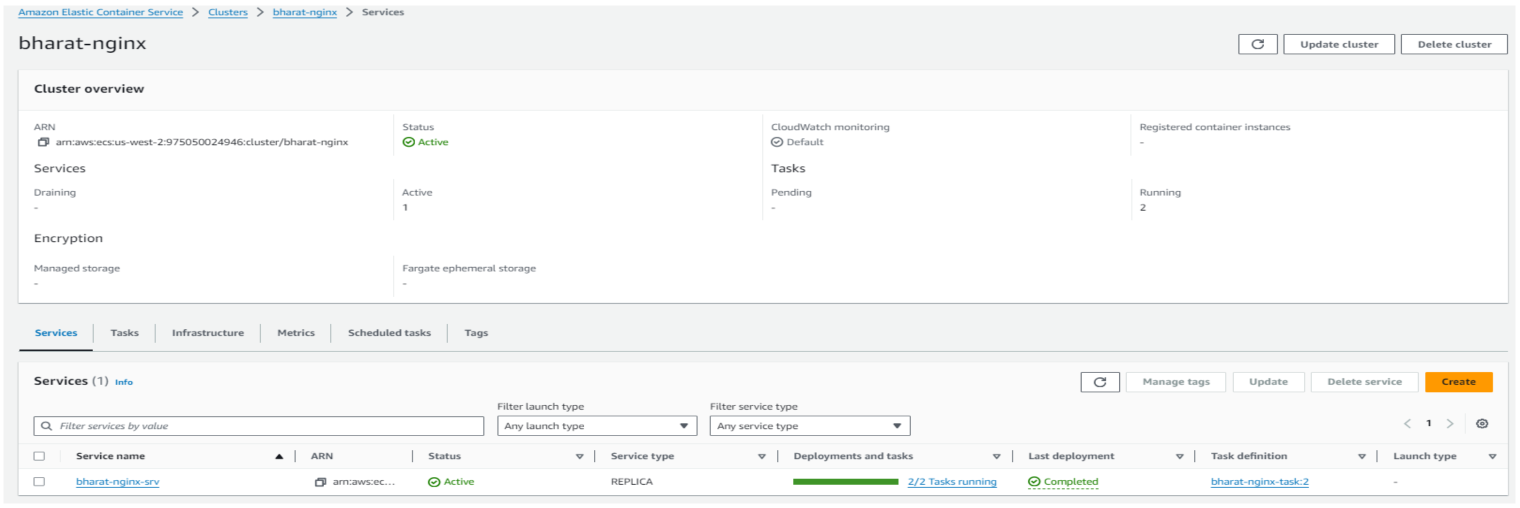
Artifact type
Image

Repository
bharat/nginx

Pushed at
September 27, 2024, 11:25:25 (UTC+05.5)

Size (MB)
18.58

➢ Now go to ECS (Elastic Container Service) and create cluster



Amazon Elastic Container Service > Clusters > bharat-nginx > Services

bharat-nginx

Cluster overview

ARN arn:aws:ecs:us-west-2:975050024946:cluster/bharat-nginx	Status Active	CloudWatch monitoring Default	Registered container instances -

Services

Draining -	Active 1

Tasks

Pending -	Running 2

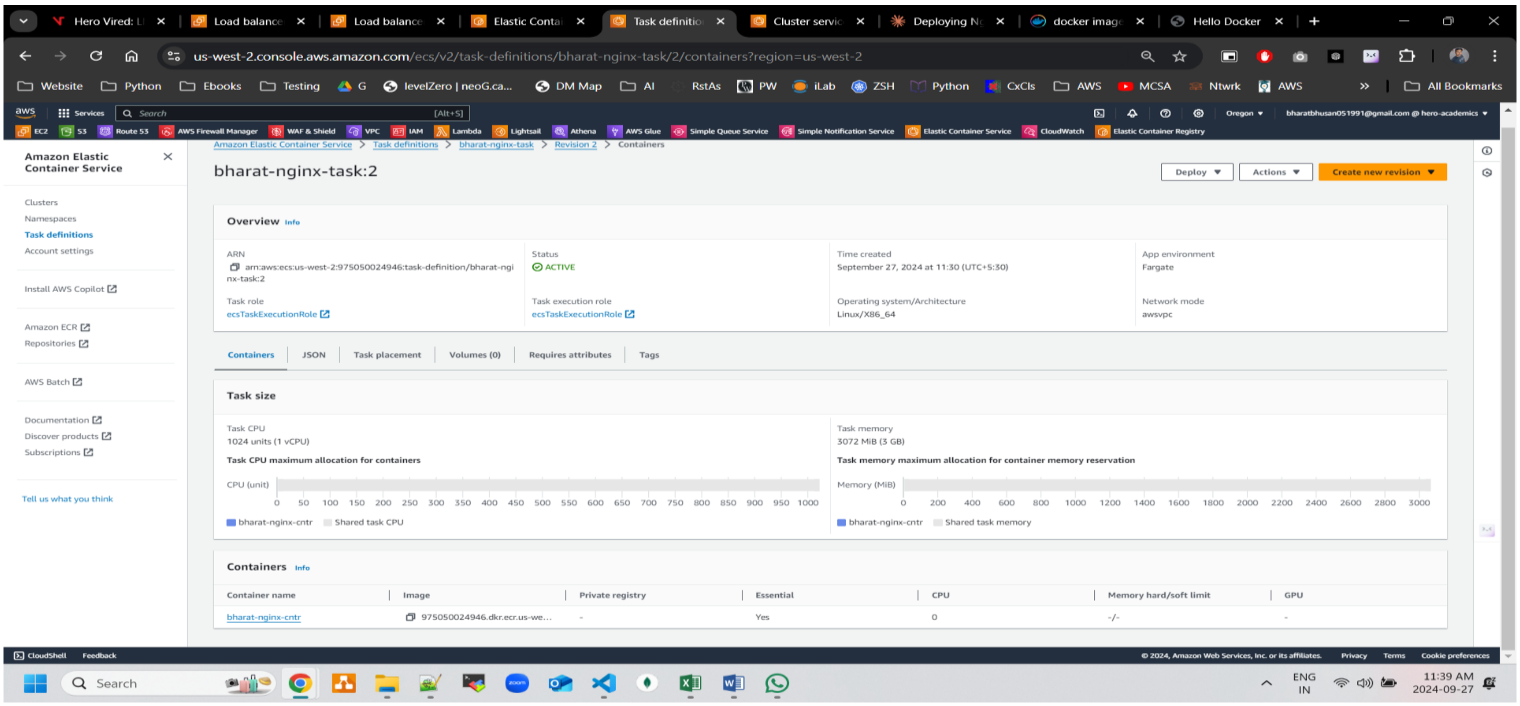
Encryption

Managed storage -	Fargate ephemeral storage -

Services (1) Info

Service name	ARN	Status	Service type	Deployments and tasks	Last deployment	Task definition	Launch type
bharat-nginx-srv	arn:aws:ec...	Active	REPLICA	2/2 Tasks running	Completed	bharat-nginx-task:2	-

➢ Create a task


us-west-2.console.aws.amazon.com/ecs/v2/task-definitions/bharat-nginx-task/2/containers?region=us-west-2

Amazon Elastic Container Service > Task definitions > bharat-nginx-task > Revision 2 > Containers

bharat-nginx-task:2

Overview Info

ARN
arn:aws:ecs:us-west-2:975050024946:task-definition/bharat-nginx-task:2

Status
ACTIVE

Time created
September 27, 2024 at 11:30 (UTC+5:30)

App environment
Fargate

Task role
ecsTaskExecutionRole

Task execution role
ecsTaskExecutionRole

Operating system/Architecture
Linux/X86_64

Network mode
awsvpc

Task size

Task CPU
1024 units (1 vCPU)

Task CPU maximum allocation for containers

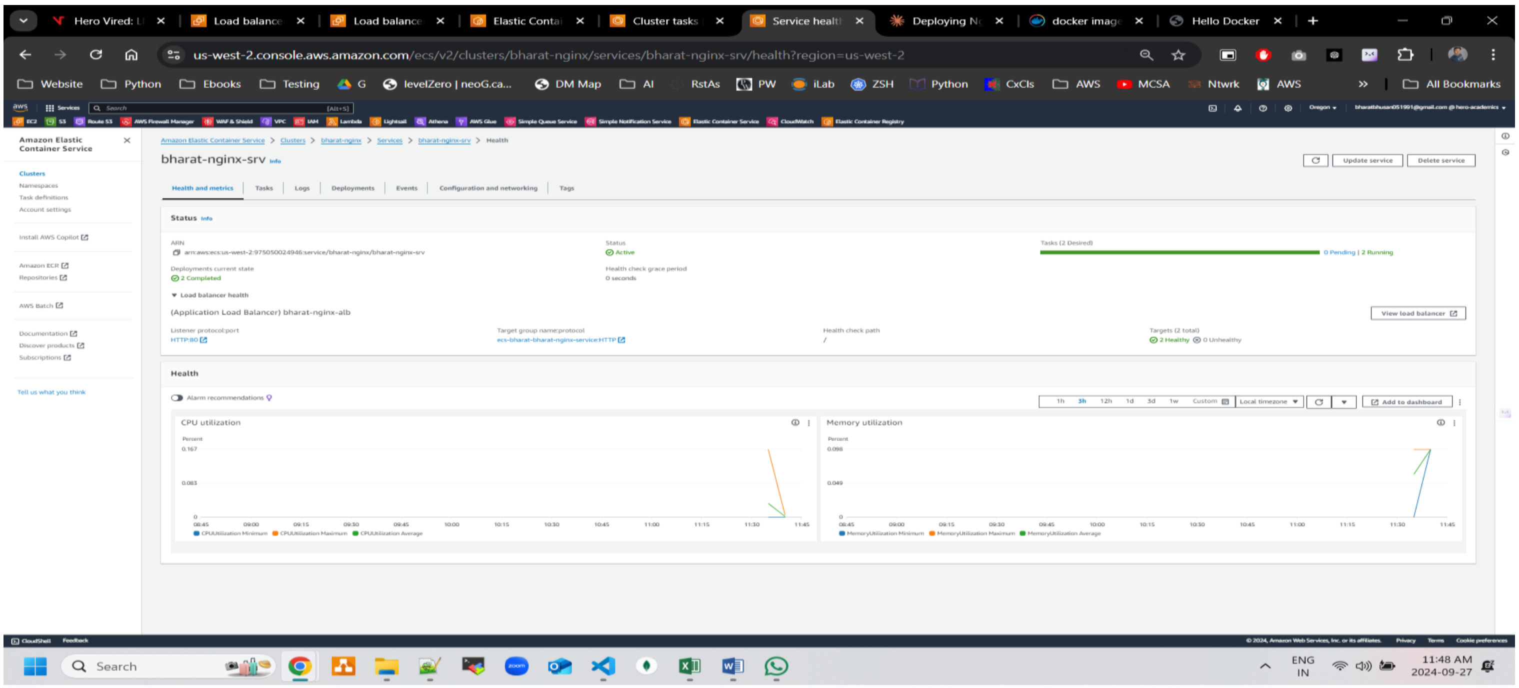
Task memory
3072 MiB (3 GB)

Task memory maximum allocation for container memory reservation

Containers Info

Container name	Image	Private registry	Essential	CPU	Memory hard/soft limit	GPU
bharat-nginx-cntr	975050024946.dkr.ecr.us-we...	-	Yes	0	-/-	-

➢ Create service (click on the image to open pdf of service)


us-west-2.console.aws.amazon.com/ecs/v2/clusters/bharat-nginx/services/bharat-nginx-srv/health?region=us-west-2

Amazon Elastic Container Service > Clusters > bharat-nginx > Services > bharat-nginx-srv > Health

bharat-nginx-srv Info

Status Info

ARN
arn:aws:ecs:us-west-2:975050024946:service/bharat-nginx/bharat-nginx-srv

Status
Active

Tasks (2 Desired)
0 Pending | 2 Running

Deployments current state
2 Completed

Health check grace period
0 seconds

Load balancer health

(Application Load Balancer) bharat-nginx-alb

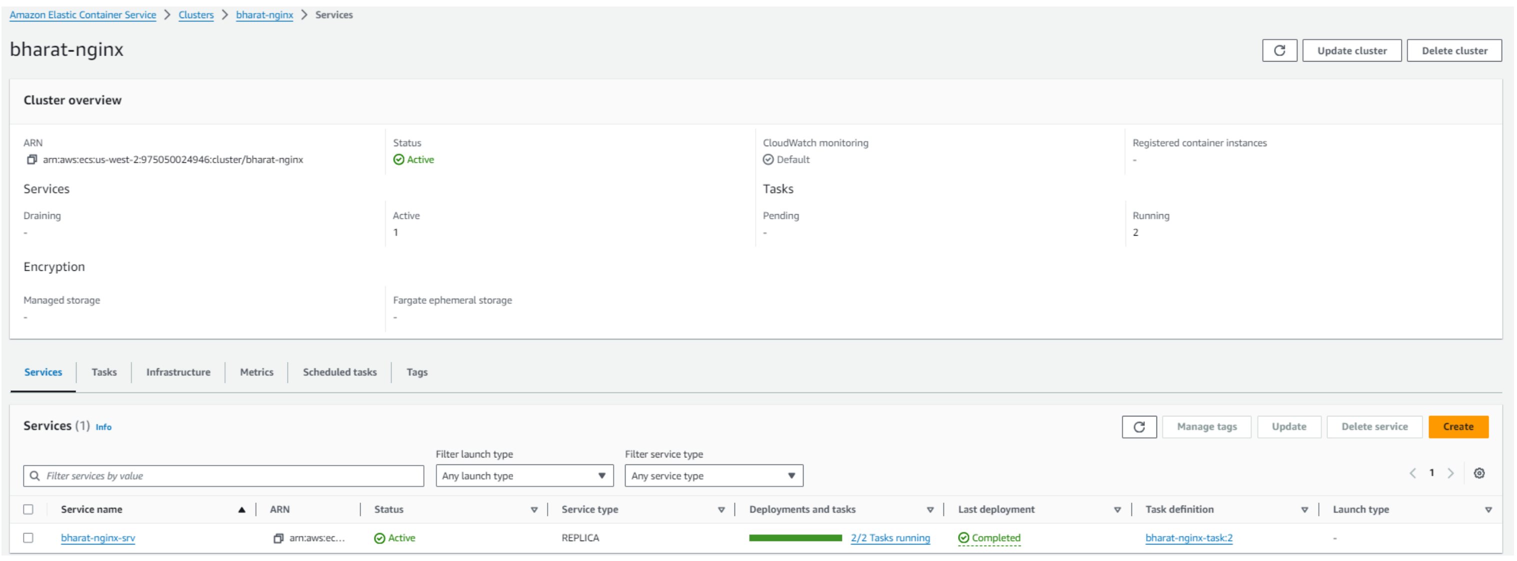
Listener protocol:port
HTTP:80

Target group name:protocol
ecs-bharat-bharat-nginx-service:HTTP

Health check path
/

Targets (2 total)
2 Healthy 0 Unhealthy

○ Custer bharat-nginx with Service Snap


Amazon Elastic Container Service > Clusters > bharat-nginx > Services

bharat-nginx

Cluster overview

ARN
arn:aws:ecs:us-west-2:975050024946:cluster/bharat-nginx

Status
Active

CloudWatch monitoring
Default

Registered container instances
-

Services

Draining
-

Active
1

Tasks

Pending
-

Running
2

Encryption

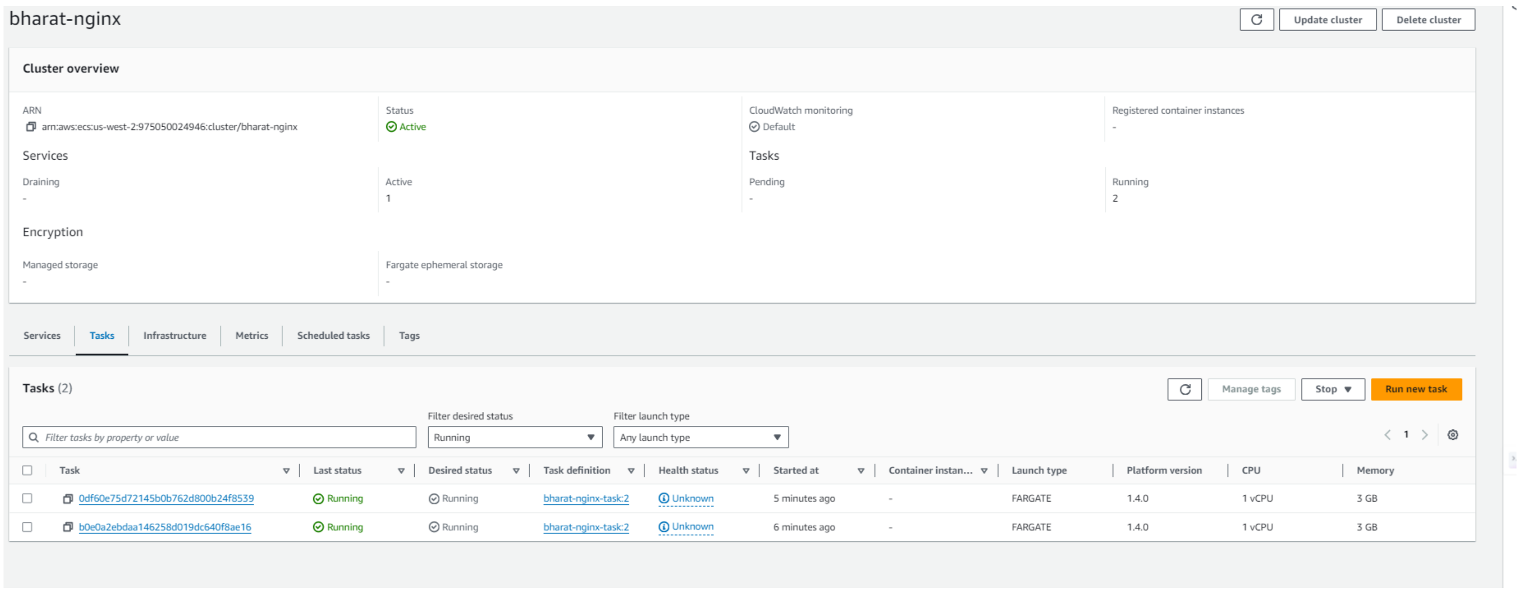
Managed storage
-

Fargate ephemeral storage
-

Services (1) Info

Service name	ARN	Status	Service type	Deployments and tasks	Last deployment	Task definition	Launch type
bharat-nginx-srv	arn:aws:ec...	Active	REPLICA	2/2 Tasks running	Completed	bharat-nginx-task:2	-

- Custer bharat-nginx with Task Snap



bharat-nginx

Cluster overview

ARN	Status	CloudWatch monitoring	Registered container instances
arn:aws:ecs:us-west-2:975050024946:cluster/bharat-nginx	Active	Default	-

Services

Draining	Active
-	1

Tasks

Pending	Running
-	2

Encryption

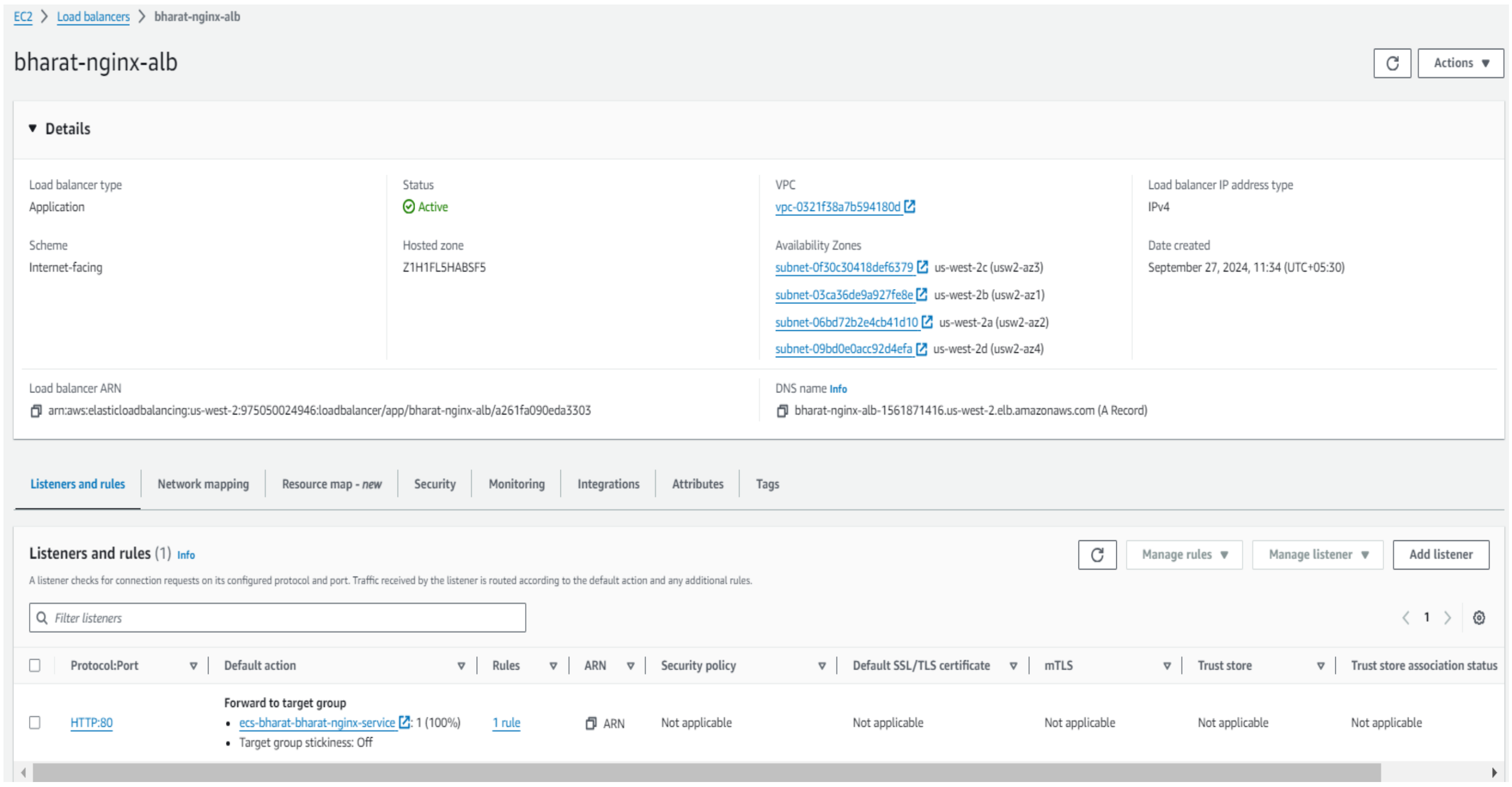
Managed storage	Fargate ephemeral storage
-	-

Tasks (2)

Filter desired status: Running
Filter launch type: Any launch type

Task	Last status	Desired status	Task definition	Health status	Started at	Container instan...	Launch type	Platform version	CPU	Memory
0df60e75d72145b0b762d800b24f8539	Running	Running	bharat-nginx-task:2	Unknown	5 minutes ago	-	FARGATE	1.4.0	1 vCPU	3 GB
b0e0a2ebdaa146258d019dc640f8ae16	Running	Running	bharat-nginx-task:2	Unknown	6 minutes ago	-	FARGATE	1.4.0	1 vCPU	3 GB

- LB created during service



EC2 > Load balancers > bharat-nginx-alb

bharat-nginx-alb

Details

Load balancer type	Status	VPC	Load balancer IP address type
Application	Active	vpc-0321f38a7b594180d	IPv4

Scheme	Hosted zone	Availability Zones	Date created
Internet-facing	Z1H1FL5HABSF5	subnet-0f30c30418def6379 us-west-2c (usw2-az3) subnet-03ca36de9a927fe8e us-west-2b (usw2-az1) subnet-06bd72b2e4cb41d10 us-west-2a (usw2-az2) subnet-09bd0e0acc92d4efa us-west-2d (usw2-az4)	September 27, 2024, 11:34 (UTC+05:30)

Load balancer ARN
arn:aws:elasticloadbalancing:us-west-2:975050024946:loadbalancer/app/bharat-nginx-alb/a261fa090eda3303

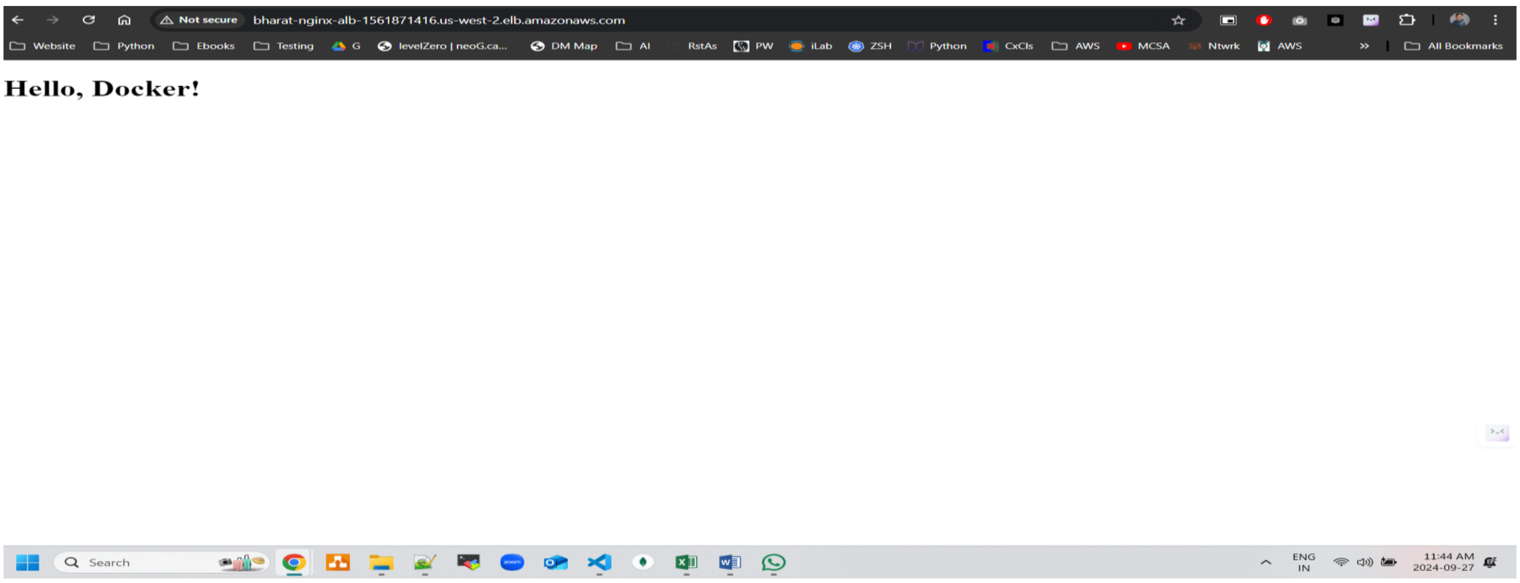
DNS name
bharat-nginx-alb-1561871416.us-west-2.elb.amazonaws.com (A Record)

Listeners and rules (1)

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS	Trust store	Trust store association status
HTTP:80	Forward to target group • ecs-bharat-bharat-nginx-service: 1 (100%) • Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable

- After accessing ALB DNS



Not secure bharat-nginx-alb-1561871416.us-west-2.elb.amazonaws.com

Hello, Docker!