

1. Create a tomcat container on 8080 and deploy sample application in tomcat.

- docker pull tomcat:7.0-jdk8-corretto

```
[root@ip-172-31-88-66 ~]# docker pull tomcat:7.0-jdk8-corretto

7.0-jdk8-corretto: Pulling from library/tomcat
85ee02fe1a09: Pull complete
f840dce53cea: Pull complete
80adf2308340: Pull complete
32a5b6fa71f3: Pull complete
16a32eb23061: Pull complete
Digest: sha256:8c8515cbfb8619f2ef28e0a96b8c025163b1cf3779c3f5db6131f20d8bf6682d
Status: Downloaded newer image for tomcat:7.0-jdk8-corretto
docker.io/library/tomcat:7.0-jdk8-corretto
[root@ip-172-31-88-66 ~]# |
```

- docker container run -itd -p 8000:8000 --name tomcat_test tomcat

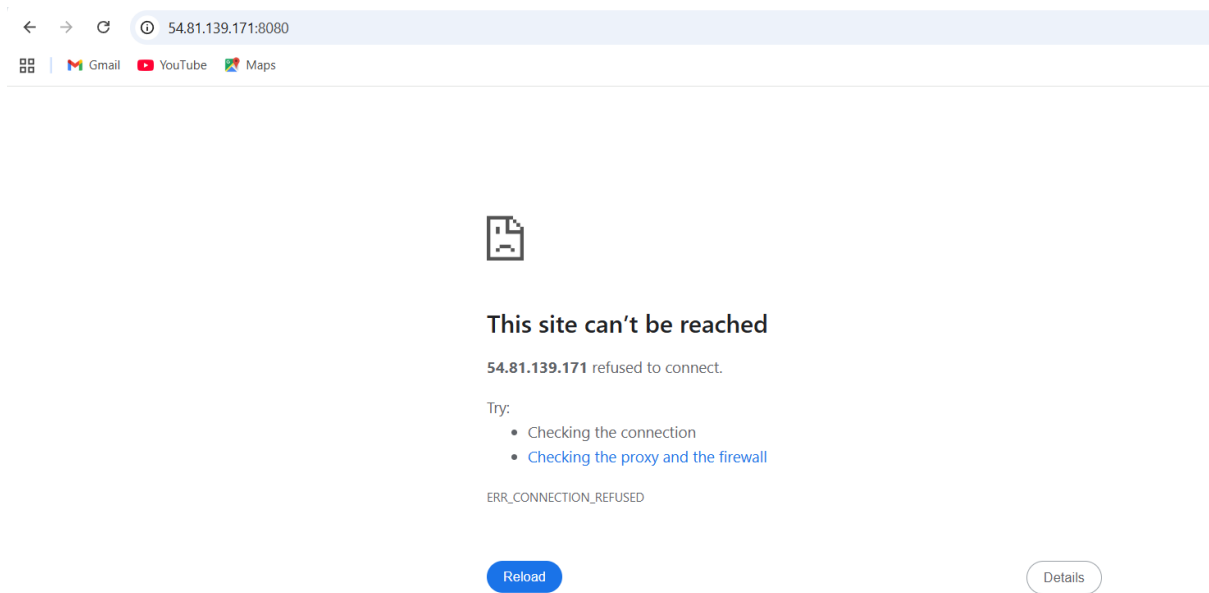
A container has been created for tomcat

```
[root@ip-172-31-88-66 ~]# docker container run -itd -p 8000:8000 --name tomcat_test tomcat
1f07335098f220813a5d7660fd42c337c58ac4fdb7c5c7c93c33cf833305fc2c
[root@ip-172-31-88-66 ~]# |
```

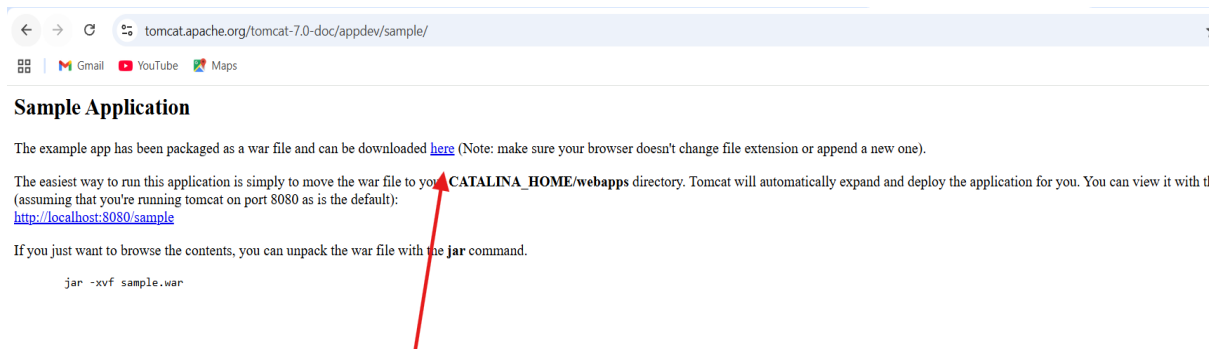
- docker container ps

```
[root@ip-172-31-88-66 ~]# docker container ps
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS              PORTS
1f07335098f2   tomcat    "catalina.sh run"        About a minute ago    Up About a minute   0.0.0.0:8000->8000/tcp, :::8000->8000/tcp
p, 8080/tcp    tomcat_test
[root@ip-172-31-88-66 ~]# |
```

Copy public ip and port number 8080 if we find error.



Login to your container and deploy a sample.war file



- download sample war in root location

```
[root@ip-172-31-75-180 ~]# wget https://tomcat.apache.org/tomcat-7.0-doc/appdev/sample/
--2025-11-24 14:18:41-- https://tomcat.apache.org/tomcat-7.0-doc/appdev/sample/sample.
Resolving tomcat.apache.org (tomcat.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to tomcat.apache.org (tomcat.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4606 (4.5K)
Saving to: 'sample.war'

sample.war                               100%[=====]
2025-11-24 14:18:41 (60.9 MB/s) - 'sample.war' saved [4606/4606]
```

Go into your container by using

- `docker exec -it container_id /bin/bash`

```
[root@ip-172-31-75-180 ~]# docker exec -it 89b0eff2eb90 /bin/bash
root@89b0eff2eb90:/usr/local/tomcat# ls
bin          CONTRIBUTING.md  LICENSE        NOTICE        RUNNING.txt    webapps
BUILDING.txt  filtered-KEYS    logs           README.md      temp           webapps.dist
conf         lib             native-jni-lib  RELEASE-NOTES  upstream-KEYS  work
root@89b0eff2eb90:/usr/local/tomcat# cd webapps
root@89b0eff2eb90:/usr/local/tomcat/webapps# ll
total 0
drwxr-xr-x. 2 root root  6 Nov 14 00:41 ./
drwxr-xr-x. 1 root root 30 Nov 14 00:41 ../
```

Copy the sample war and move to your container.

- docker cp sample.war
container_name:/usr/local/tomcat/webapps/

```
[root@ip-172-31-75-180 ~]# docker cp sample.war tomcat_test:/usr/local/tomcat/webapps/
Successfully copied 6.14kB to tomcat_test:/usr/local/tomcat/webapps/
[root@ip-172-31-75-180 ~]# docker exec -it 89b0eff2eb90 /bin/bash
root@89b0eff2eb90:/usr/local/tomcat# cd /webapps
bash: cd: /webapps: No such file or directory
root@89b0eff2eb90:/usr/local/tomcat# ll
total 220
drwxr-xr-x. 1 root root   57 Nov 14 00:41 ./
drwxr-xr-x. 1 root root   20 Nov 14 00:40 ../
drwxr-xr-x. 2 root root 16384 Nov 14 00:41 bin/
-rw-r--r--. 1 root root 24262 Nov  5 19:03 BUILDING.txt
```

```
drwxr-xr-x. 7 root root   81 Nov  5 19:03 webapps.dist/
drwxrwxrwt. 1 root root   22 Nov 24 14:21 work/
root@89b0eff2eb90:/usr/local/tomcat# cd webapps
root@89b0eff2eb90:/usr/local/tomcat/webapps# ls
sample  sample.war
root@89b0eff2eb90:/usr/local/tomcat/webapps#
```

Go to browser search by port number 8080 and /sample

Sample "Hello, World" Application

This is the home page for a sample application used to illustrate the source directory organization of a web application utilizing the principles outli

To prove that they work, you can execute either of the following links:

- To a [JSP page](#).
- To a [servlet](#).

2. Create volume and deploy tomcat container on port 8081.

* docker volume create deployment

```
[root@ip-172-31-75-180 ~]# docker volume create deployment
deployment
[root@ip-172-31-75-180 ~]# docker volume ls
DRIVER      VOLUME NAME
local       deployment
[root@ip-172-31-75-180 ~]#
```

Go to volumes

- cd /var/lib/docker/volumes/deployment/_data
- wget samplewar

```
[root@ip-172-31-75-180 ~]# cd /var/lib/docker/volumes
[root@ip-172-31-75-180 volumes]# ls
backingFsBlockDev deployment metadata.db
[root@ip-172-31-75-180 volumes]# cd deployment/
[root@ip-172-31-75-180 deployment]# ls
_data
[root@ip-172-31-75-180 deployment]# cd _data/
[root@ip-172-31-75-180 _data]# wget https://tomcat.apache.org/tomcat-7.0-doc
--2025-11-24 14:38:25-- https://tomcat.apache.org/tomcat-7.0-doc/appdev/sam
Resolving tomcat.apache.org (tomcat.apache.org)... 151.101.2.132, 2a04:4e42:
Connecting to tomcat.apache.org (tomcat.apache.org)|151.101.2.132|:443... co
HTTP request sent, awaiting response... 200 OK
Length: 4606 (4.5K)
Saving to: 'sample.war'

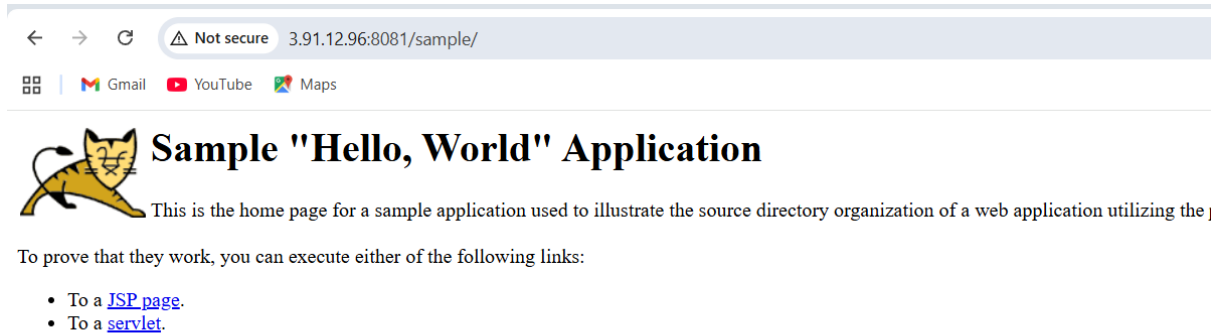
sample.war                               100%[=====
2025-11-24 14:38:25 (62.4 MB/s) - 'sample.war' saved [4606/4606]

[root@ip-172-31-75-180 _data]#
```

To map your container to volume.

- docker run -itd -p 8081:8080 -v
/var/lib/docker/volumes/deployment/_data:/usr/local/t
omcat/webapps tomcat

```
[root@ip-172-31-75-180 ~]# docker run -itd -p 8081:8080 -v /var/lib/docker/volumes/deployment/_data:/usr/local/tomcat/webapps
tomcat
66e01efd28a18c72dae1c24c6d307612e978cef552ea0fec822b81a365cbfa95
[root@ip-172-31-75-180 ~]#
```



3. Limit the nginx container to 500 MB.

* docker run -d --name nginx -p 80:80 --memory="500m" nginx:latest

```
[root@ip-172-31-75-180 ~]# docker run -d --name nginx -p 80:80 --memory="500m" nginx:latest
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
0e4bc2bd6656: Pull complete
b5feb73171bf: Pull complete
108ab8292820: Pull complete
53d743880af4: Pull complete
77fa2eb06317: Pull complete
192e2451f875: Pull complete
de57a609c9d5: Pull complete
Digest: sha256:553f64aecdc31b5bf944521731cd70e35da4faed96b2b7548a3d8e2598c52a42
Status: Downloaded newer image for nginx:latest
d2358797bd3c2d7c4e683310359094f86c8d844b5d41b4f5bc1e2d7d76cfb296
[root@ip-172-31-75-180 ~]#
```

- docker stats

root@ip-172-31-75-180~

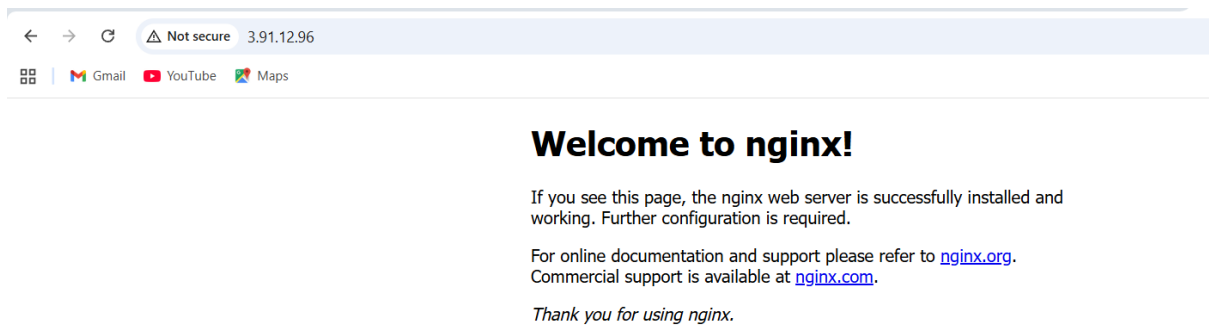
CONTAINER ID	NAME	CPU %	MEM USAGE / LIMIT	MEM %	NET I/O	BLOCK I/O	PIDS
d2358797bd3c	nginx	0.00%	13.32MiB / 500MiB	2.66%	726B / 0B	10.8MB / 12.3kB	3
66e01efd28a1	trusting_merkle	0.04%	74.7MiB / 904.8MiB	8.26%	21.2kB / 40.3kB	483kB / 815kB	29
89b0eff2eb90	tomcat_test	0.10%	80.18MiB / 904.8MiB	8.86%	28.5kB / 51.2kB	13.5MB / 1.07MB	29

- docker inspect nginx | grep -i memory

```
[root@ip-172-31-75-180 ~]# docker inspect nginx | grep -i memory
      "Memory": 524288000,
      "MemoryReservation": 0,
      "MemorySwap": 1048576000,
      "MemorySwappiness": null,
[root@ip-172-31-75-180 ~]# |
```

```
      "MemorySwappiness": null,
[root@ip-172-31-75-180 ~]# docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED    STATUS    PORTS
NAMES
d2358797bd3c   nginx:latest  "/docker-entrypoint..." 3 minutes ago    Up 3 minutes    0.0.0.0:80->80/tcp, :::80->80/tcp
nginx
66e01efd28a1   tomcat      "catalina.sh run"        55 minutes ago    Up 55 minutes    0.0.0.0:8081->8080/tcp, :::8081->8080/tcp
tcp_trusting_merkle
89b0eff2eb90   tomcat      "catalina.sh run"        2 hours ago      Up 2 hours      0.0.0.0:8080->8080/tcp, :::8080->8080/tcp
tcp_tomcat_test
[root@ip-172-31-75-180 ~]# |
```

Search in browser with public_ip and portnumber 80



4. Create a sample docker file using below instructions.

a. Base module as amazonlinux:latest

- b. Maintainer you name**
- c. Install nginx**
- d. COPY one index.html file to image**
- e. Expose on port 80**
- f. Command to start the nginx container**

select root location

- cd
- mkdir nginx-sample
- cd nginx-sample

```
root@ip-172-31-75-180:~/nginx-sample
[root@ip-172-31-75-180 ~]# cd
[root@ip-172-31-75-180 ~]# mkdir nginx-sample
[root@ip-172-31-75-180 ~]# ls
nginx-sample  sample.war
[root@ip-172-31-75-180 ~]# cd nginx-sample/
[root@ip-172-31-75-180 nginx-sample]# ls
[root@ip-172-31-75-180 nginx-sample]#
```

In the nginx-sample directory create a file vi index.html

```
[root@ip-172-31-75-180 nginx-sample]# vi index.html
[root@ip-172-31-75-180 nginx-sample]#
```

```
<!doctype html>
<html>
<head>
<title>My Nginx Sample</title>
</head>
<body>
<h1>Hello from Nginx on Amazon Linux!</h1>
<p>This page is served inside a Docker container.</p>
</body>
</html>
```

```
~
~
~
~
```

In the same location create another file named as vi dockerfile.

```
[root@ip-172-31-75-180 nginx-sample]# vi dockerfile
```

```
# a. Base image
FROM amazonlinux:latest

# b. Maintainer
LABEL maintainer="Shaik Mohammad Mujaheed"

# c. Install nginx
RUN yum update -y && \
    yum install -y nginx && \
    yum clean all

# d. Copy index.html into nginx default directory
COPY index.html /usr/share/nginx/html/index.html

# e. Expose port 80
EXPOSE 80

# f. Command to start nginx in the foreground
CMD ["nginx", "-g", "daemon off;"]
```

```
~
~
~
~
~
~
~
```

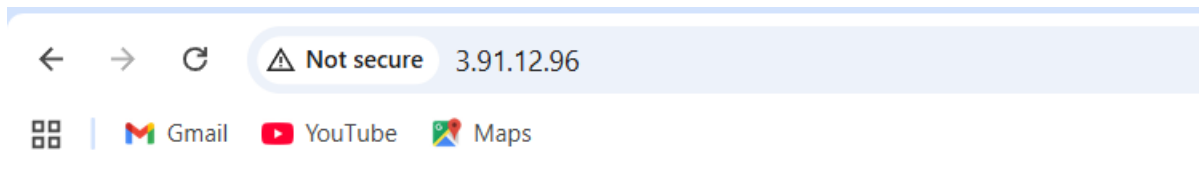

- `docker build -t my-nginx .`

```
[root@ip-172-31-75-180 nginx-sample]# docker build -t my-nginx .
[+] Building 27.9s (8/8) FINISHED                                docker:default
=> [internal] load build definition from dockerfile              0.0s
=> => transferring dockerfile: 504B                               0.0s
=> [internal] load metadata for docker.io/library/amazonlinux:latest 0.1s
=> [internal] load .dockerignore                                  0.0s
=> => transferring context: 2B                                       0.0s
=> [1/3] FROM docker.io/library/amazonlinux:latest@sha256:99c2f938acd0d2a154a005b8dd30c90718727736c60cc93e718f68020948 3.8s
=> => resolve docker.io/library/amazonlinux:latest@sha256:99c2f938acd0d2a154a005b8dd30c90718727736c60cc93e718f68020948 0.0s
=> => sha256:1c7de4eb5ced9ea3f72366a34ec955a53e9b0f4ac53d332a155de21eb808d732 53.97MB / 53.97MB 0.6s
=> => sha256:99c2f938acd0d2a154a005b8dd30c90718727736c60cc93e718f68020948a41d 2.38kB / 2.38kB 0.0s
=> => sha256:1406cbeb92543cf0d6428ee6ac4bef62a05f529ecdb27320a35872e26203178d 1.02kB / 1.02kB 0.0s
=> => sha256:116cbad2beaf749c7560b2b53ad7e0428b2890fe88ad0977880e8cc77ca69803 586B / 586B 0.0s
=> => extracting sha256:1c7de4eb5ced9ea3f72366a34ec955a53e9b0f4ac53d332a155de21eb808d732 3.1s
=> [internal] load build context                                  0.0s
=> => transferring context: 288B                                       0.0s
=> [2/3] RUN yum update -y && yum install -y nginx && yum clean all 23.0s
=> [3/3] COPY index.html /usr/share/nginx/html/index.html       0.2s
=> exporting to image                                             0.6s
=> => exporting layers                                                 0.5s
=> => writing image sha256:84fb0e19d80edbc02eb6e6ddf28037f020972bcc43f74d47328f2aef7ab282a7 0.0s
=> => naming to docker.io/library/my-nginx                          0.0s
[root@ip-172-31-75-180 nginx-sample]#
```

- `docker run -d -p 80:80 my-nginx`

```
[root@ip-172-31-75-180 nginx-sample]# docker run -d -p 80:80 my-nginx
001c897da01b068fedc6f36796f158137bb02f84591a5d54dbeeb01b37f1511b
[root@ip-172-31-75-180 nginx-sample]#
```

Search on browser with the port number 80.



Hello from Nginx on Amazon Linux!


This page is served inside a Docker container.

5 . Push image to Docker hub

Signup to dockerhub.

← → ↻ app.docker.com/signup?_gl=1*69k7if*_gcl_au*MjA1MDI5MDExNS4xNzU2NTMxODE0*_ga*MTU5MDU1MDgzNy4xNzU2NTMxNjEy*_ga_XJWPQMjYHQ*czE3NjQwMD

Gmail YouTube Maps



Create your account

Work Personal

Work email



Username

Password

☐ Send me occasional product updates and announcements.

Sign up

OR


 Continue with Google  Continue with GitHub

[Already have an account? Sign in](#)

This site is protected by hCaptcha and the hCaptcha [Privacy Policy](#) and [Terms of Service](#) apply.

← → ↻ app.docker.com/accounts/mujaheed00

Gmail YouTube Maps

 **mujaheed00**
Docker Personal


- Home
- Hub
- Build Cloud
- Hardened Images
- Scout
- Testcontainers Cloud
- Docker Desktop
- Settings
- Billing

Welcome to Docker Home, mujaheed00

Access and manage your Docker Desktop, Build Cloud, Scout, and Hub products, and get access to resources for learning, support, and account settings, including billing management.

[Get started with Docker guidance](#) [Learn about Docker concepts](#)


Docker products

**docker.desktop**

Innovate with **Docker Desktop**

Your command center for innovative local and cloud native container development.


[Go to download](#) [Launch Docker Desktop](#)

**buildcloud**

Build with **Docker Build Cloud**

Accelerate image build times with access to cloud-based builders and shared cache.


[Go to Build Cloud](#)


**scout**


Secure with **Docker Scout**

Address security issues before they hit production through actionable insights across the software supply chain.

[Go to Scout](#)

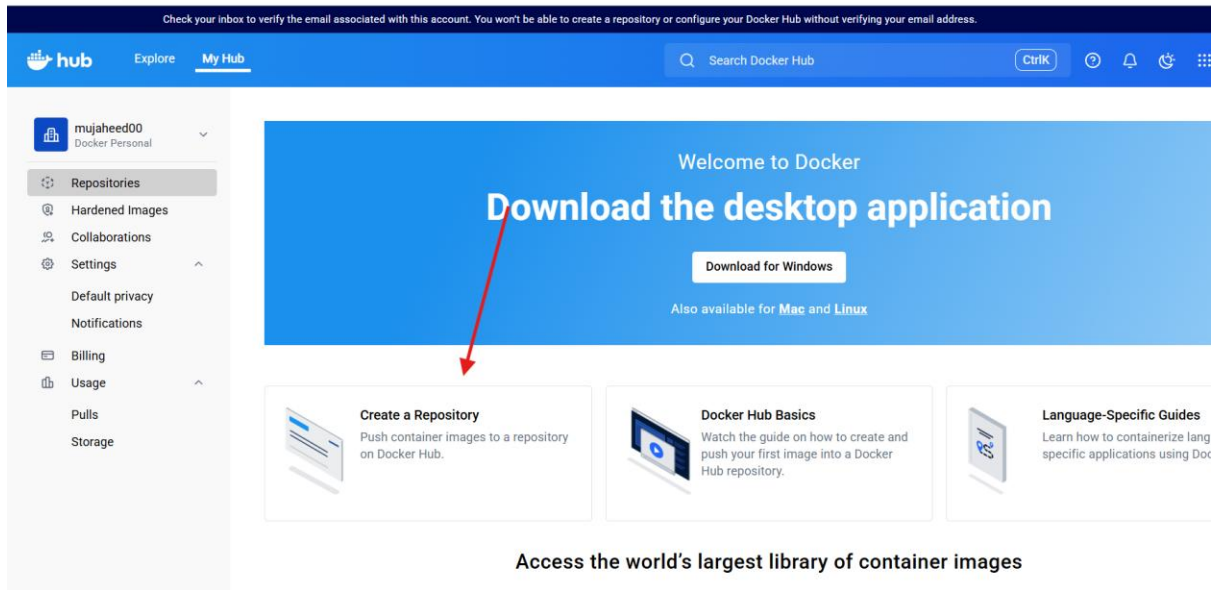
**docker.hub**

**Testcontainers Cloud**

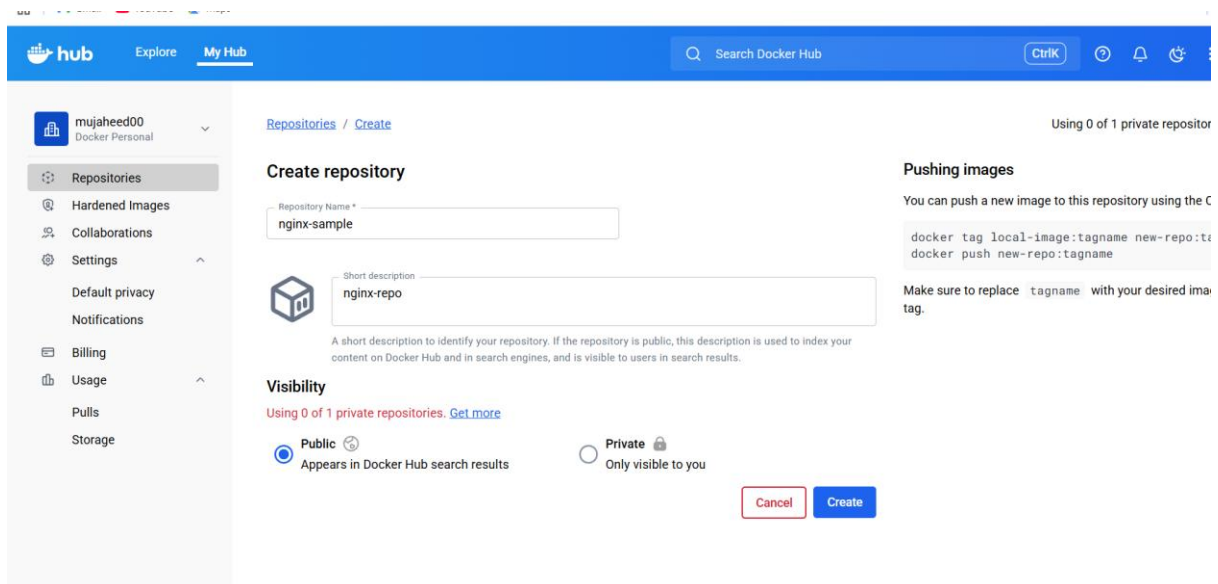


[Give feedback](#)

Go to hub click on repository,create repository.



Give name as nginx-sample click on create.



- docker login

give your username and password.

```
root@ip-172-31-75-180:~  
[root@ip-172-31-75-180 ~]# docker login  
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you don't have  
to https://hub.docker.com/ to create one.  
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT grants be  
quired for organizations using SSO. Learn more at https://docs.docker.com/go/access-tokens/  
  
Username: mujaheed00  
Password:  
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-172-31-75-180 ~]# |
```

- `docker tag my-nginx mujaheed00/nginx-sample:latest`

```
[root@ip-172-31-75-180 ~]# docker tag my-nginx mujaheed00/nginx-sample:latest  
[root@ip-172-31-75-180 ~]# |
```

- `docker push mujaheed00/nginx-sample:latest`

```
[root@ip-172-31-75-180 ~]# docker tag my-nginx mujaheed00/nginx-sample:latest  
[root@ip-172-31-75-180 ~]# docker push mujaheed00/nginx-sample:latest  
The push refers to repository [docker.io/mujaheed00/nginx-sample]  
e3dda3eed3d: Pushed  
f0224e513680: Pushed  
0e7c6314d998: Mounted from library/amazonlinux  
latest: digest: sha256:f6377b28d28e64b83d0a5af4066dd829b085c6eecb60e7a2d53192ca3b7082d1 size: 948  
[root@ip-172-31-75-180 ~]# |
```

The screenshot shows a Docker Hub repository page for 'mujaheed00/nginx-sample'. The page includes a sidebar with navigation options like 'Repositories', 'Hardened Images', 'Collaborations', 'Settings', 'Default privacy', 'Notifications', 'Billing', 'Usage', 'Pulls', and 'Storage'. The main content area shows the repository name, last pushed time (1 minute ago), repository size (73.6 MB), and star/fork counts (0 stars, 0 forks, 0 downloads). Below this, there are tabs for 'General', 'Tags', 'Image Management', 'Collaborators', 'Webhooks', and 'Settings'. The 'Tags' tab is active, showing a table with columns 'Tag', 'OS', 'Type', 'Pulled', and 'Pushed'. The table contains one entry: 'latest' (Image type, pulled less than 1 day ago, pushed 1 minute ago). To the right of the table, there is a 'DOCKER SCOUT INACTIVE' status with an 'Activate' link. Below the table, there is a 'Repository overview' section with an 'Add overview' button. On the far right, there is a 'buildcloud' advertisement with the text 'Build with Docker Build Cloud' and a 'Go to Docker Build Cloud' link.

6. push image to AWS ECR

Go to aws ECR create a repository.

The screenshot shows the Amazon Elastic Container Registry (ECR) console. The header includes the AWS logo, a search bar, and navigation icons. The main content area has a dark background with the text 'Containers' and 'Amazon Elastic Container Registry'. Below this, it says 'Share and deploy container software, publicly or privately'. A 'Create a repository' button is visible. The 'Pricing (US)' section explains that users pay only for the amount of data stored in their public or private repositories and data transferred to the Internet. It also mentions 'ECR pricing' and 'Enable private connection to ECR private repositories using VPC interface endpoints and reduce cost.' The footer includes links for 'CloudShell', 'Feedback', 'Console Mobile App', and copyright information for Amazon Web Services, Inc. or its affiliates.

Give the repository name.

aws Search [Alt+S]

Amazon ECR > Private registry > Repositories > Create private repository

Create private repository

General settings

Repository name
Enter a concise name. Repositories support namespaces, which you can use to group similar repositories.

235351028455.dkr.ecr.us-east-1.amazonaws.com/

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 settings [Info](#)

Image tag mutability
Choose the tag mutability setting.

☒ **Mutable**
Image tags can be overwritten.

☐ **Immutable**
Image tags can't be overwritten.

Mutable tag exclusions
Tags that match these filters will be immutable (can't be overwritten). Using wildcards (*) will match zero or more image tag characters.

[Add filter](#)

Filters must only contain letters, numbers, and special characters (._-*). Each filter is limited to 128 characters, 2 wildcards (*), and you can add up to 5 filters in the exclusion list.

There are no images.

us-east-1.console.aws.amazon.com/ecr/repositories/private/235351028455/nginx-sample/_/details?region=us-east-1

Amazon ECR > Private registry > Repositories > Images

nginx-sample

Summary **Images**

Images [Refresh](#) [Delete](#) [Copy URI](#) [Details](#) [Scan](#) [View push commands](#)

<input type="checkbox"/>	Image tags	Type	Created at	Image size	Image digest	Last pulled at
No active images						

- aws configure

```
[root@ip-172-31-75-180 ~]# aws configure
AWS Access Key ID [None]: AKIATNTADWLT6MZQNDMN
AWS Secret Access Key [None]: ie52YFxmecWog+cOgvq3XpqKogt7Vcr+5T8DfIS
Default region name [None]: us-east-1
Default output format [None]: json
[root@ip-172-31-75-180 ~]#
```

- `aws ecr get-login-password --region us-east-1 \`
`| docker login --username AWS --password-stdin`
`235351028455.dkr.ecr.us-east-1.amazonaws.com`

```
[root@ip-172-31-75-180 ~]# aws ecr get-login-password --region us-east-1 \
| docker login --username AWS --password-stdin 235351028455.dkr.ecr.us-east-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-172-31-75-180 ~]# |
```

- `docker tag my-nginx:latest 235351028455.dkr.ecr.us-east-1.amazonaws.com/nginx-sample:latest`
- `docker push 235351028455.dkr.ecr.us-east-1.amazonaws.com/nginx-sample:latest`

```
[root@ip-172-31-75-180 ~]# docker tag my-nginx:latest 235351028455.dkr.ecr.us-east-1.amazonaws.com/nginx-sample:latest
[root@ip-172-31-75-180 ~]# docker push 235351028455.dkr.ecr.us-east-1.amazonaws.com/nginx-sample:latest
The push refers to repository [235351028455.dkr.ecr.us-east-1.amazonaws.com/nginx-sample]
e3dda3eead3d: Pushed
f0224e513680: Pushed
0e7c6314d998: Pushed
latest: digest: sha256:f6377b28d28e64b83d0a5af4066dd829b085c6eeecb60e7a2d53192ca3b7082d1 size: 948
[root@ip-172-31-75-180 ~]# |
```

The image was pushed to ECR

The screenshot shows the Amazon Elastic Container Registry (ECR) console. The left sidebar displays the navigation menu with options like 'Private registry', 'Public registry', 'Repositories', 'Permissions', 'Lifecycle Policy', 'Repository tags', 'Features & Settings', 'ECR public gallery', 'Amazon ECS', and 'Amazon EKS'. The main content area is titled 'nginx-sample' and shows the 'Images' tab. A table lists the images in the repository:

Image tags	Type	Created at	Image size	Image digest	Last pulled at
latest	Image	November 24, 2025, 22:39:10 (UTC+05.5)	77.17	sha256:f6377b28...	-

Buttons for 'Delete', 'Copy URI', 'Details', 'Scan', and 'View push commands' are visible above the table.

