CI/CD pipeline: Dockers and Containers

Overview

In this document, we will detail the use of Docker in a CI/CD pipeline and outline steps to set up the Docker service on an AWS EC2 instance and create containers to run the code for a static website.

Introduction

Docker is a platform that uses containers to package and run applications, ensuring consistency across different environments. Containers are lightweight, isolated environments that include everything an application needs to run, such as code, runtime, system tools, and libraries. This allows developers to build, test, and deploy applications more efficiently and reliably, regardless of the underlying infrastructure.
Account in Docker Hub (Optional)

Docker plays a **crucial role in CI/CD pipelines** by standardizing environments, enabling consistent testing and deployment, and simplifying dependency management. Here's a breakdown of **how Docker works in a CI/CD pipeline**:

CI/CD Pipeline Overview

CI/CD stands for:

- **CI (Continuous Integration):** Automatically build and test code when developers push changes.
- **CD (Continuous Delivery/Deployment):** Automatically deliver or deploy the code to staging or production.

Role of Docker in CI/CD

Docker helps by:

- Packaging applications with all dependencies into **containers**.
- Running consistent builds and tests across environments.
- Reducing "works on my machine" issues.
- Simplifying scaling and deployment.

Typical CI/CD Pipeline with Docker

1. Code is Pushed

A developer pushes code to a Git repo (e.g., GitHub, GitLab, Bitbucket).

2. CI Pipeline is Triggered

CI server (like Jenkins, GitLab CI, GitHub Actions, CircleCI) detects the push and runs the pipeline.

3. Docker Build

The CI tool runs a step like:

bash

CopyEdit

```
docker build -t myapp:latest .
```

- This uses a Dockerfile to package your app and its dependencies(into an image).

4. Run Unit/Integration Tests

Tests are run inside Docker containers:

bash

CopyEdit

```
docker run myapp:latest npm test
```

Or with docker-compose:

bash

CopyEdit

```
docker-compose -f docker-compose.test.yml up --abort-on-container-exit
```

5. Security/Quality Checks (Optional)

Tools like Trivy (for image scanning) or SonarQube (for code quality) run checks on the Docker image.

6. Push Docker Image to Registry

Once tests pass, the image is pushed to a registry:

bash

CopyEdit

```
docker push myapp:latest
```

Examples:

- Docker Hub
- GitHub Container Registry
- Amazon ECR
- Google Artifact Registry

7. Deploy to Environment

Deployment tools use the Docker image to update environments:

- Kubernetes (kubectl apply)
- Docker Swarm
- ECS, EKS, or GKE
- Helm charts or Terraform can help automate infrastructure

Benefits of Using Docker in CI/CD

- Same environment across dev, test, and production
- Easy to reproduce builds
- Isolated, stateless, and disposable testing
- Works well with microservices

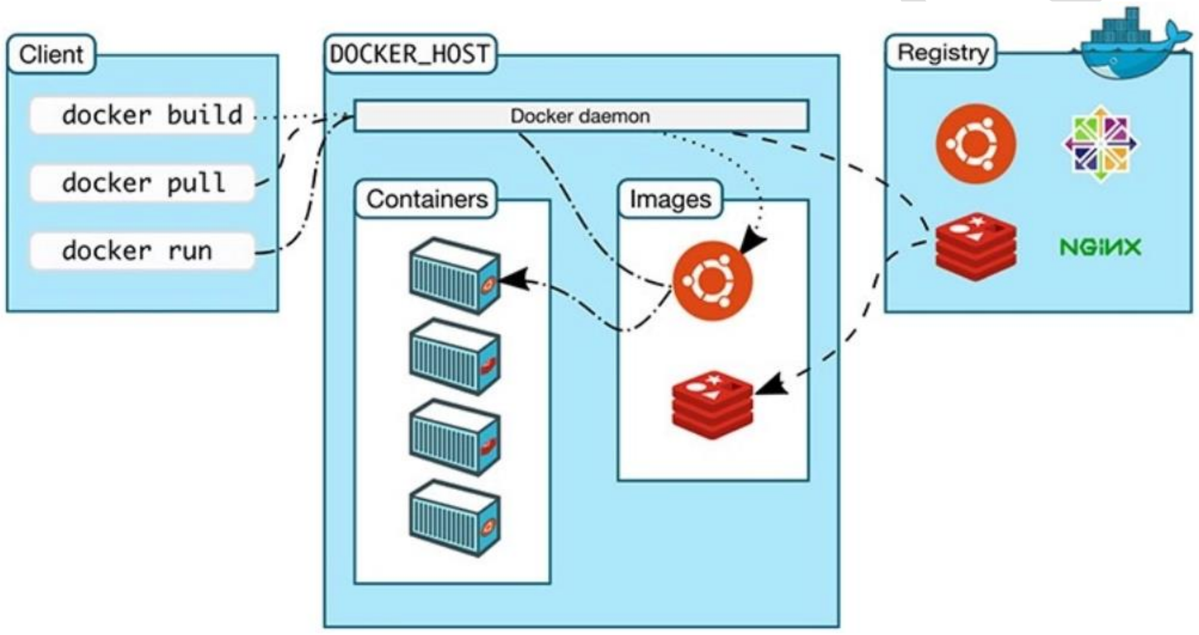
Without Docker (Traditional Pipelines)

- Environment drift (different setups across environments)
- Hard-to-replicate bugs
- Manual dependency setup

Docker Architecture

The Core components of Docker consist of

1. Docker daemon
2. Docker client
3. Docker Desktop
4. Docker registry
5. Docker images
6. Docker containers



The Docker daemon

The Docker daemon (dockerd) listens for Docker API requests and manages Docker objects such as images, containers, networks, and volumes. A daemon can also communicate with other daemons to manage Docker services.

The Docker client

The Docker client (docker) is the primary means by which many Docker users interact with Docker. When you use commands such as docker run, the client sends these commands to dockerd, which carries them out. The docker command uses the Docker API. The Docker client can communicate with more than one daemon.

Docker Desktop

Docker Desktop is an easy-to-install application for your Mac or Windows environment that enables you to build and share containerized applications and Microservices. Docker Desktop includes the Docker daemon (dockerd), the Docker client (docker), Docker Compose, Docker Content Trust, Kubernetes, and Credential Helper. For more information, see Docker Desktop.

Docker registries

A Docker registry stores Docker images. Docker Hub is a public registry that anyone can use, and Docker is configured to look for images on Docker Hub by default. You can even run your own private registry.
When you use the docker pull or docker run commands, the required images are pulled from your configured registry. When you use the docker push command, your image is pushed to your configured registry.

Docker objects

When you use Docker, you are creating and using images, containers, networks, volumes, plugins, and other objects. This section is a brief overview of some of those objects.

Images

An image is a read-only template with instructions for creating a Docker container. Often, an image is based on another image, with some additional customization. For example, you may build an image which is based on the ubuntu image, but installs the Apache web server and your application, as well as the configuration details needed to make your application run.

You might create your own images or you might only use those created by others and published in a registry. To build your own image, you create a Dockerfile with a simple syntax for defining the steps needed to create the image and run it. Each instruction in a Dockerfile creates a layer in the image. When you change the Dockerfile and rebuild the image, only those layers which have changed are rebuilt. This is part of what makes images so lightweight, small, and fast, when compared to other virtualization technologies.

Containers

A container is a runnable instance of an image. You can create, start, stop, move, or delete a container using the Docker API or CLI. You can connect a container to one or more networks, attach storage to it, or even create a new image based on its current state.

By default, a container is relatively well isolated from other containers and its host machine. You can control how isolated a container's network, storage, or other underlying subsystems are from other containers or from the host machine.

A container is defined by its image as well as any configuration options you provide to it when you create or start it. When a container is removed, any changes to its state that are not stored in persistent storage disappear.

Prerequisites for the project

An active and working AWS account

Basic knowledge of AWS, LINUX, Docker, and Containers

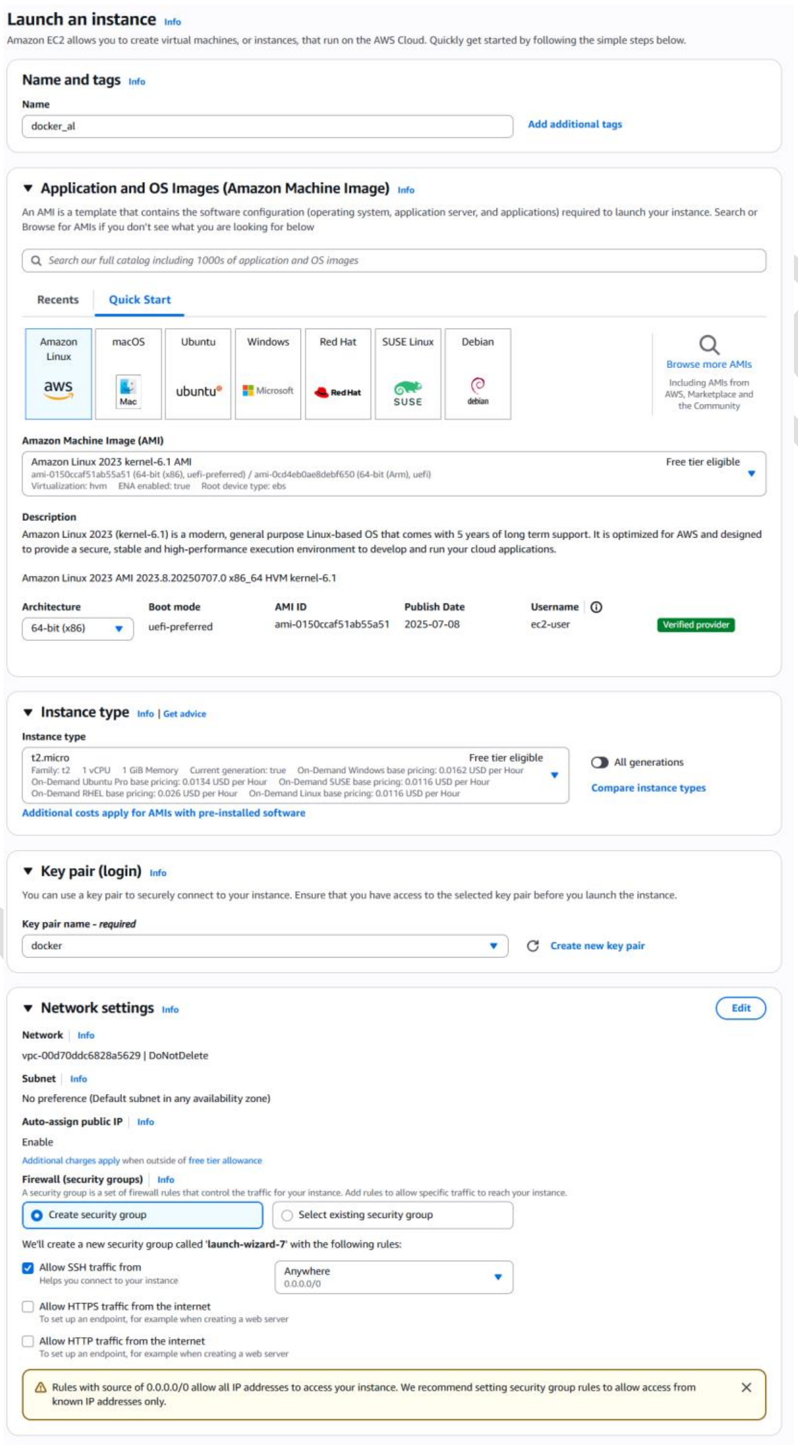
Static Website Code (HTML, CSS, JS, etc)

Filezilla or MobaXterm

Implementation Steps

EC2 instance and Docker setup

1. Create an AWS instance named Docker with “Docker” key pair, T3 micro



Launch an instance Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags Info

Name: docker_al

Add additional tags

▼ **Application and OS Images (Amazon Machine Image)** Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Recents | Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Linux | Debian

Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 kernel-6.1 AMI — Free tier eligible
ami-0150ccaf51ab55a51 (64-bit (x86), uefi-preferred) / ami-0cd4eb0ae8debf650 (64-bit (Arm), uefi)
Virtualization: hvm   ENA enabled: true   Root device type: ebs

Description

Amazon Linux 2023 (kernel-6.1) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.8.20250707.0 x86_64 HVM kernel-6.1

Architecture	Boot mode	AMI ID	Publish Date	Username	
64-bit (x86)	uefi-preferred	ami-0150ccaf51ab55a51	2025-07-08	ec2-user	Verified provider

▼ **Instance type** Info | Get advice

Instance type

t2.micro — Free tier eligible
Family: t2   1 vCPU   1 GiB Memory   Current generation: true   On-Demand Windows base pricing: 0.0162 USD per Hour
On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour   On-Demand SUSE base pricing: 0.0116 USD per Hour
On-Demand RHEL base pricing: 0.026 USD per Hour   On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

Compare instance types

Additional costs apply for AMIs with pre-installed software

▼ **Key pair (login)** Info

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*: docker

Create new key pair

▼ **Network settings** Info — Edit

Network Info
vpc-00d70ddc6828a5629 | DoNotDelete

Subnet Info
No preference (Default subnet in any availability zone)

Auto-assign public IP Info
Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

[● Create security group] ○ Select existing security group

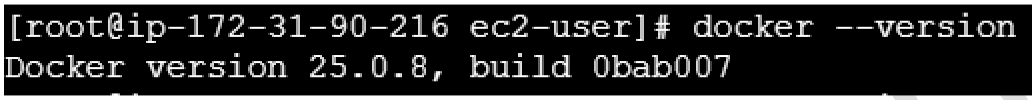
We'll create a new security group called '**launch-wizard-7**' with the following rules:

- [x] Allow SSH traffic from — Anywhere 0.0.0.0/0
  Helps you connect to your instance
- [ ] Allow HTTPS traffic from the internet
  To set up an endpoint, for example when creating a web server
- [ ] Allow HTTP traffic from the internet
  To set up an endpoint, for example when creating a web server

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

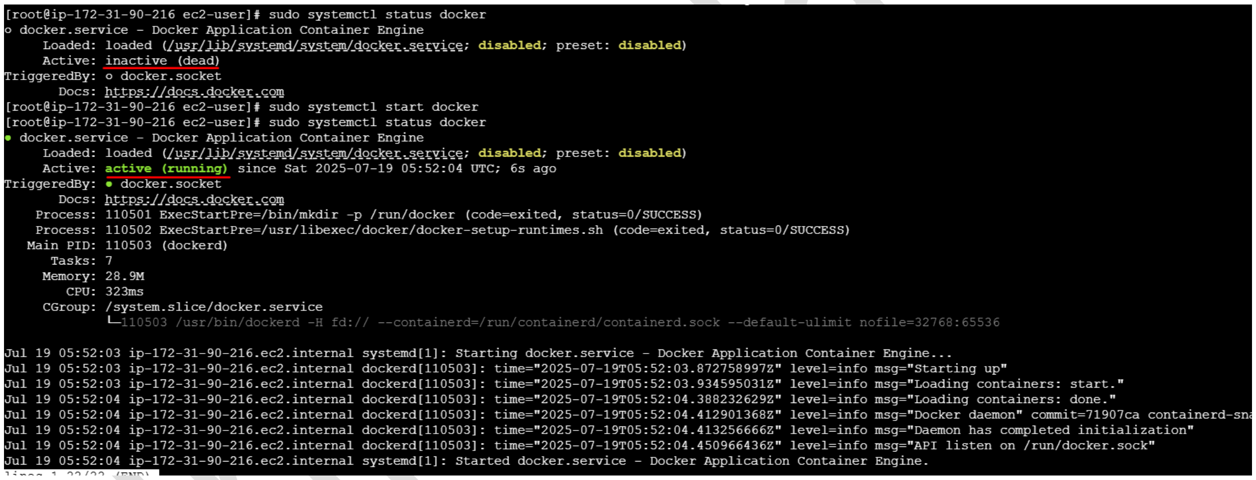
2. Connect to EC2 and install all packages.
   sudo yum install -y
   sudo yum install nginx -y
   sudo yum install docker -y

3. check if docker installed
   docker --version



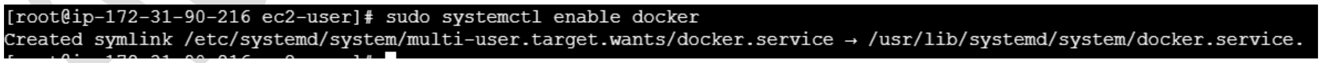
```
[root@ip-172-31-90-216 ec2-user]# docker --version
Docker version 25.0.8, build 0bab007
```

4. If Docker daemon has not started, then start the Docker daemon
   sudo systemctl status docker
   sudo systemctl start docker



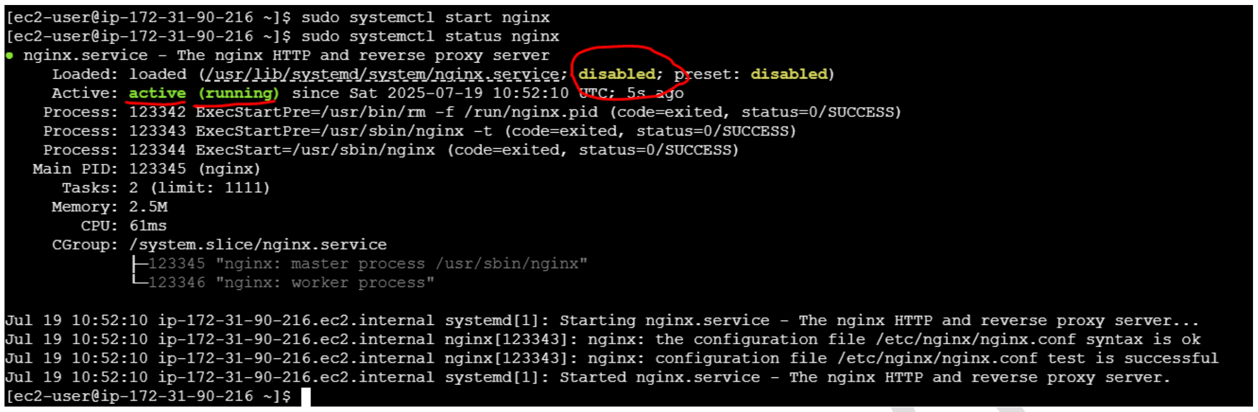
```
[root@ip-172-31-90-216 ec2-user]# sudo systemctl status docker
o docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
     Active: inactive (dead)
TriggeredBy: o docker.socket
       Docs: https://docs.docker.com
[root@ip-172-31-90-216 ec2-user]# sudo systemctl start docker
[root@ip-172-31-90-216 ec2-user]# sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; preset: disabled)
     Active: active (running) since Sat 2025-07-19 05:52:04 UTC; 6s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
    Process: 110501 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0/SUCCESS)
    Process: 110502 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (code=exited, status=0/SUCCESS)
   Main PID: 110503 (dockerd)
      Tasks: 7
     Memory: 28.9M
        CPU: 323ms
     CGroup: /system.slice/docker.service
             └─110503 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Jul 19 05:52:03 ip-172-31-90-216.ec2.internal systemd[1]: Starting docker.service - Docker Application Container Engine...
Jul 19 05:52:03 ip-172-31-90-216.ec2.internal dockerd[110503]: time="2025-07-19T05:52:03.872758997Z" level=info msg="Starting up"
Jul 19 05:52:03 ip-172-31-90-216.ec2.internal dockerd[110503]: time="2025-07-19T05:52:03.934595031Z" level=info msg="Loading containers: start."
Jul 19 05:52:04 ip-172-31-90-216.ec2.internal dockerd[110503]: time="2025-07-19T05:52:04.388232629Z" level=info msg="Loading containers: done."
Jul 19 05:52:04 ip-172-31-90-216.ec2.internal dockerd[110503]: time="2025-07-19T05:52:04.412901368Z" level=info msg="Docker daemon" commit=71907ca containerd-sna
Jul 19 05:52:04 ip-172-31-90-216.ec2.internal dockerd[110503]: time="2025-07-19T05:52:04.413256666Z" level=info msg="Daemon has completed initialization"
Jul 19 05:52:04 ip-172-31-90-216.ec2.internal dockerd[110503]: time="2025-07-19T05:52:04.450966436Z" level=info msg="API listen on /run/docker.sock"
Jul 19 05:52:04 ip-172-31-90-216.ec2.internal systemd[1]: Started docker.service - Docker Application Container Engine.
```

5. Enable the service so that it starts every time a system starts
   sudo systemctl enable docker

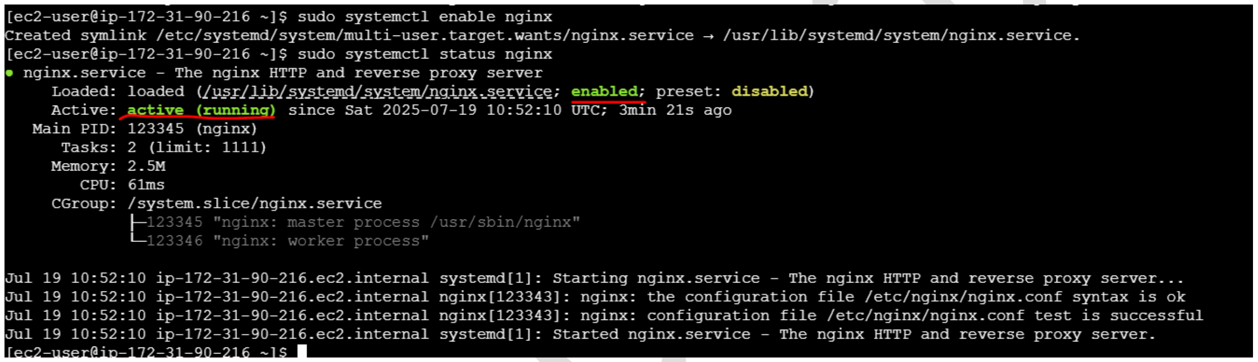


```
[root@ip-172-31-90-216 ec2-user]# sudo systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
```

6. Install, Start and enable nginx service in similar manner
   sudo systemctl status nginx
   sudo systemctl start nginx
   sudo systemctl enable nginx

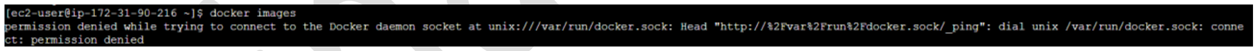


```
[ec2-user@ip-172-31-90-216 ~]$ sudo systemctl start nginx
[ec2-user@ip-172-31-90-216 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; preset: disabled)
     Active: active (running) since Sat 2025-07-19 10:52:10 UTC; 5s ago
    Process: 123342 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 123343 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 123344 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 123345 (nginx)
      Tasks: 2 (limit: 1111)
     Memory: 2.5M
        CPU: 61ms
     CGroup: /system.slice/nginx.service
             ├─123345 "nginx: master process /usr/sbin/nginx"
             └─123346 "nginx: worker process"

Jul 19 10:52:10 ip-172-31-90-216.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Jul 19 10:52:10 ip-172-31-90-216.ec2.internal nginx[123343]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Jul 19 10:52:10 ip-172-31-90-216.ec2.internal nginx[123343]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Jul 19 10:52:10 ip-172-31-90-216.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-172-31-90-216 ~]$
```



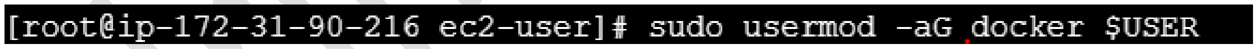
```
[ec2-user@ip-172-31-90-216 ~]$ sudo systemctl enable nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
[ec2-user@ip-172-31-90-216 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Sat 2025-07-19 10:52:10 UTC; 3min 21s ago
   Main PID: 123345 (nginx)
      Tasks: 2 (limit: 1111)
     Memory: 2.5M
        CPU: 61ms
     CGroup: /system.slice/nginx.service
             ├─123345 "nginx: master process /usr/sbin/nginx"
             └─123346 "nginx: worker process"

Jul 19 10:52:10 ip-172-31-90-216.ec2.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Jul 19 10:52:10 ip-172-31-90-216.ec2.internal nginx[123343]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Jul 19 10:52:10 ip-172-31-90-216.ec2.internal nginx[123343]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Jul 19 10:52:10 ip-172-31-90-216.ec2.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-172-31-90-216 ~]$
```

7. To get permission to see images or containers, use SUDO or alternately add the use to the docker user group.



```
[ec2-user@ip-172-31-90-216 ~]$ docker images
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping": dial unix /var/run/docker.sock: conne
ct: permission denied
```

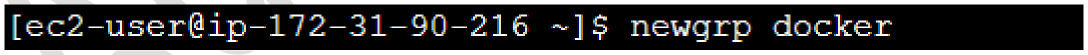
sudo usermod -aG docker $USER



```
[root@ip-172-31-90-216 ec2-user]# sudo usermod -aG docker $USER
```

8. To activate the change, run command below

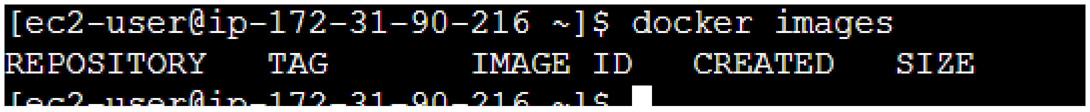
newgrp docker



```
[ec2-user@ip-172-31-90-216 ~]$ newgrp docker
```

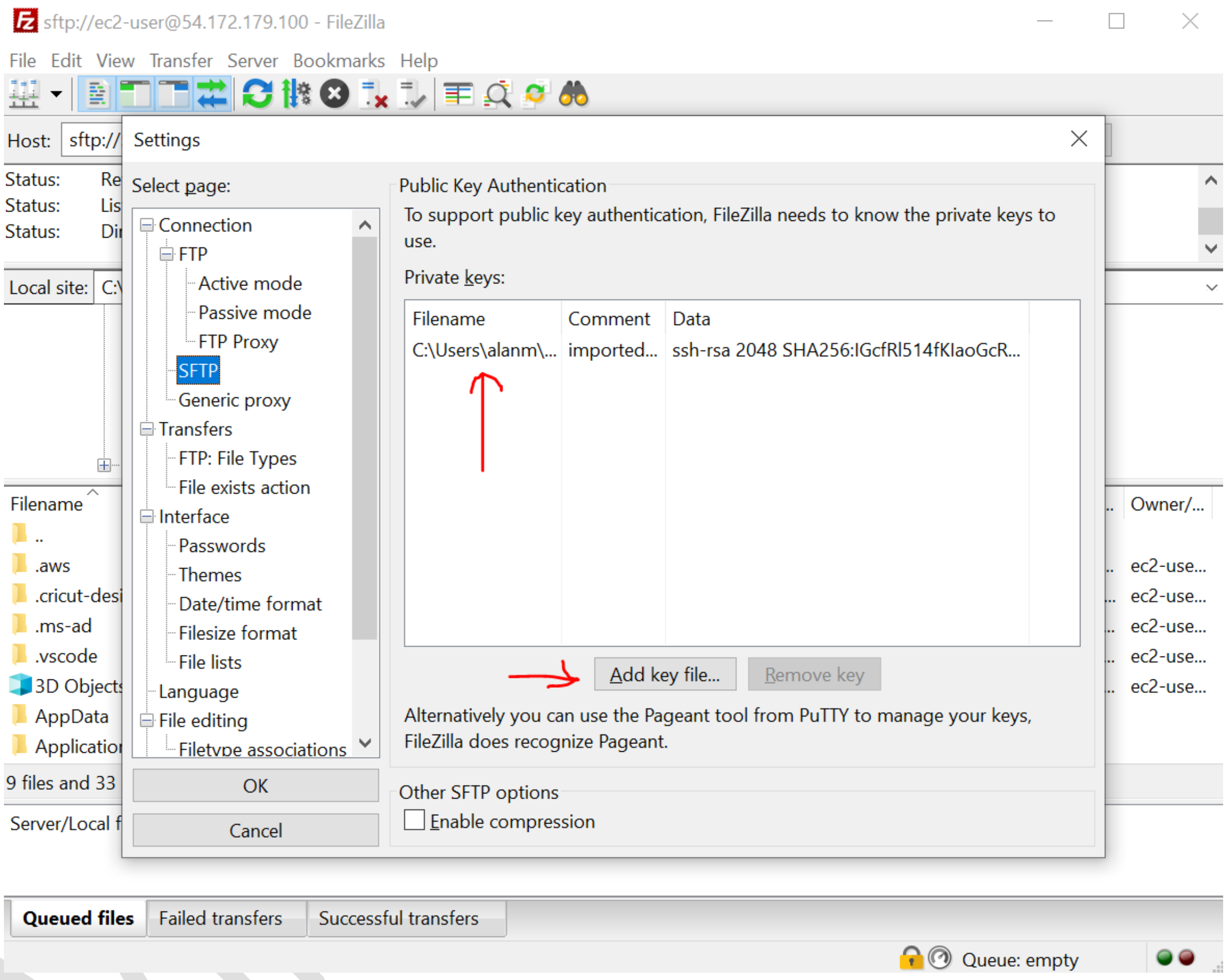
9. Then run

docker images



```
[ec2-user@ip-172-31-90-216 ~]$ docker images
REPOSITORY    TAG         IMAGE ID    CREATED    SIZE
[ec2-user@ip-172-31-90-216 ~]$
```

10. Use FileZilla to move the project located on the local machine to EC2(MobaXterm can also be used)
    Go to Edit → Settings → Under Connection → SFTP → add the PEM key that downloaded while creating your EC2 instance.



Public Key Authentication

To support public key authentication, FileZilla needs to know the private keys to use.

Private keys:

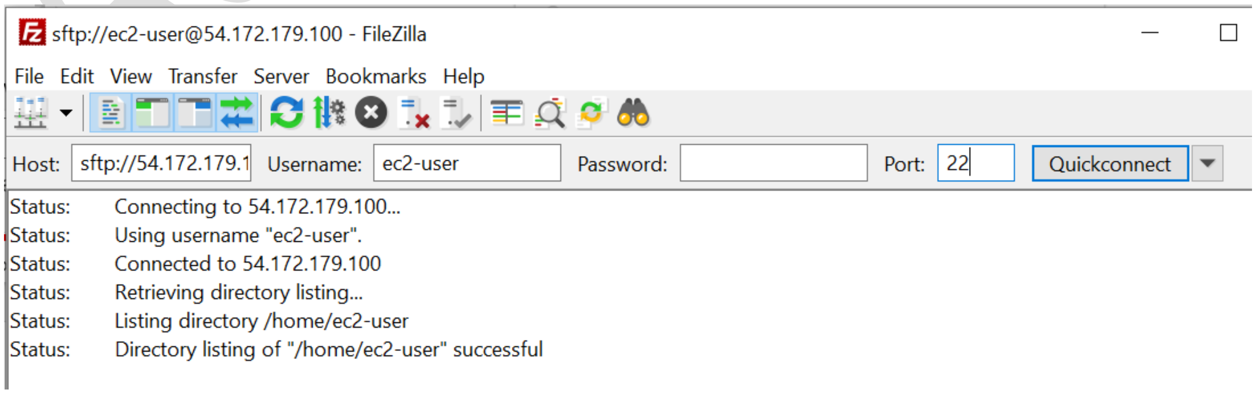
Filename	Comment	Data
C:\Users\alanm\...	imported...	ssh-rsa 2048 SHA256:IGcfRI514fKlaoGcR...

Alternatively you can use the Pageant tool from PuTTY to manage your keys, FileZilla does recognize Pageant.

Other SFTP options

[ ] Enable compression

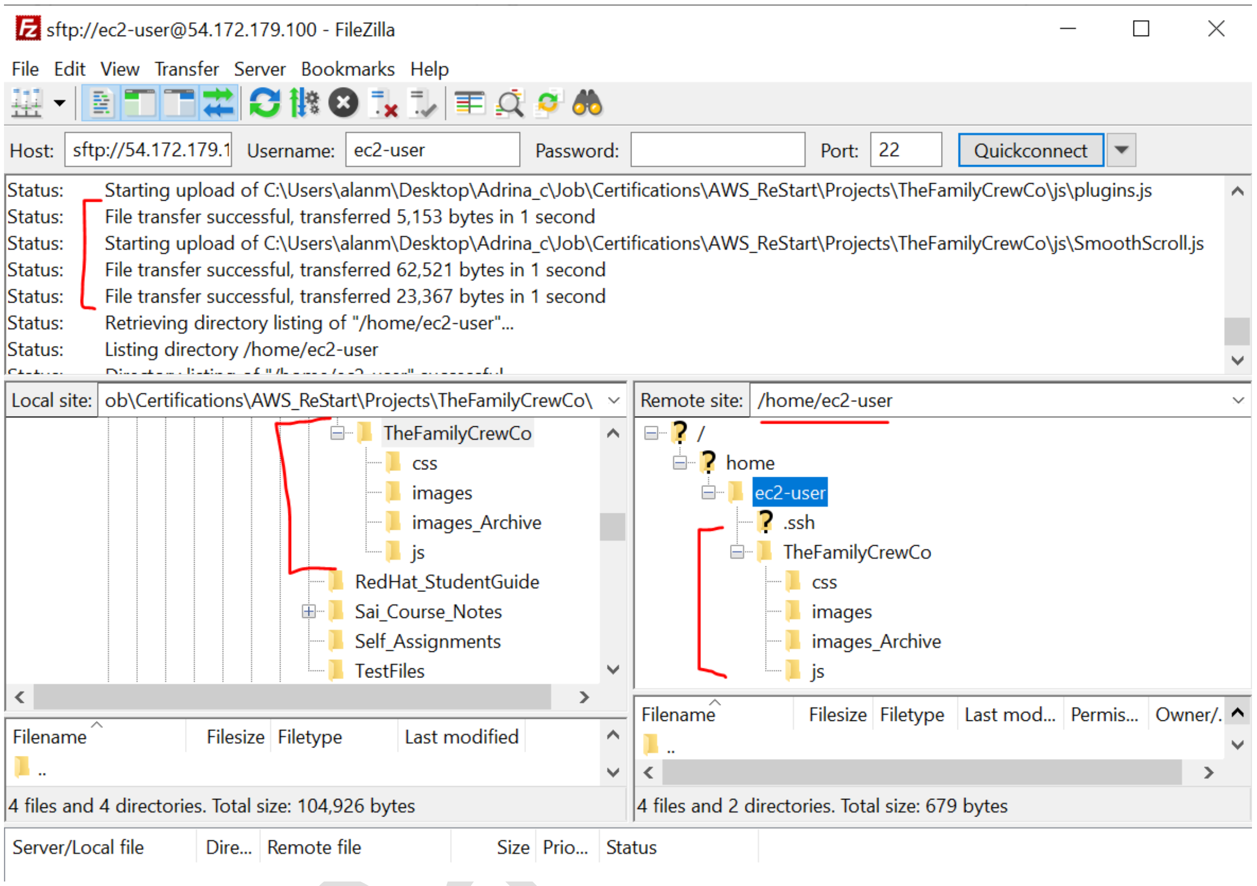
Quick Connect to EC2 by entering the details in the fields.


sftp://ec2-user@54.172.179.100 - FileZilla
File Edit View Transfer Server Bookmarks Help
Quickconnect

Host: sftp://54.172.179.1 Username: ec2-user Password: Port: 22

```
Status:    Connecting to 54.172.179.100...
Status:    Using username "ec2-user".
Status:    Connected to 54.172.179.100
Status:    Retrieving directory listing...
Status:    Listing directory /home/ec2-user
Status:    Directory listing of "/home/ec2-user" successful
```

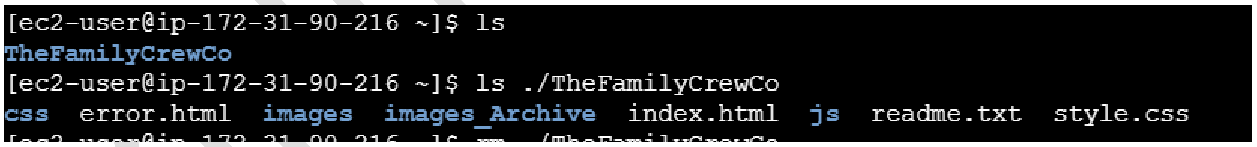
11. Copy the files from local machine to EC2 instance.


sftp://ec2-user@54.172.179.100 - FileZilla
File Edit View Transfer Server Bookmarks Help
Host: sftp://54.172.179.1 Username: ec2-user Password: Port: 22 Quickconnect

```
Status:    Starting upload of C:\Users\alanm\Desktop\Adrina_c\Job\Certifications\AWS_ReStart\Projects\TheFamilyCrewCo\js\plugins.js
Status:    File transfer successful, transferred 5,153 bytes in 1 second
Status:    Starting upload of C:\Users\alanm\Desktop\Adrina_c\Job\Certifications\AWS_ReStart\Projects\TheFamilyCrewCo\js\SmoothScroll.js
Status:    File transfer successful, transferred 62,521 bytes in 1 second
Status:    File transfer successful, transferred 23,367 bytes in 1 second
Status:    Retrieving directory listing of "/home/ec2-user"...
Status:    Listing directory /home/ec2-user
Status:    Directory listing of "/home/ec2-user" successful
```

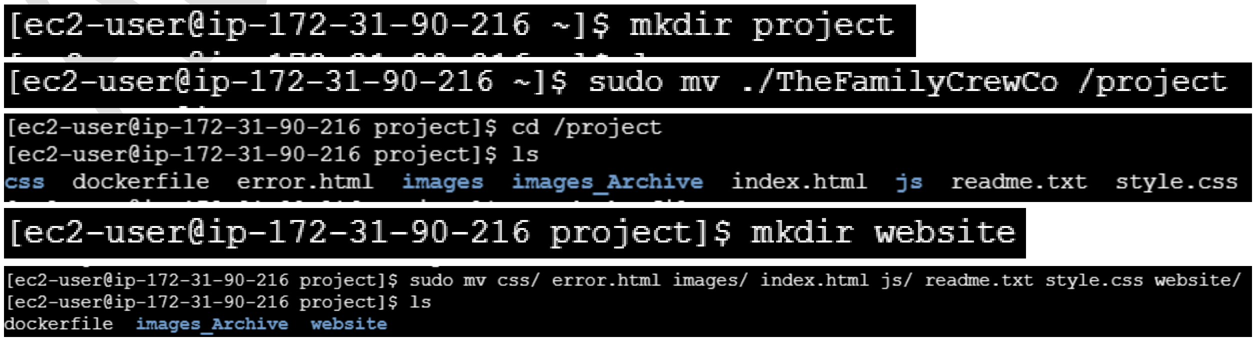
Local site: ob\Certifications\AWS_ReStart\Projects\TheFamilyCrewCo\
TheFamilyCrewCo
css
images
images_Archive
js
RedHat_StudentGuide
Sai_Course_Notes
Self_Assignments
TestFiles
Filename Filesize Filetype Last modified
..
4 files and 4 directories. Total size: 104,926 bytes
Remote site: /home/ec2-user
/
home
ec2-user
.ssh
TheFamilyCrewCo
css
images
images_Archive
js
Filename Filesize Filetype Last mod... Permis... Owner/.
..
4 files and 2 directories. Total size: 679 bytes
Server/Local file Dire... Remote file Size Prio... Status

12. Do “ls” in EC2 command line to verify that all files are copied.



```
[ec2-user@ip-172-31-90-216 ~]$ ls
TheFamilyCrewCo
[ec2-user@ip-172-31-90-216 ~]$ ls ./TheFamilyCrewCo
css  error.html  images  images_Archive  index.html  js  readme.txt  style.css
```

13. Create a folder named “project” and move the website files into it.



```
[ec2-user@ip-172-31-90-216 ~]$ mkdir project
```

```
[ec2-user@ip-172-31-90-216 ~]$ sudo mv ./TheFamilyCrewCo /project
```

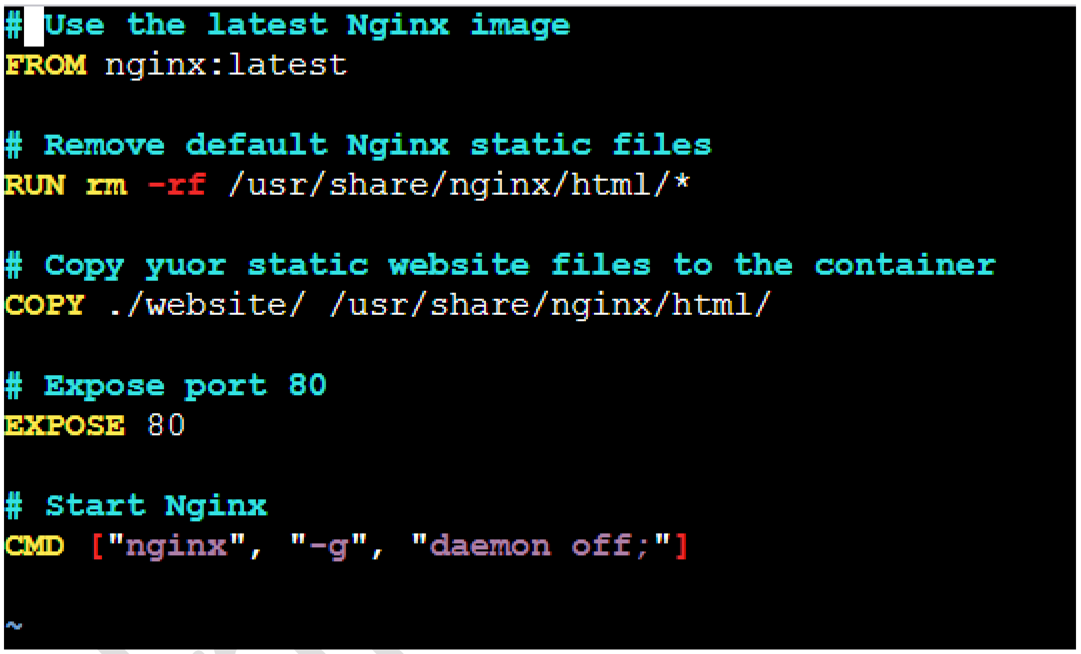
```
[ec2-user@ip-172-31-90-216 project]$ cd /project
[ec2-user@ip-172-31-90-216 project]$ ls
css  dockerfile  error.html  images  images_Archive  index.html  js  readme.txt  style.css
```

```
[ec2-user@ip-172-31-90-216 project]$ mkdir website
```

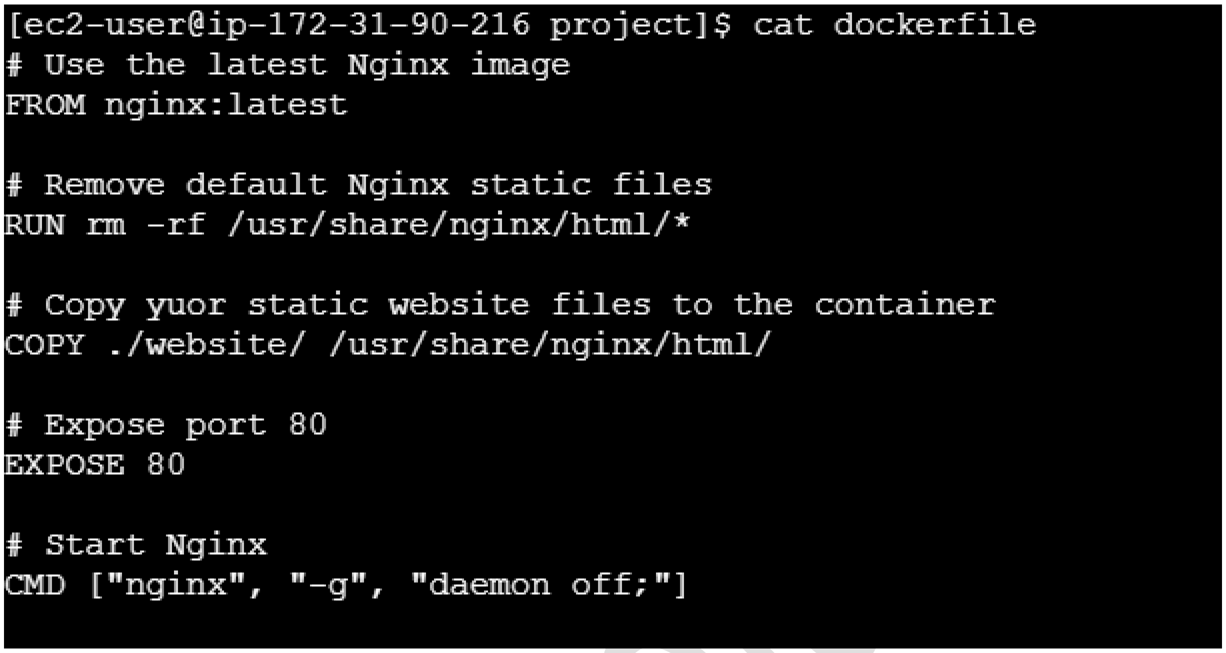
```
[ec2-user@ip-172-31-90-216 project]$ sudo mv css/ error.html images/ index.html js/ readme.txt style.css website/
[ec2-user@ip-172-31-90-216 project]$ ls
dockerfile  images_Archive  website
```

Create DockerFile

14. Create docker file inside “project” folder where website folder also resides.
    nano dockerfile
    Inside the file , make sure that the quotes are correct if you copy paste
        FROM nginx:latest
        COPY . /usr/share/nginx/html
        EXPOSE 80
        CMD [“nginx”, “-g”, “daemon off;”]



```
# Use the latest Nginx image
FROM nginx:latest

# Remove default Nginx static files
RUN rm -rf /usr/share/nginx/html/*

# Copy yuor static website files to the container
COPY ./website/ /usr/share/nginx/html/

# Expose port 80
EXPOSE 80

# Start Nginx
CMD ["nginx", "-g", "daemon off;"]

~
```

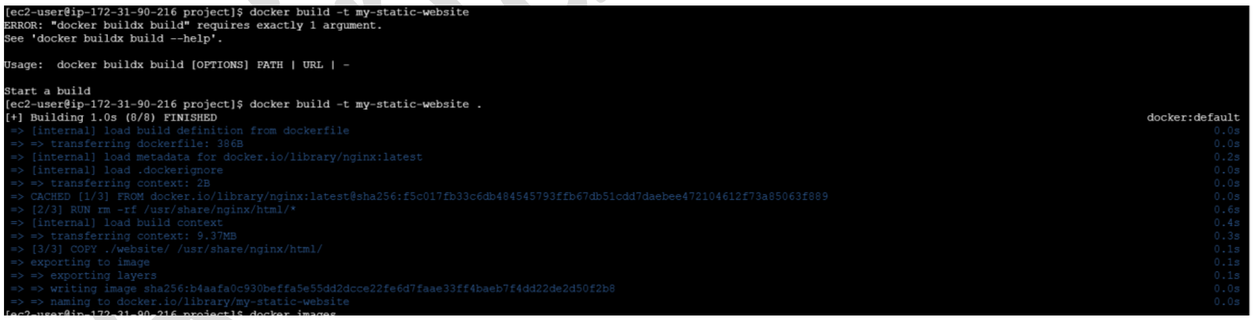


```
[ec2-user@ip-172-31-90-216 project]$ cat dockerfile
# Use the latest Nginx image
FROM nginx:latest

# Remove default Nginx static files
RUN rm -rf /usr/share/nginx/html/*

# Copy yuor static website files to the container
COPY ./website/ /usr/share/nginx/html/

# Expose port 80
EXPOSE 80

# Start Nginx
CMD ["nginx", "-g", "daemon off;"]
```

Create docker image

docker build -t my-static-website .

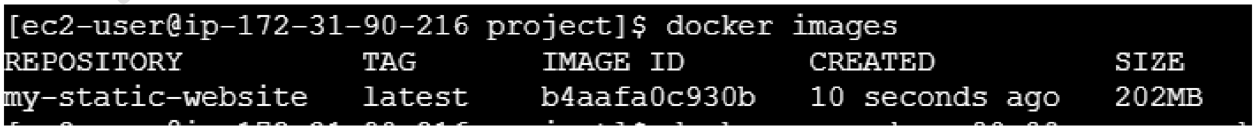
(The dot(.) at the end of above command is to tell that the docker file is in the local directory.



```
[ec2-user@ip-172-31-90-216 project]$ docker build -t my-static-website
ERROR: "docker buildx build" requires exactly 1 argument.
See 'docker buildx build --help'.

Usage:  docker buildx build [OPTIONS] PATH | URL | -

Start a build
[ec2-user@ip-172-31-90-216 project]$ docker build -t my-static-website .
[+] Building 1.0s (8/8) FINISHED                                                                                             docker:default
 => [internal] load build definition from dockerfile                                                                                   0.0s
 => => transferring dockerfile: 386B                                                                                                   0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                                                        0.2s
 => [internal] load .dockerignore                                                                                                      0.0s
 => => transferring context: 2B                                                                                                        0.0s
 => CACHED [1/3] FROM docker.io/library/nginx:latest@sha256:f5c017fb33c6db484545793ffb67db51cdd7daebee472104612f73a85063f889            0.0s
 => [2/3] RUN rm -rf /usr/share/nginx/html/*                                                                                           0.6s
 => [internal] load build context                                                                                                      0.4s
 => => transferring context: 9.37MB                                                                                                    0.3s
 => [3/3] COPY ./website/ /usr/share/nginx/html/                                                                                       0.1s
 => exporting to image                                                                                                                 0.1s
 => => exporting layers                                                                                                                0.1s
 => => writing image sha256:b4aafa0c930beffa5e55dd2dcce22fe6d7faae33ff4baeb7f4dd22de2d50f2b8                                           0.0s
 => => naming to docker.io/library/my-static-website                                                                                   0.0s
```

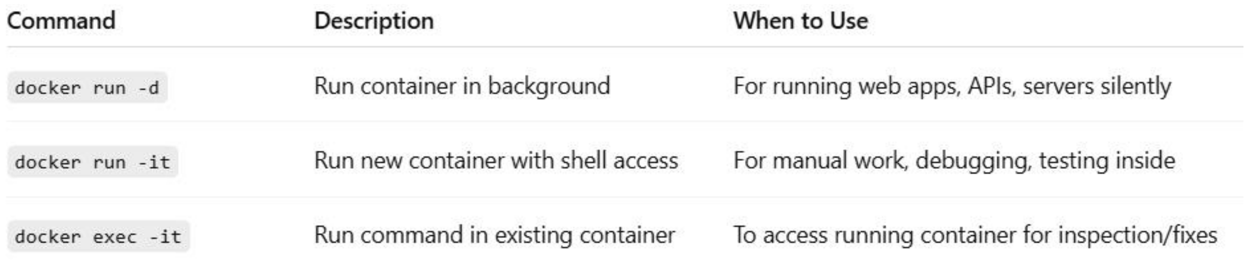
15. To view all images

docker images



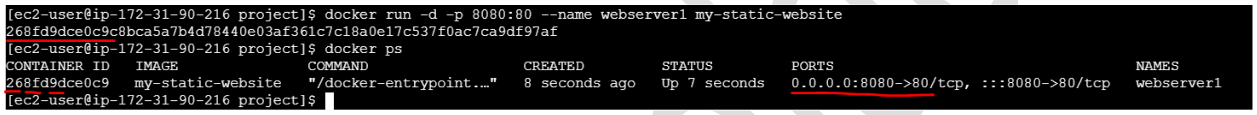
```
[ec2-user@ip-172-31-90-216 project]$ docker images
REPOSITORY          TAG       IMAGE ID       CREATED          SIZE
my-static-website   latest    b4aafa0c930b   10 seconds ago   202MB
```

Different ways to run the docker images



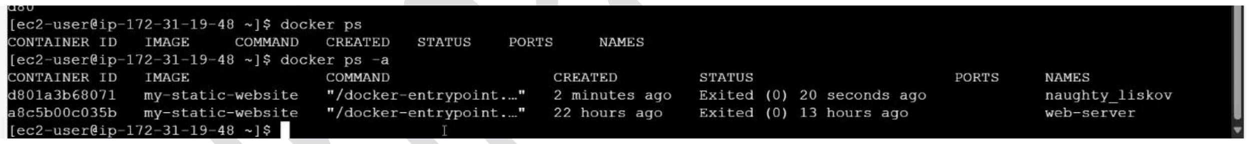
Command	Description	When to Use
`docker run -d`	Run container in background	For running web apps, APIs, servers silently
`docker run -it`	Run new container with shell access	For manual work, debugging, testing inside
`docker exec -it`	Run command in existing container	To access running container for inspection/fixes

16. Run the docker image by doing port forwarding
    docker run -d -p 8080:80 --name webserver my-static-website
    -p 8080:80 is to map the port 8080 of the EC2 instance to port 80 of the container



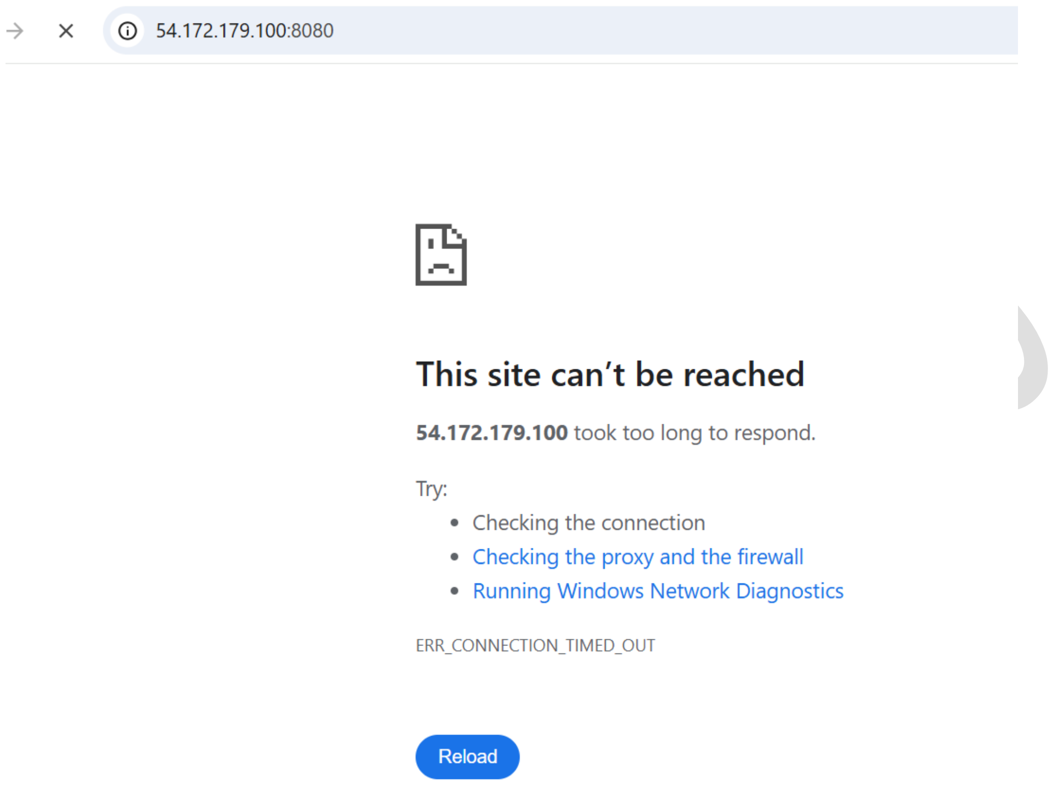
```
[ec2-user@ip-172-31-90-216 project]$ docker run -d -p 8080:80 --name webserver1 my-static-website
268fd9dce0c9c8bca5a7b4d78440e03af361c7c18a0e17c537f0ac7ca9df97af
[ec2-user@ip-172-31-90-216 project]$ docker ps
CONTAINER ID   IMAGE               COMMAND                  CREATED         STATUS         PORTS                                   NAMES
268fd9dce0c9   my-static-website   "/docker-entrypoint.…"   8 seconds ago   Up 7 seconds   0.0.0.0:8080->80/tcp, :::8080->80/tcp   webserver1
[ec2-user@ip-172-31-90-216 project]$
```

17. To view Running Containers
    docker ps
    To view **All** containers
    docker ps -a



```
d80
[ec2-user@ip-172-31-19-48 ~]$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[ec2-user@ip-172-31-19-48 ~]$ docker ps -a
CONTAINER ID   IMAGE               COMMAND                  CREATED          STATUS                      PORTS     NAMES
d801a3b68071   my-static-website   "/docker-entrypoint.…"   2 minutes ago    Exited (0) 20 seconds ago             naughty_liskov
a8c5b00c035b   my-static-website   "/docker-entrypoint.…"   22 hours ago     Exited (0) 13 hours ago               web-server
[ec2-user@ip-172-31-19-48 ~]$
```

18. Check if the (Public IP of instance) website is working. It won't work because port 8080 has not been added to the security group.



54.172.179.100:8080

This site can’t be reached

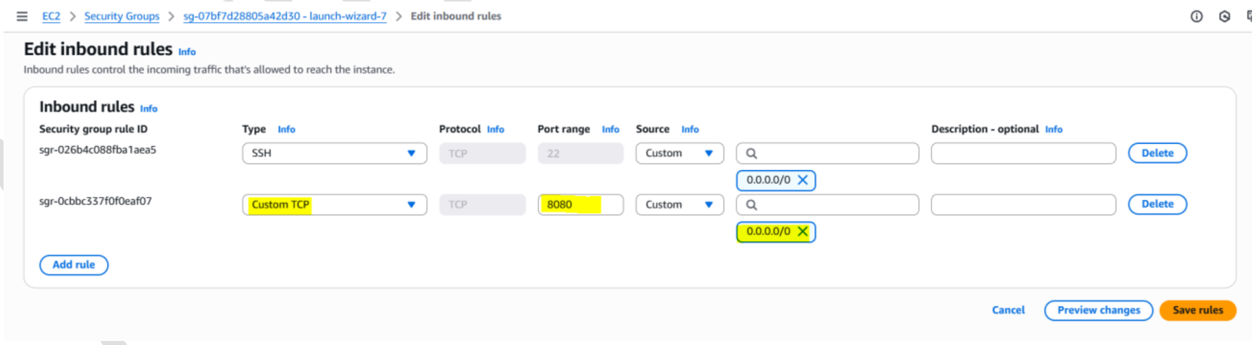
54.172.179.100 took too long to respond.

Try:

- Checking the connection
- Checking the proxy and the firewall
- Running Windows Network Diagnostics

ERR_CONNECTION_TIMED_OUT

19. Add port 8080 to the security group.



EC2 > Security Groups > sg-07bf7d28805a42d30 - launch-wizard-7 > Edit inbound rules

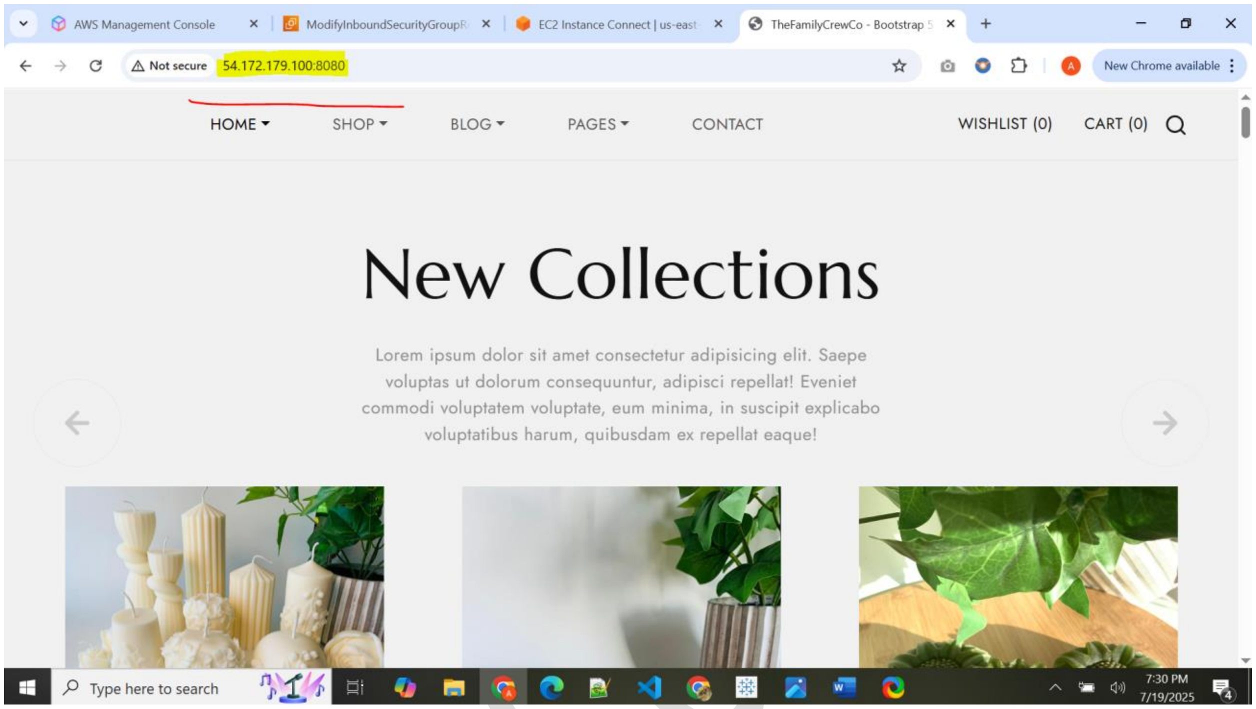
Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

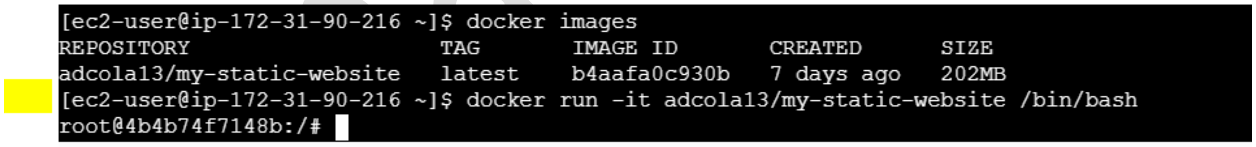
Security group rule ID	Type	Protocol	Port range	Source		Description - optional	
sgr-026b4c088fba1aea5	SSH	TCP	22	Custom	0.0.0.0/0		Delete
sgr-0cbbc337f0f0eaf07	Custom TCP	TCP	8080	Custom	0.0.0.0/0		Delete

20. Refresh the webpage


AWS Management Console
ModifyInboundSecurityGroupR
EC2 Instance Connect | us-east
TheFamilyCrewCo - Bootstrap 5
Not secure
54.172.179.100:8080
New Chrome available
HOME
SHOP
BLOG
PAGES
CONTACT
WISHLIST (0)
CART (0)
New Collections
Lorem ipsum dolor sit amet consectetur adipisicing elit. Saepe voluptas ut dolorum consequuntur, adipisci repellat! Eveniet commodi voluptatem voluptate, eum minima, in suscipit explicabo voluptatibus harum, quibusdam ex repellat eaque!
Type here to search
7:30 PM
7/19/2025

To run a container in "interaction" mode

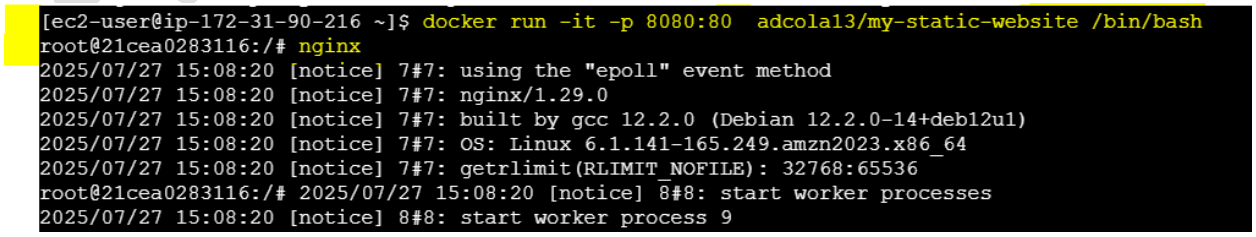
docker run -it adcola13/my-static-website /bin/bash



```
[ec2-user@ip-172-31-90-216 ~]$ docker images
REPOSITORY                    TAG       IMAGE ID       CREATED       SIZE
adcola13/my-static-website    latest    b4aafa0c930b   7 days ago    202MB
[ec2-user@ip-172-31-90-216 ~]$ docker run -it adcola13/my-static-website /bin/bash
root@4b4b74f7148b:/# 
```

Interactive bin/bash mode will override CMD from the Docker image. You won't see your website yet, so you need to run the web server explicitly. i.e. start nginx

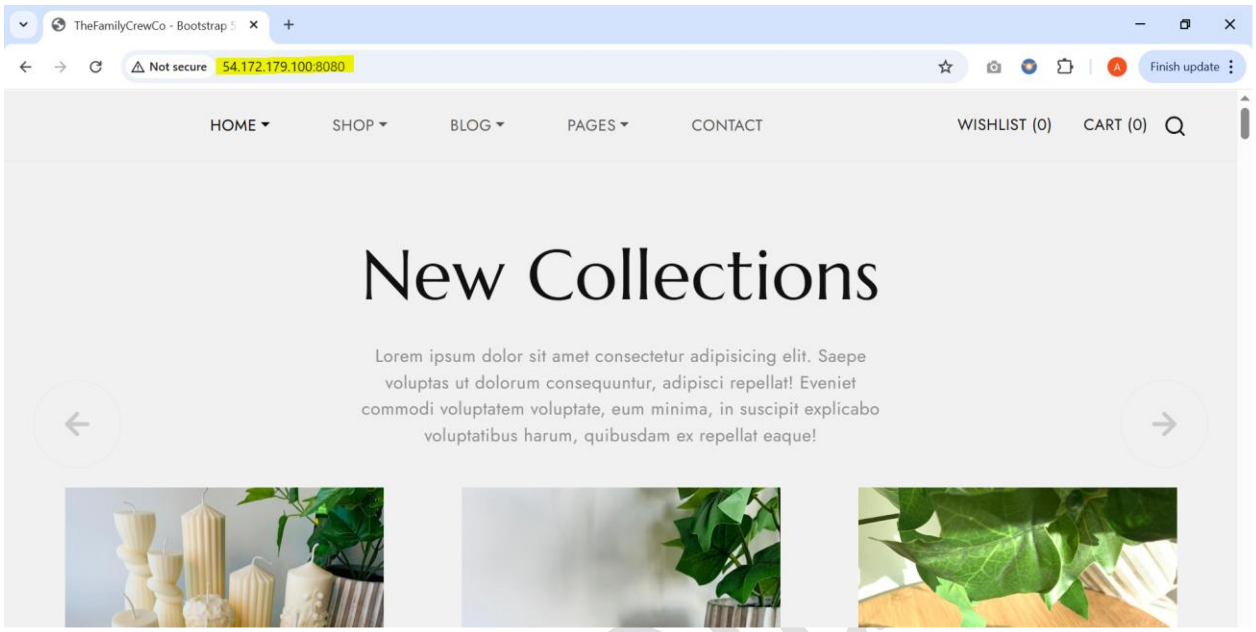
To run NGINX inside bin bash, simply type nginx in the BASH CLI.



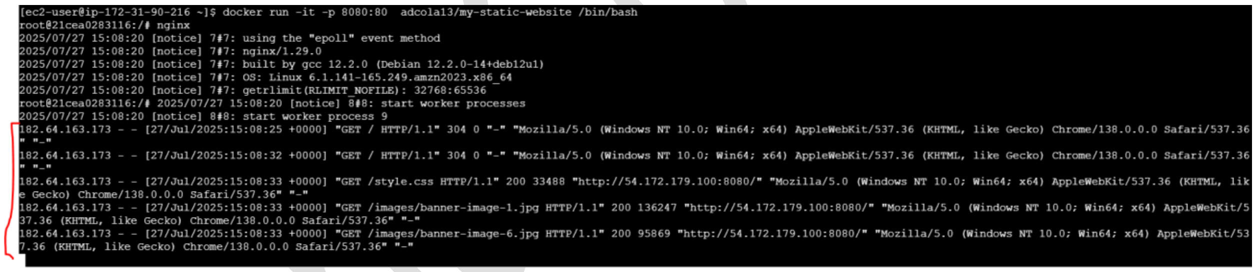
```
[ec2-user@ip-172-31-90-216 ~]$ docker run -it -p 8080:80  adcola13/my-static-website /bin/bash
root@21cea0283116:/# nginx
2025/07/27 15:08:20 [notice] 7#7: using the "epoll" event method
2025/07/27 15:08:20 [notice] 7#7: nginx/1.29.0
2025/07/27 15:08:20 [notice] 7#7: built by gcc 12.2.0 (Debian 12.2.0-14+deb12u1)
2025/07/27 15:08:20 [notice] 7#7: OS: Linux 6.1.141-165.249.amzn2023.x86_64
2025/07/27 15:08:20 [notice] 7#7: getrlimit(RLIMIT_NOFILE): 32768:65536
root@21cea0283116:/# 2025/07/27 15:08:20 [notice] 8#8: start worker processes
2025/07/27 15:08:20 [notice] 8#8: start worker process 9
```

You may feel that the process is **STUCK**, but NGINX is running in the background at this stage.

Run the public address redirecting to port 8080 as shown below.

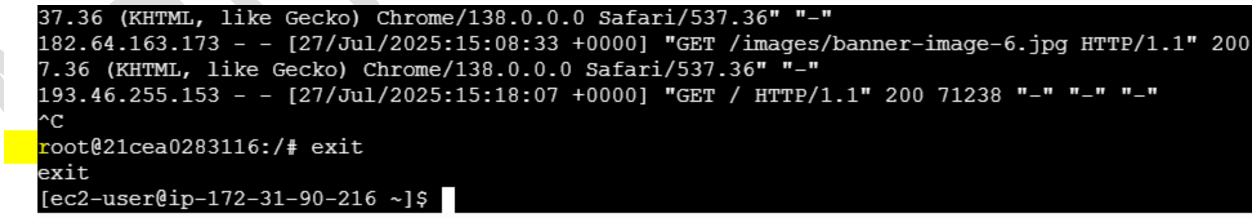

TheFamilyCrewCo - Bootstrap 5
Not secure
54.172.179.100:8080
Finish update
HOME
SHOP
BLOG
PAGES
CONTACT
WISHLIST (0)
CART (0)
New Collections
Lorem ipsum dolor sit amet consectetur adipisicing elit. Saepe voluptas ut dolorum consequuntur, adipisci repellat! Eveniet commodi voluptatem voluptate, eum minima, in suscipit explicabo voluptatibus harum, quibusdam ex repellat eaque!

If you go back to CLI screen, you will see some activity in BASH.

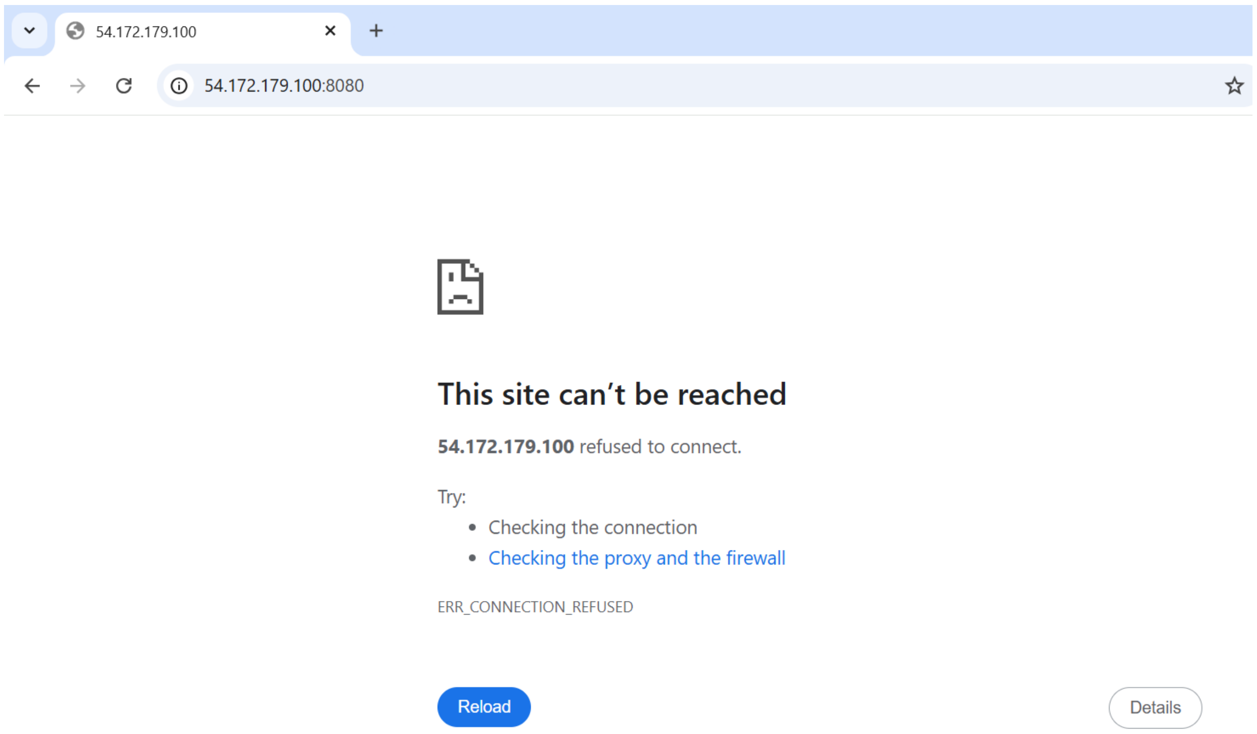


```
[ec2-user@ip-172-31-90-216 ~]$ docker run -it -p 8080:80  adcola13/my-static-website /bin/bash
root@21cea0283116:/# nginx
2025/07/27 15:08:20 [notice] 7#7: using the "epoll" event method
2025/07/27 15:08:20 [notice] 7#7: nginx/1.29.0
2025/07/27 15:08:20 [notice] 7#7: built by gcc 12.2.0 (Debian 12.2.0-14+deb12u1)
2025/07/27 15:08:20 [notice] 7#7: OS: Linux 6.1.141-165.249.amzn2023.x86_64
2025/07/27 15:08:20 [notice] 7#7: getrlimit(RLIMIT_NOFILE): 32768:65536
root@21cea0283116:/# 2025/07/27 15:08:20 [notice] 8#8: start worker processes
2025/07/27 15:08:20 [notice] 8#8: start worker process 9
182.64.163.173 - - [27/Jul/2025:15:08:25 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
" "-"
182.64.163.173 - - [27/Jul/2025:15:08:32 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36
" "-"
182.64.163.173 - - [27/Jul/2025:15:08:33 +0000] "GET /style.css HTTP/1.1" 200 33488 "http://54.172.179.100:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, lik
e Gecko) Chrome/138.0.0.0 Safari/537.36" "-"
182.64.163.173 - - [27/Jul/2025:15:08:33 +0000] "GET /images/banner-image-1.jpg HTTP/1.1" 200 136247 "http://54.172.179.100:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/5
37.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36" "-"
182.64.163.173 - - [27/Jul/2025:15:08:33 +0000] "GET /images/banner-image-6.jpg HTTP/1.1" 200 95869 "http://54.172.179.100:8080/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/53
7.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36" "-"
```

If you type "exit" command, it will stop the container as well.
Use CTRL+p and CTRL+q to exit without stopping the container.



```
37.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36" "-"
182.64.163.173 - - [27/Jul/2025:15:08:33 +0000] "GET /images/banner-image-6.jpg HTTP/1.1" 200
7.36 (KHTML, like Gecko) Chrome/138.0.0.0 Safari/537.36" "-"
193.46.255.153 - - [27/Jul/2025:15:18:07 +0000] "GET / HTTP/1.1" 200 71238 "-" "-" "-"
^C
root@21cea0283116:/# exit
exit
[ec2-user@ip-172-31-90-216 ~]$
```


54.172.179.100:8080

This site can't be reached

54.172.179.100 refused to connect.

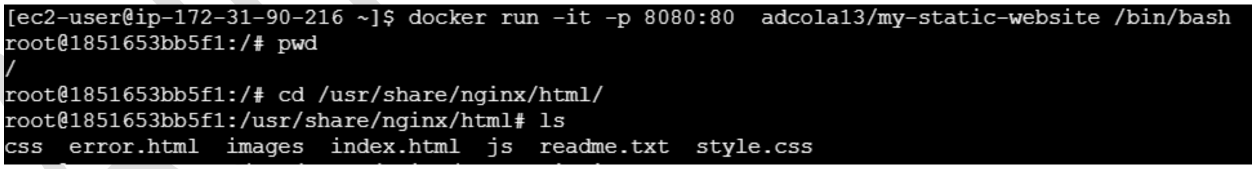
Try:

- Checking the connection
- Checking the proxy and the firewall

ERR_CONNECTION_REFUSED

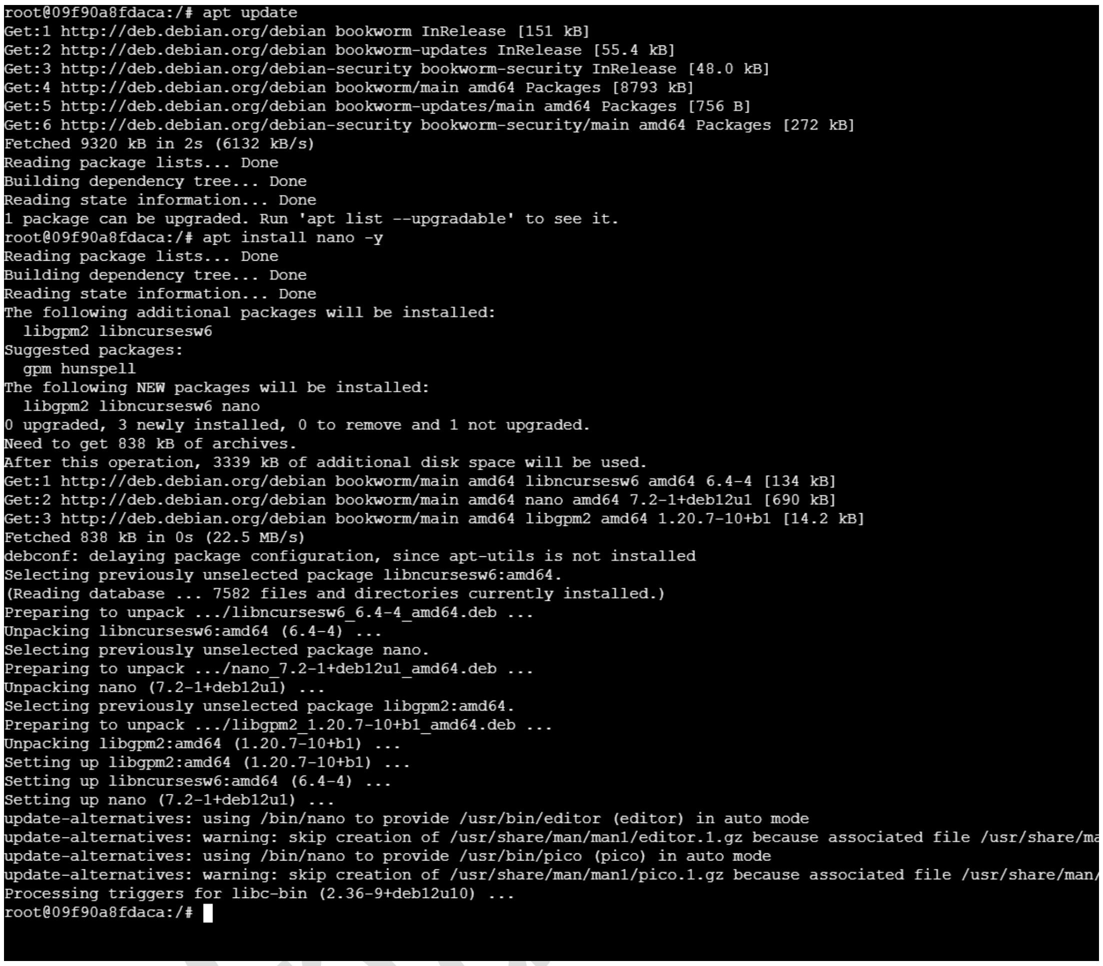
To change the project files within the container

21. If you want to make changes only to the project files stored in the container, use interactive mode like below.
docker run -it -p 8080:80 adcola13/my-static-website /bin/bash
Go to the below folder and modify the project file (in this case, the index.html file)
cd /usr/share/nginx/html/



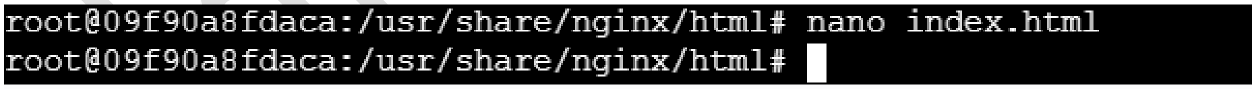
```
[ec2-user@ip-172-31-90-216 ~]$ docker run -it -p 8080:80  adcola13/my-static-website /bin/bash
root@1851653bb5f1:/# pwd
/
root@1851653bb5f1:/# cd /usr/share/nginx/html/
root@1851653bb5f1:/usr/share/nginx/html# ls
css  error.html  images  index.html  js  readme.txt  style.css
```

At this point, if "nano" doesn't exist, then install it by running the commands below; otherwise, you should be able to modify the index.html by running the nano command.
apt update
apt install nano -y

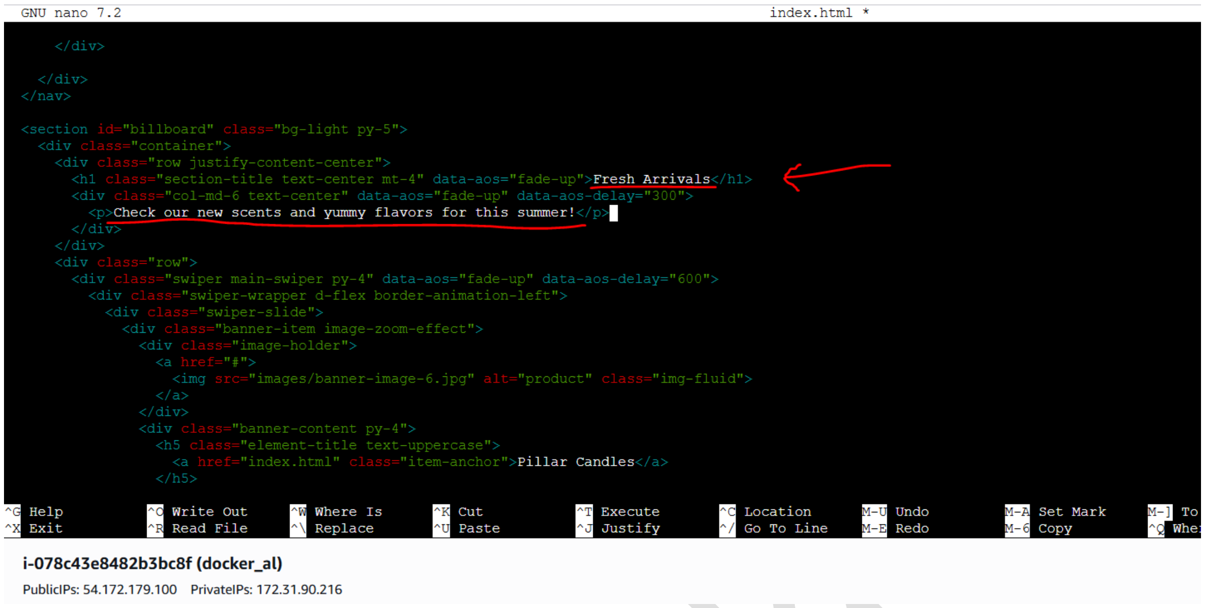


```
root@09f90a8fdaca:/# apt update
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8793 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [756 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [272 kB]
Fetched 9320 kB in 2s (6132 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
root@09f90a8fdaca:/# apt install nano -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libgpm2 libncursesw6
Suggested packages:
  gpm hunspell
The following NEW packages will be installed:
  libgpm2 libncursesw6 nano
0 upgraded, 3 newly installed, 0 to remove and 1 not upgraded.
Need to get 838 kB of archives.
After this operation, 3339 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 libncursesw6 amd64 6.4-4 [134 kB]
Get:2 http://deb.debian.org/debian bookworm/main amd64 nano amd64 7.2-1+deb12u1 [690 kB]
Get:3 http://deb.debian.org/debian bookworm/main amd64 libgpm2 amd64 1.20.7-10+b1 [14.2 kB]
Fetched 838 kB in 0s (22.5 MB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package libncursesw6:amd64.
(Reading database ... 7582 files and directories currently installed.)
Preparing to unpack .../libncursesw6_6.4-4_amd64.deb ...
Unpacking libncursesw6:amd64 (6.4-4) ...
Selecting previously unselected package nano.
Preparing to unpack .../nano_7.2-1+deb12u1_amd64.deb ...
Unpacking nano (7.2-1+deb12u1) ...
Selecting previously unselected package libgpm2:amd64.
Preparing to unpack .../libgpm2_1.20.7-10+b1_amd64.deb ...
Unpacking libgpm2:amd64 (1.20.7-10+b1) ...
Setting up libgpm2:amd64 (1.20.7-10+b1) ...
Setting up libncursesw6:amd64 (6.4-4) ...
Setting up nano (7.2-1+deb12u1) ...
update-alternatives: using /bin/nano to provide /usr/bin/editor (editor) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/editor.1.gz because associated file /usr/share/ma
update-alternatives: using /bin/nano to provide /usr/bin/pico (pico) in auto mode
update-alternatives: warning: skip creation of /usr/share/man/man1/pico.1.gz because associated file /usr/share/man/
Processing triggers for libc-bin (2.36-9+deb12u10) ...
root@09f90a8fdaca:/#
```

Modify the index.html file and save it.



```
root@09f90a8fdaca:/usr/share/nginx/html# nano index.html
root@09f90a8fdaca:/usr/share/nginx/html#
```

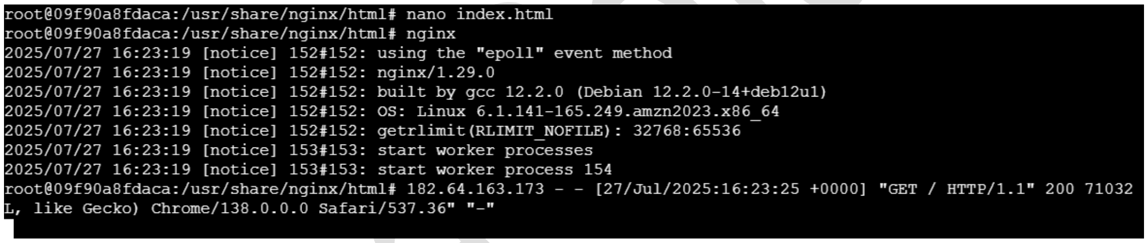


```
GNU nano 7.2                                                          index.html *

    </div>

  </div>
</nav>

<section id="billboard" class="bg-light py-5">
  <div class="container">
    <div class="row justify-content-center">
      <h1 class="section-title text-center mt-4" data-aos="fade-up">Fresh Arrivals</h1>
      <div class="col-md-6 text-center" data-aos="fade-up" data-aos-delay="300">
        <p>Check our new scents and yummy flavors for this summer!</p>
      </div>
    </div>
    <div class="row">
      <div class="swiper main-swiper py-4" data-aos="fade-up" data-aos-delay="600">
        <div class="swiper-wrapper d-flex border-animation-left">
          <div class="swiper-slide">
            <div class="banner-item image-zoom-effect">
              <div class="image-holder">
                <a href="#">
                  <img src="images/banner-image-6.jpg" alt="product" class="img-fluid">
                </a>
              </div>
              <div class="banner-content py-4">
                <h5 class="element-title text-uppercase">
                  <a href="index.html" class="item-anchor">Pillar Candles</a>
                </h5>
^G Help          ^O Write Out     ^W Where Is      ^K Cut           ^T Execute       ^C Location      M-U Undo         M-A Set Mark     M-] To
^X Exit          ^R Read File     ^\ Replace       ^U Paste         ^J Justify       ^/ Go To Line    M-E Redo         M-6 Copy         ^Q Whe
```

i-078c43e8482b3bc8f (docker_al)

PublicIPs: 54.172.179.100 PrivateIPs: 172.31.90.216

Install nginx if the website is still not working.

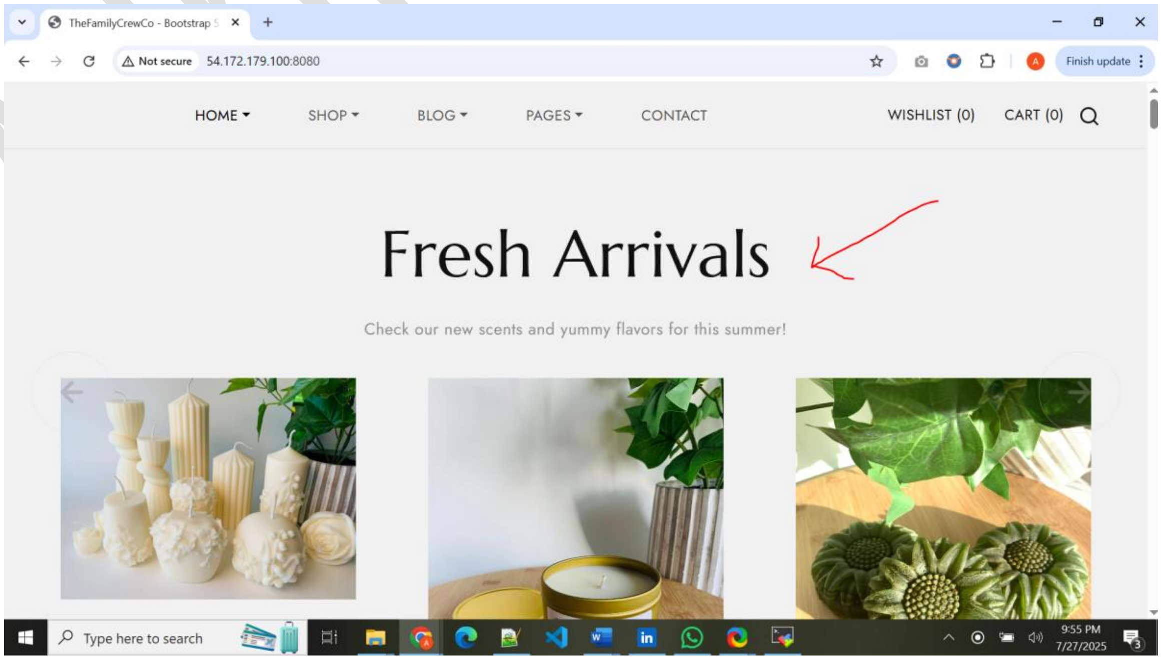


```
root@09f90a8fdaca:/usr/share/nginx/html# nano index.html
root@09f90a8fdaca:/usr/share/nginx/html# nginx
2025/07/27 16:23:19 [notice] 152#152: using the "epoll" event method
2025/07/27 16:23:19 [notice] 152#152: nginx/1.29.0
2025/07/27 16:23:19 [notice] 152#152: built by gcc 12.2.0 (Debian 12.2.0-14+deb12u1)
2025/07/27 16:23:19 [notice] 152#152: OS: Linux 6.1.141-165.249.amzn2023.x86_64
2025/07/27 16:23:19 [notice] 152#152: getrlimit(RLIMIT_NOFILE): 32768:65536
2025/07/27 16:23:19 [notice] 153#153: start worker processes
2025/07/27 16:23:19 [notice] 153#153: start worker process 154
root@09f90a8fdaca:/usr/share/nginx/html# 182.64.163.173 - - [27/Jul/2025:16:23:25 +0000] "GET / HTTP/1.1" 200 71032
L, like Gecko) Chrome/138.0.0.0 Safari/537.36" "-"
```

Verify that the changes reflect on the website.

In this scenario, I have modified the heading and context below it.

The heading was "New collections" changed to "Fresh Arrivals"

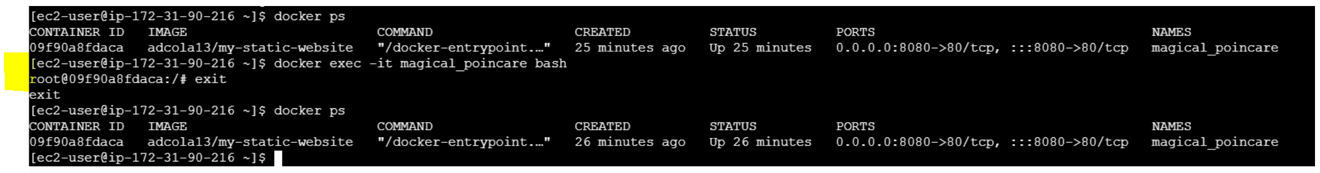


To enter a running container(EXEC Mode)

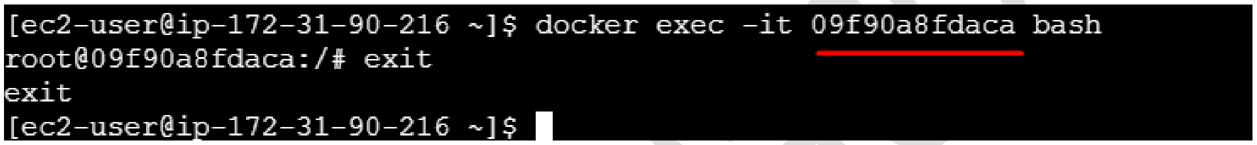
docker exec -it

Here, you can use the exit command without stopping the container.

Also, note how I used the container name; you may use the container ID instead.



```
[ec2-user@ip-172-31-90-216 ~]$ docker ps
CONTAINER ID   IMAGE                        COMMAND                  CREATED          STATUS          PORTS                                   NAMES
09f90a8fdaca   adcola13/my-static-website   "/docker-entrypoint.…"   25 minutes ago   Up 25 minutes   0.0.0.0:8080->80/tcp, :::8080->80/tcp   magical_poincare
[ec2-user@ip-172-31-90-216 ~]$ docker exec -it magical_poincare bash
root@09f90a8fdaca:/# exit
exit
[ec2-user@ip-172-31-90-216 ~]$ docker ps
CONTAINER ID   IMAGE                        COMMAND                  CREATED          STATUS          PORTS                                   NAMES
09f90a8fdaca   adcola13/my-static-website   "/docker-entrypoint.…"   26 minutes ago   Up 26 minutes   0.0.0.0:8080->80/tcp, :::8080->80/tcp   magical_poincare
[ec2-user@ip-172-31-90-216 ~]$
```



```
[ec2-user@ip-172-31-90-216 ~]$ docker exec -it 09f90a8fdaca bash
root@09f90a8fdaca:/# exit
exit
[ec2-user@ip-172-31-90-216 ~]$
```

Use of C groups to allocate CPU, memory or I/O space for containers

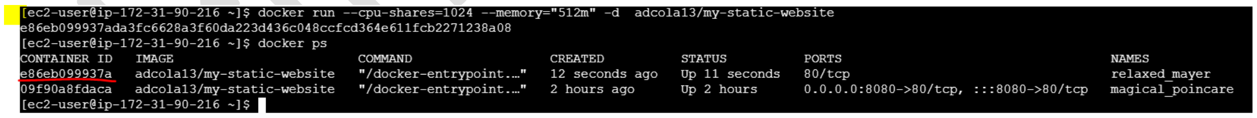
Using Control Groups (C groups) in Docker allows you to allocate and limit system resources such as memory, CPU, and I/O for containers at the time of their creation. When running a Docker container, you can pass specific flags during the docker run command to configure the resources allocated to the container using cgroups.

docker run --cpu-shares=512 your_image

docker run --memory="512m" your_image

docker run --blkio-weight=300 your_image

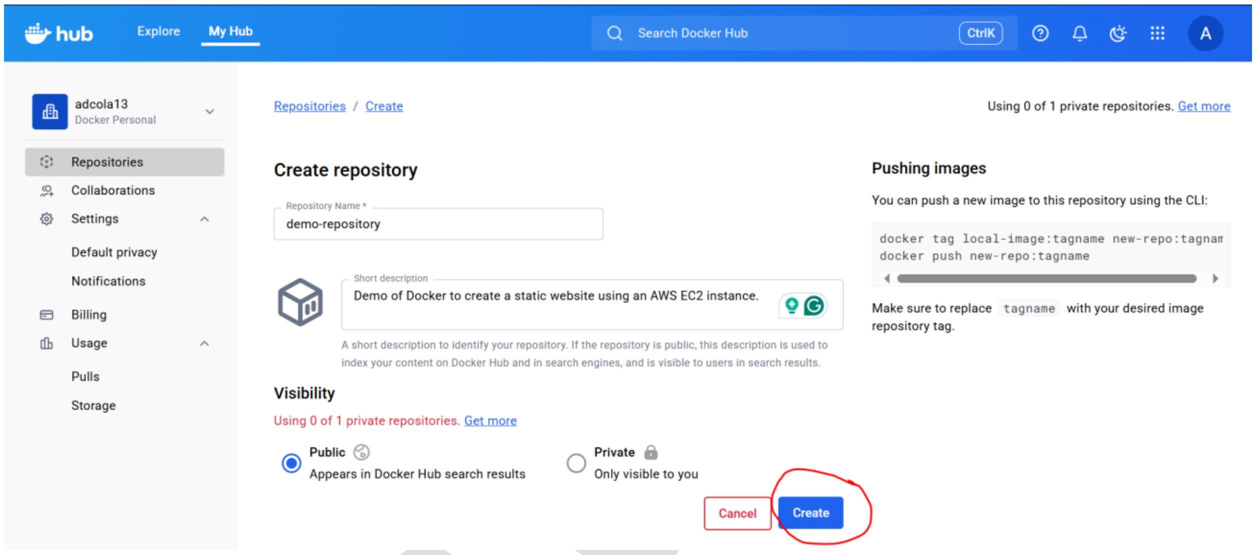
docker run -it --cpus="1.5" --memory="1g" ubuntu bash



```
[ec2-user@ip-172-31-90-216 ~]$ docker run --cpu-shares=1024 --memory="512m" -d  adcola13/my-static-website
e86eb099937ada3fc6628a3f60da223d436c048ccfcd364e611fcb2271238a08
[ec2-user@ip-172-31-90-216 ~]$ docker ps
CONTAINER ID   IMAGE                        COMMAND                  CREATED          STATUS          PORTS                                   NAMES
e86eb099937a   adcola13/my-static-website   "/docker-entrypoint.…"   12 seconds ago   Up 11 seconds   80/tcp                                  relaxed_mayer
09f90a8fdaca   adcola13/my-static-website   "/docker-entrypoint.…"   2 hours ago      Up 2 hours      0.0.0.0:8080->80/tcp, :::8080->80/tcp   magical_poincare
[ec2-user@ip-172-31-90-216 ~]$
```

Adding docker images to docker hub (PUSH)

a. Create a repository



Create repository

Repository Name *
demo-repository

Short description
Demo of Docker to create a static website using an AWS EC2 instance.

A short description to identify your repository. If the repository is public, this description is used to index your content on Docker Hub and in search engines, and is visible to users in search results.

Visibility
Using 0 of 1 private repositories. Get more

[✓ Public] Appears in Docker Hub search results
Private Only visible to you

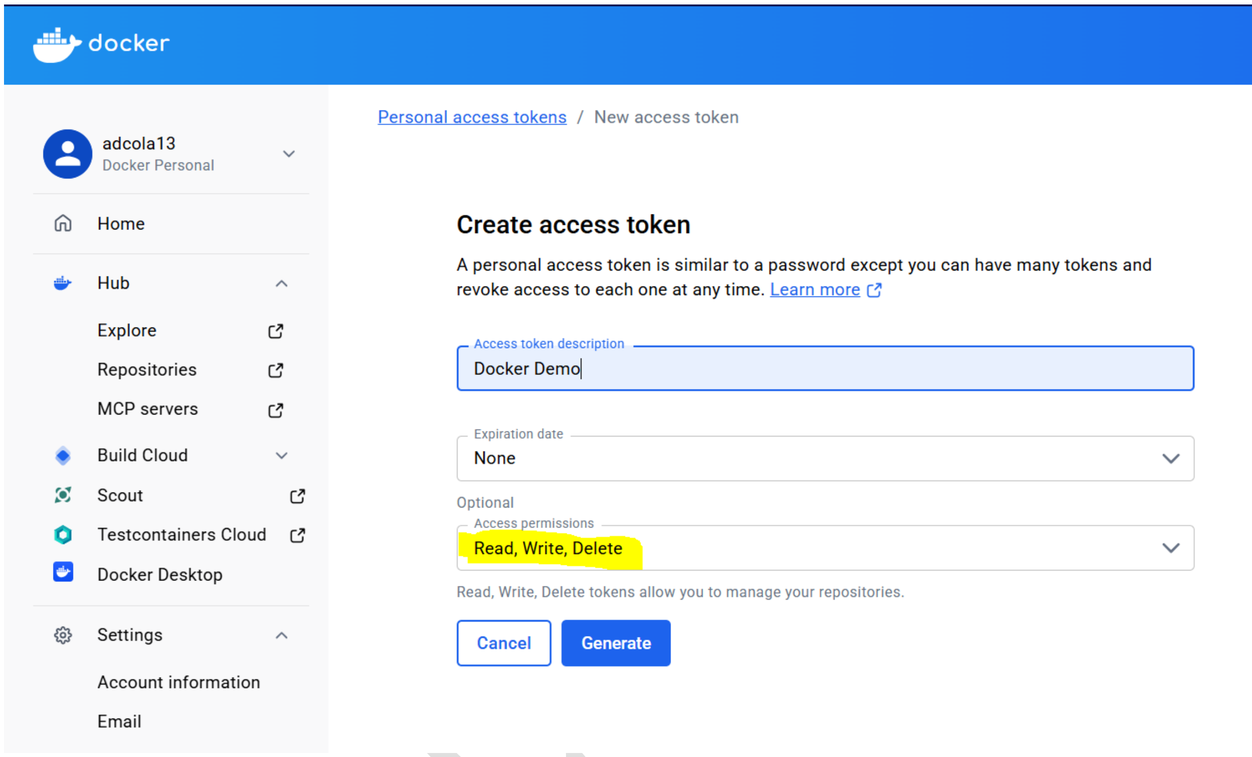
Pushing images

You can push a new image to this repository using the CLI:

```
docker tag local-image:tagname new-repo:tagnam
docker push new-repo:tagname
```

Make sure to replace `tagname` with your desired image repository tag.

b. Create an access token by going to Settings → Personal access tokens → Generate new token



Personal access tokens / New access token

Create access token

A personal access token is similar to a password except you can have many tokens and revoke access to each one at any time. Learn more

Access token description: Docker Demo

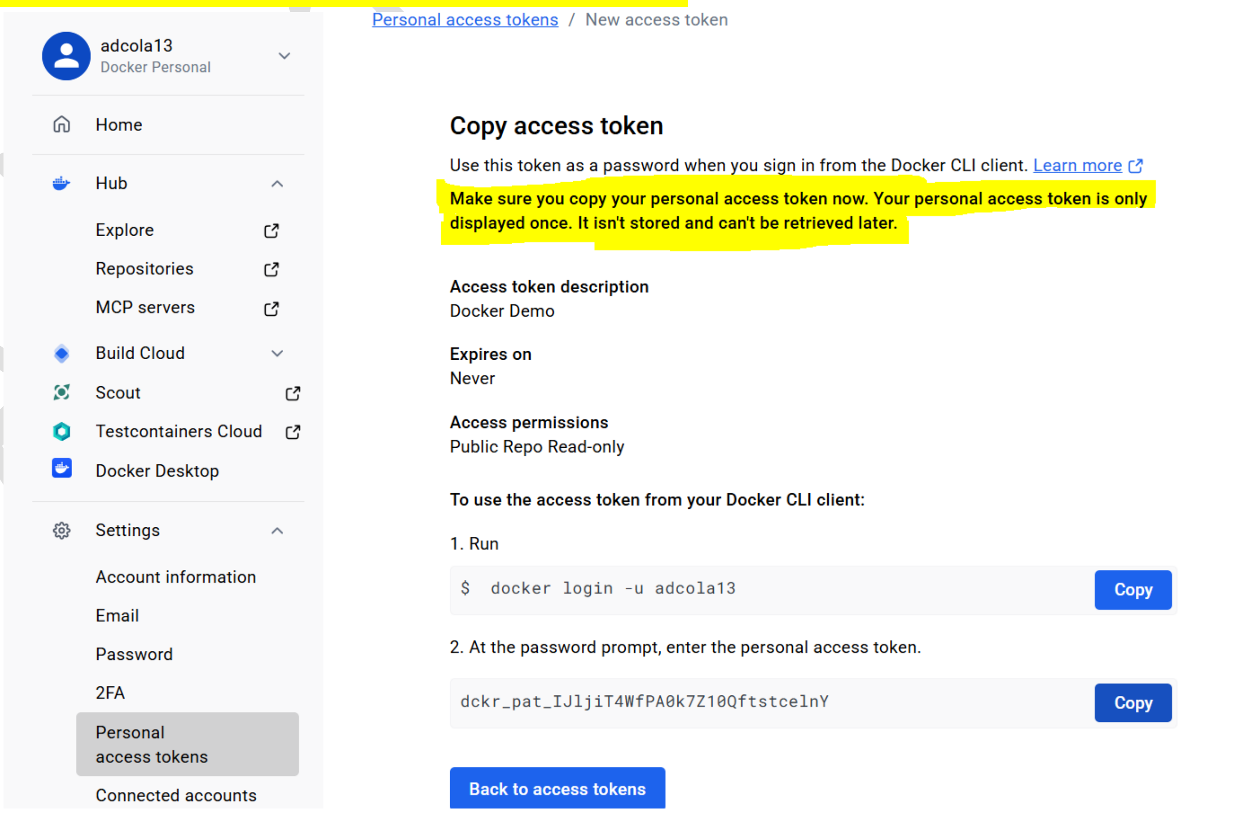
Expiration date: None

Optional

Access permissions: Read, Write, Delete

Read, Write, Delete tokens allow you to manage your repositories.

c. DON'T FORGET TO COPY DETAILS OF ACCESS TOKEN



Personal access tokens / New access token

Copy access token

Use this token as a password when you sign in from the Docker CLI client. Learn more

Make sure you copy your personal access token now. Your personal access token is only displayed once. It isn't stored and can't be retrieved later.

Access token description
Docker Demo

Expires on
Never

Access permissions
Public Repo Read-only

To use the access token from your Docker CLI client:

1. Run

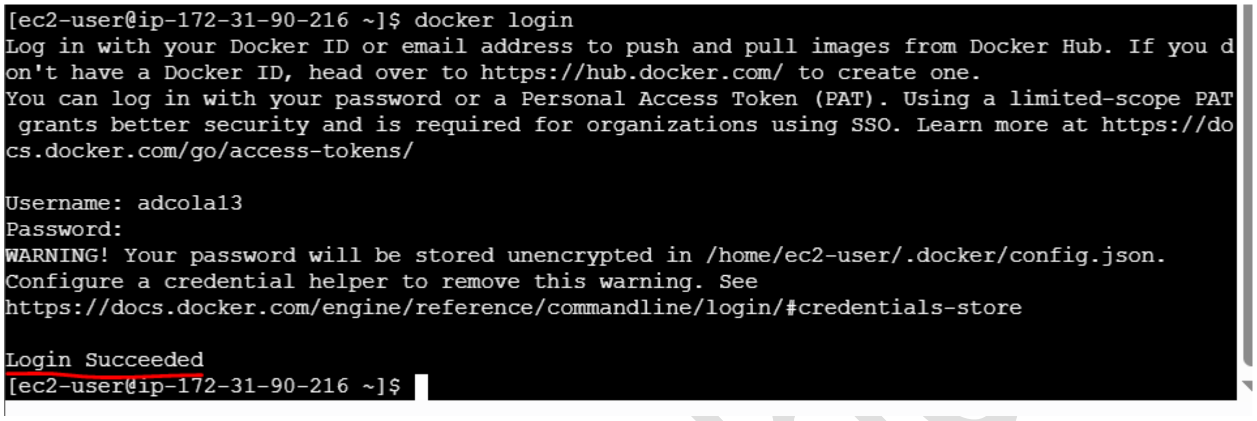
```
$  docker login -u adcola13
```

2. At the password prompt, enter the personal access token.

```
dckr_pat_IJljiT4WfPA0k7Z10QftstcelnY
```

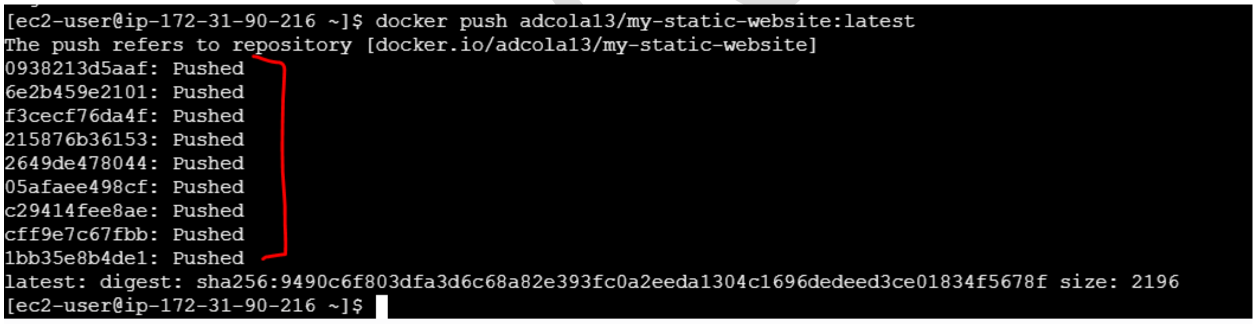
22. Use the below command to log into docker hub.

docker login



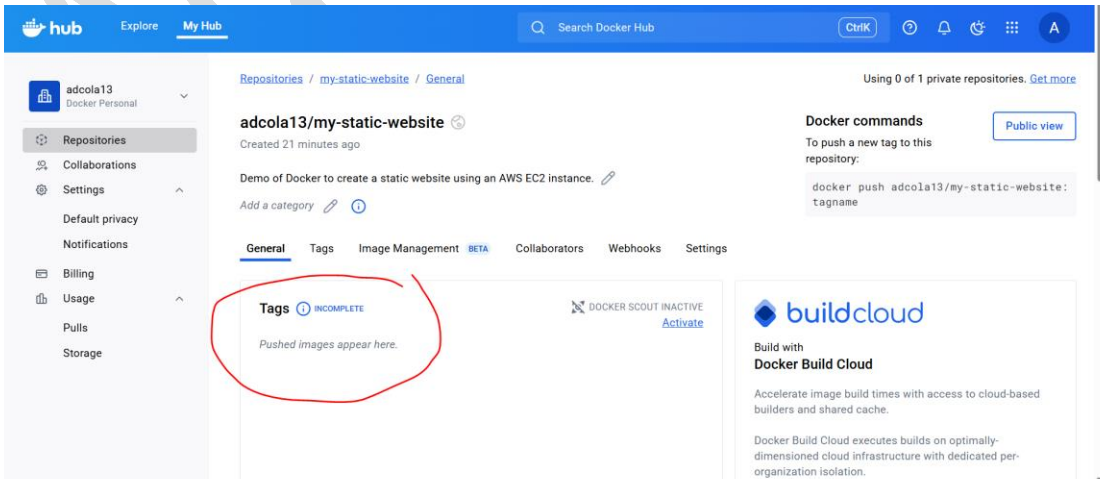
```
[ec2-user@ip-172-31-90-216 ~]$ docker login
Log in with your Docker ID or email address to push and pull images from Docker Hub. If you d
on't have a Docker ID, head over to https://hub.docker.com/ to create one.
You can log in with your password or a Personal Access Token (PAT). Using a limited-scope PAT
 grants better security and is required for organizations using SSO. Learn more at https://do
cs.docker.com/go/access-tokens/

Username: adcola13
Password:
WARNING! Your password will be stored unencrypted in /home/ec2-user/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[ec2-user@ip-172-31-90-216 ~]$
```

23. Use docker push to push image to docker hub and pull to pull image from dockerhub
24. Push the docker image into the docker hub using below command.
    docker push adcola13/my-static-website:latest



```
[ec2-user@ip-172-31-90-216 ~]$ docker push adcola13/my-static-website:latest
The push refers to repository [docker.io/adcola13/my-static-website]
0938213d5aaf: Pushed
6e2b459e2101: Pushed
f3cecf76da4f: Pushed
215876b36153: Pushed
2649de478044: Pushed
05afaee498cf: Pushed
c29414fee8ae: Pushed
cff9e7c67fbb: Pushed
1bb35e8b4de1: Pushed
latest: digest: sha256:9490c6f803dfa3d6c68a82e393fc0a2eeda1304c1696dedeed3ce01834f5678f size: 2196
[ec2-user@ip-172-31-90-216 ~]$
```

25. Look out for the pushed image in docker hub.



Repositories / my-static-website / General

Using 0 of 1 private repositories. Get more

adcola13/my-static-website

Created 21 minutes ago

Demo of Docker to create a static website using an AWS EC2 instance.

Add a category

Docker commands

Public view

To push a new tag to this repository:

```
docker push adcola13/my-static-website:tagname
```

General | Tags | Image Management BETA | Collaborators | Webhooks | Settings

Tags INCOMPLETE

Pushed images appear here.

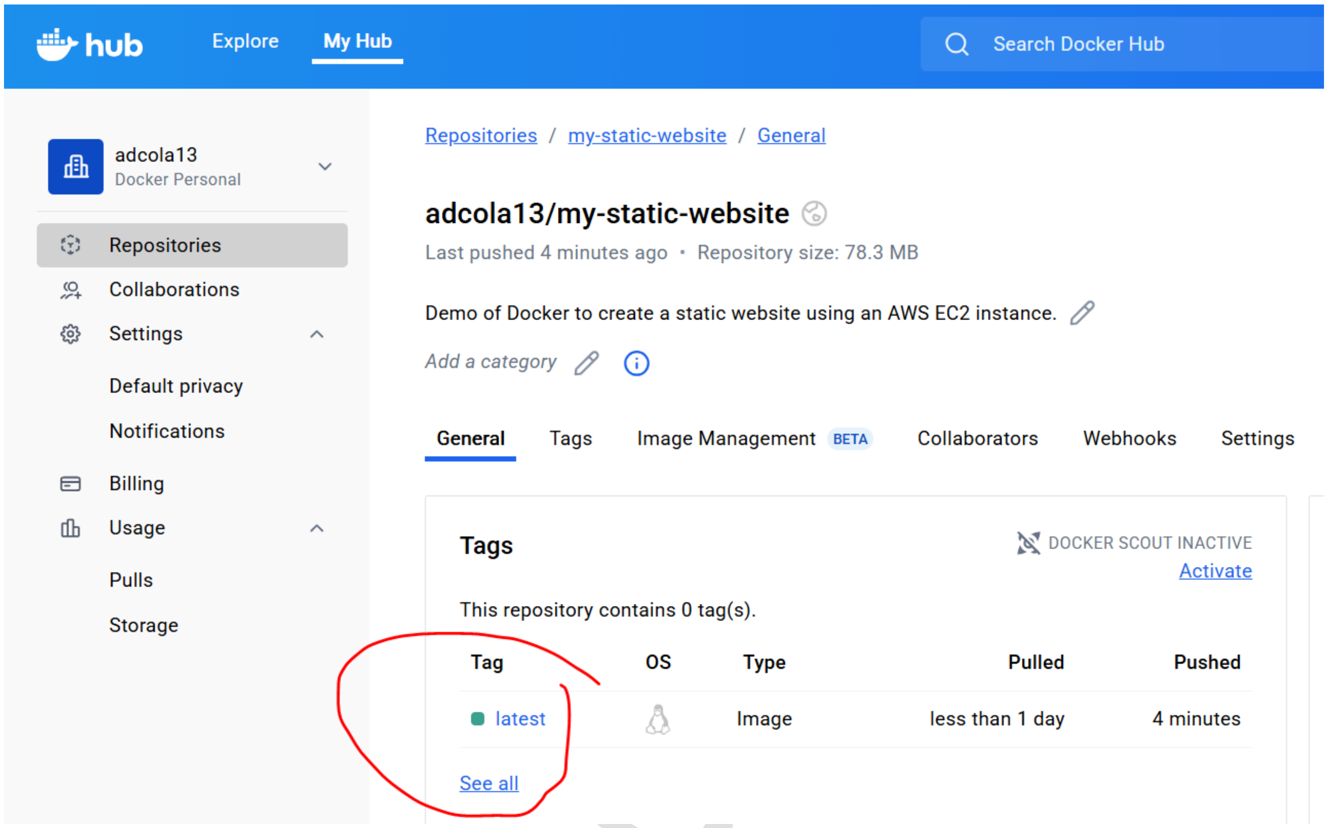
DOCKER SCOUT INACTIVE
Activate

buildcloud

Build with
Docker Build Cloud

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

Docker Build Cloud executes builds on optimally-dimensioned cloud infrastructure with dedicated per-organization isolation.



Repositories / my-static-website / General

adcola13/my-static-website

Last pushed 4 minutes ago • Repository size: 78.3 MB

Demo of Docker to create a static website using an AWS EC2 instance.

Add a category

General | Tags | Image Management BETA | Collaborators | Webhooks | Settings

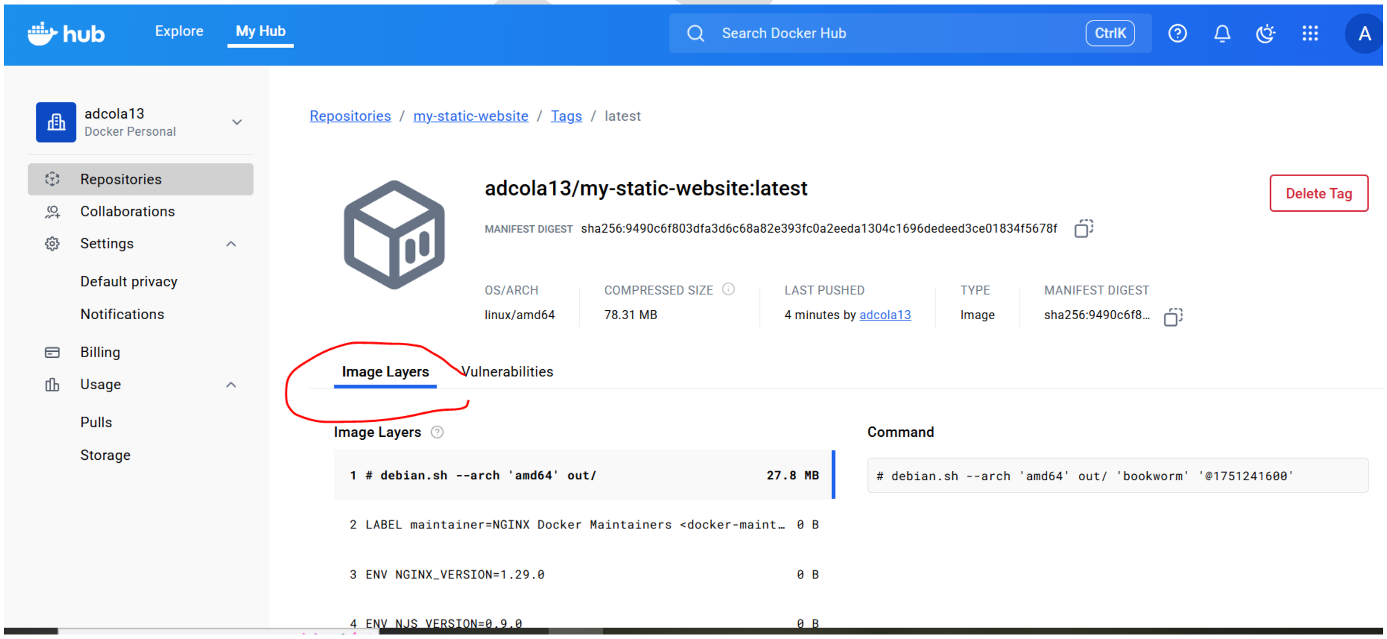
Tags

DOCKER SCOUT INACTIVE Activate

This repository contains 0 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	less than 1 day	4 minutes

See all



Repositories / my-static-website / Tags / latest

adcola13/my-static-website:latest

MANIFEST DIGEST sha256:9490c6f803dfa3d6c68a82e393fc0a2eeda1304c1696dedeed3ce01834f5678f

OS/ARCH	COMPRESSED SIZE	LAST PUSHED	TYPE	MANIFEST DIGEST
linux/amd64	78.31 MB	4 minutes by adcola13	Image	sha256:9490c6f8...

Image Layers | Vulnerabilities

Image Layers

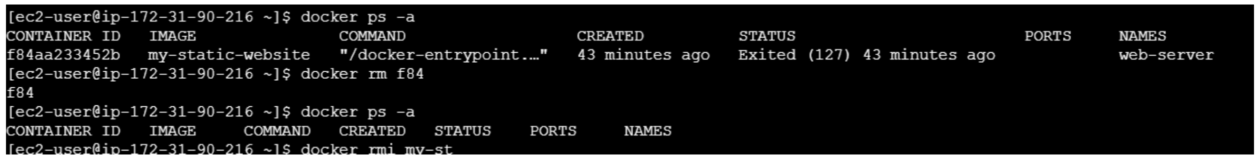
```
1 # debian.sh --arch 'amd64' out/                    27.8 MB
2 LABEL maintainer=NGINX Docker Maintainers <docker-maint…  0 B
3 ENV NGINX_VERSION=1.29.0                             0 B
4 ENV NJS_VERSION=0.9.0                                0 B
```

Command

```
# debian.sh --arch 'amd64' out/ 'bookworm' '@1751241600'
```

26. To remove a container, first stop and then remove.

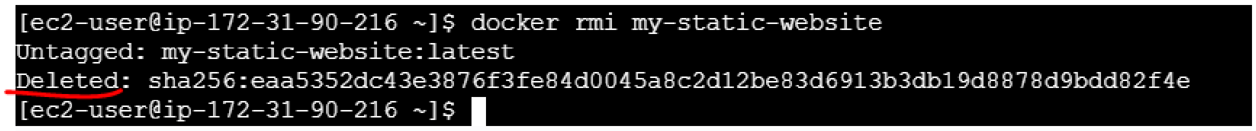
docker stop f84 (first 3 letters are enough)
docker rm f84



```
[ec2-user@ip-172-31-90-216 ~]$ docker ps -a
CONTAINER ID   IMAGE               COMMAND                  CREATED          STATUS                      PORTS     NAMES
f84aa233452b   my-static-website   "/docker-entrypoint.…"   43 minutes ago   Exited (127) 43 minutes ago           web-server
[ec2-user@ip-172-31-90-216 ~]$ docker rm f84
f84
[ec2-user@ip-172-31-90-216 ~]$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[ec2-user@ip-172-31-90-216 ~]$ docker rmi my-st
```

27. To remove image,

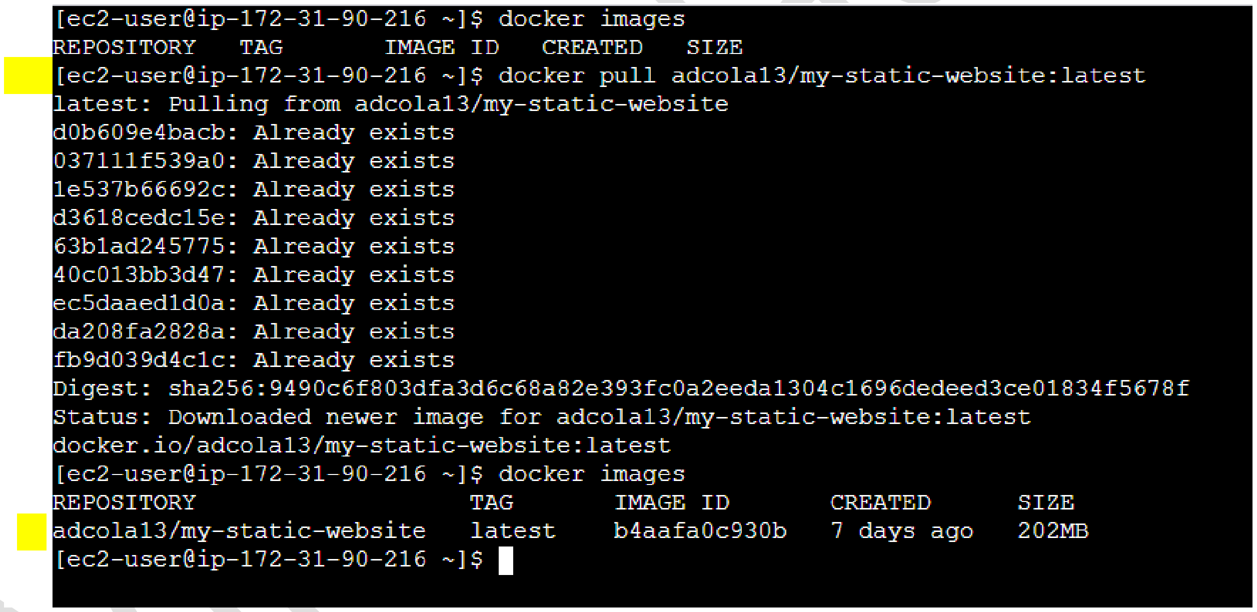
    docker rmi my-static-image



```
[ec2-user@ip-172-31-90-216 ~]$ docker rmi my-static-website
Untagged: my-static-website:latest
Deleted: sha256:eaa5352dc43e3876f3fe84d0045a8c2d12be83d6913b3db19d8878d9bdd82f4e
[ec2-user@ip-172-31-90-216 ~]$
```

Retrieving image from Docker Hub (PULL)

28. To get the image back from the docker hub, use pull command

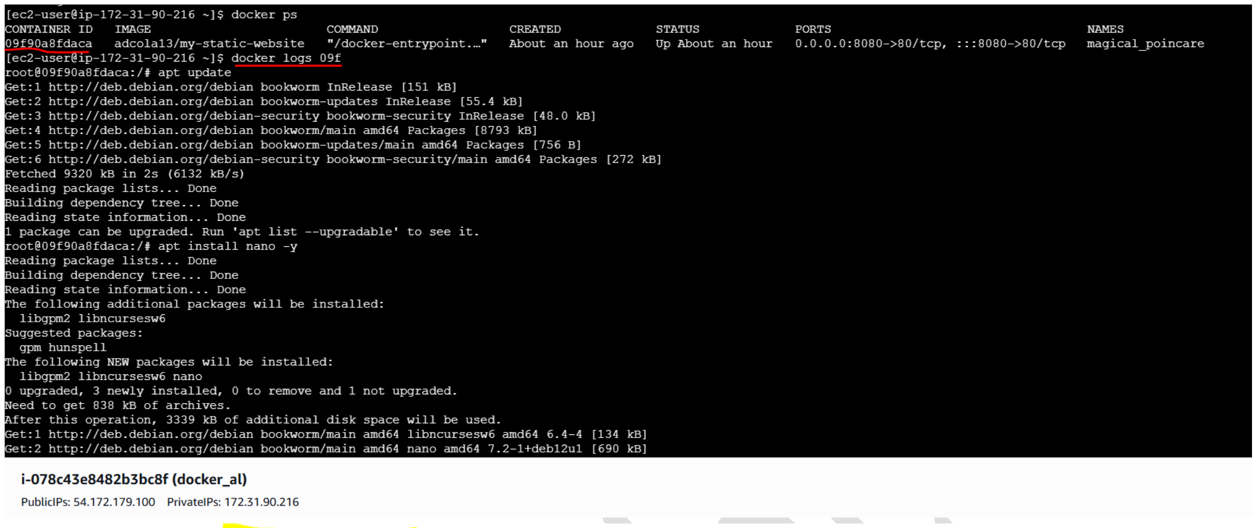


```
[ec2-user@ip-172-31-90-216 ~]$ docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
[ec2-user@ip-172-31-90-216 ~]$ docker pull adcola13/my-static-website:latest
latest: Pulling from adcola13/my-static-website
d0b609e4bacb: Already exists
037111f539a0: Already exists
1e537b66692c: Already exists
d3618cedc15e: Already exists
63b1ad245775: Already exists
40c013bb3d47: Already exists
ec5daaed1d0a: Already exists
da208fa2828a: Already exists
fb9d039d4c1c: Already exists
Digest: sha256:9490c6f803dfa3d6c68a82e393fc0a2eeda1304c1696dedeed3ce01834f5678f
Status: Downloaded newer image for adcola13/my-static-website:latest
docker.io/adcola13/my-static-website:latest
[ec2-user@ip-172-31-90-216 ~]$ docker images
REPOSITORY                   TAG       IMAGE ID       CREATED      SIZE
adcola13/my-static-website   latest    b4aafa0c930b   7 days ago   202MB
[ec2-user@ip-172-31-90-216 ~]$
```

Checking logs of the respective container

docker logs to see logs(concatenate head and tail commands to get fewer lines)

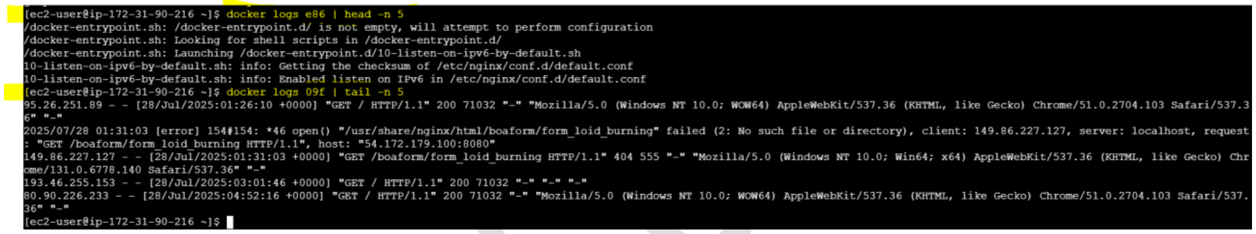
docker logs <Container ID or Name>
docker logs e86 | head -n 5
docker logs 09f | tail -n 6



```
[ec2-user@ip-172-31-90-216 ~]$ docker ps
CONTAINER ID   IMAGE                        COMMAND                  CREATED              STATUS             PORTS                                    NAMES
09f90a8fdaca   adcola13/my-static-website   "/docker-entrypoint.…"   About an hour ago    Up About an hour   0.0.0.0:8080->80/tcp, :::8080->80/tcp    magical_poincare
[ec2-user@ip-172-31-90-216 ~]$ docker logs 09f
root@09f90a8fdaca:/# apt update
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:3 http://deb.debian.org/debian-security bookworm-security InRelease [48.0 kB]
Get:4 http://deb.debian.org/debian bookworm/main amd64 Packages [8793 kB]
Get:5 http://deb.debian.org/debian bookworm-updates/main amd64 Packages [756 B]
Get:6 http://deb.debian.org/debian-security bookworm-security/main amd64 Packages [272 kB]
Fetched 9320 kB in 2s (6132 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
1 package can be upgraded. Run 'apt list --upgradable' to see it.
root@09f90a8fdaca:/# apt install nano -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libgpm2 libncursesw6
Suggested packages:
  gpm hunspell
The following NEW packages will be installed:
  libgpm2 libncursesw6 nano
0 upgraded, 3 newly installed, 0 to remove and 1 not upgraded.
Need to get 838 kB of archives.
After this operation, 3339 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 libncursesw6 amd64 6.4-4 [134 kB]
Get:2 http://deb.debian.org/debian bookworm/main amd64 nano amd64 7.2-1+deb12u1 [690 kB]
```

i-078c43e8482b3bc8f (docker_al)

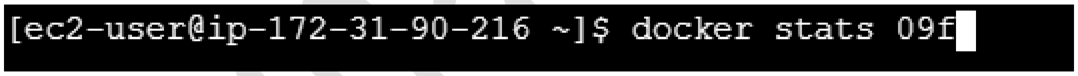
PublicIPs: 54.172.179.100   PrivateIPs: 172.31.90.216



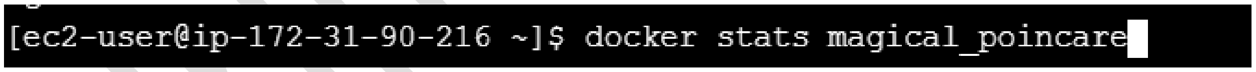
```
[ec2-user@ip-172-31-90-216 ~]$ docker logs e86 | head -n 5
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
[ec2-user@ip-172-31-90-216 ~]$ docker logs 09f | tail -n 5
95.26.251.89 - - [28/Jul/2025:01:26:10 +0000] "GET / HTTP/1.1" 200 71032 "-" "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.3
6" "-"
2025/07/28 01:31:03 [error] 154#154: *46 open() "/usr/share/nginx/html/boaform/form_loid_burning" failed (2: No such file or directory), client: 149.86.227.127, server: localhost, request
: "GET /boaform/form_loid_burning HTTP/1.1", host: "54.172.179.100:8080"
149.86.227.127 - - [28/Jul/2025:01:31:03 +0000] "GET /boaform/form_loid_burning HTTP/1.1" 404 555 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chr
ome/131.0.6778.140 Safari/537.36" "-"
193.46.255.153 - - [28/Jul/2025:03:01:46 +0000] "GET / HTTP/1.1" 200 71032 "-" "-" "-"
80.90.226.233 - - [28/Jul/2025:04:52:16 +0000] "GET / HTTP/1.1" 200 71032 "-" "Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.103 Safari/537.
36" "-"
[ec2-user@ip-172-31-90-216 ~]$
```

Check resources like CPU, RAM and disk space used by the container

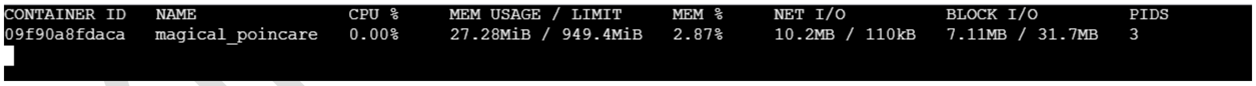
docker stats <container ID or name>



```
[ec2-user@ip-172-31-90-216 ~]$ docker stats 09f
```



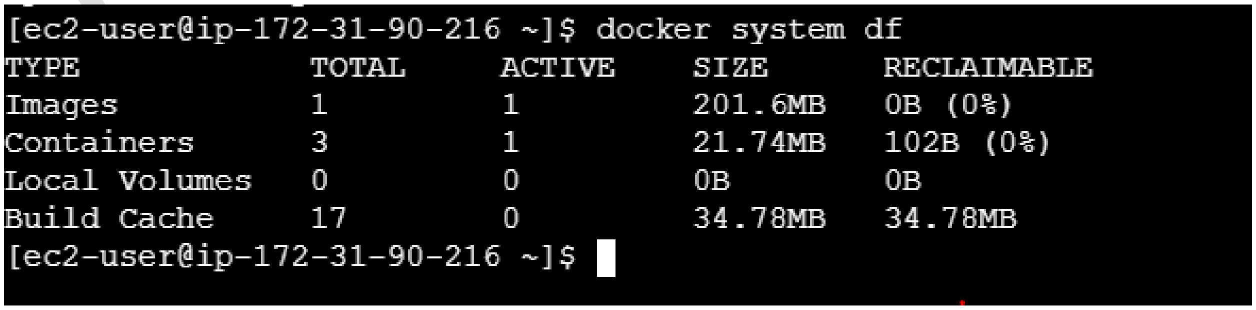
```
[ec2-user@ip-172-31-90-216 ~]$ docker stats magical_poincare
```



```
CONTAINER ID   NAME               CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O         PIDS
09f90a8fdaca   magical_poincare   0.00%     27.28MiB / 949.4MiB   2.87%     10.2MB / 110kB    7.11MB / 31.7MB   3
```

To display storage and disk space for all containers and images

docker system df

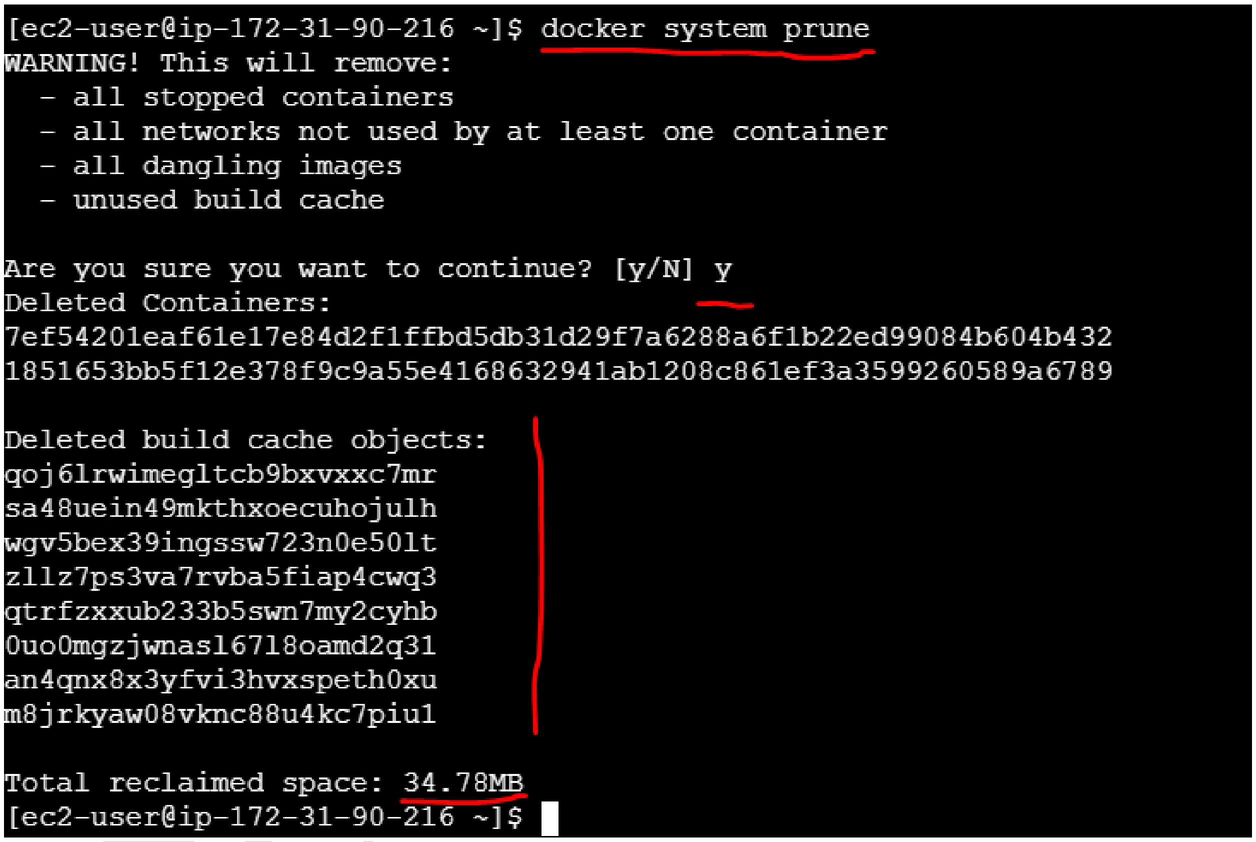


```
[ec2-user@ip-172-31-90-216 ~]$ docker system df
TYPE            TOTAL     ACTIVE    SIZE      RECLAIMABLE
Images          1         1         201.6MB   0B (0%)
Containers      3         1         21.74MB   102B (0%)
Local Volumes   0         0         0B        0B
Build Cache     17        0         34.78MB   34.78MB
[ec2-user@ip-172-31-90-216 ~]$
```

To remove images not associated with any container

Use Prune command

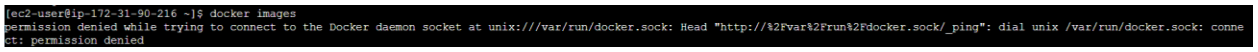
docker system prune



```
[ec2-user@ip-172-31-90-216 ~]$ docker system prune
WARNING! This will remove:
  - all stopped containers
  - all networks not used by at least one container
  - all dangling images
  - unused build cache

Are you sure you want to continue? [y/N] y
Deleted Containers:
7ef54201eaf61e17e84d2f1ffbd5db31d29f7a6288a6f1b22ed99084b604b432
1851653bb5f12e378f9c9a55e4168632941ab1208c861ef3a3599260589a6789

Deleted build cache objects:
qoj6lrwimegltcb9bxvxxc7mr
sa48uein49mkthxoecuhojulh
wgv5bex39ingssw723n0e50lt
zllz7ps3va7rvba5fiap4cwq3
qtrfzxxub233b5swn7my2cyhb
0uo0mgzjwnasl6718oamd2q31
an4qnx8x3yfvi3hvxspeth0xu
m8jrkyaw08vknc88u4kc7piu1

Total reclaimed space: 34.78MB
[ec2-user@ip-172-31-90-216 ~]$
```

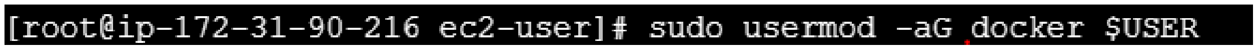
Troubleshooting Common Errors

1. To get permission to see images or containers, use SUDO or add the user to the Docker user group.



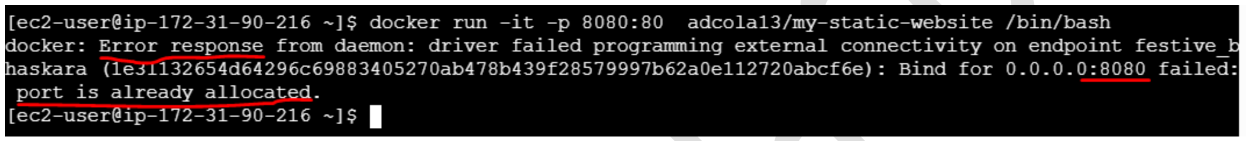
```
[ec2-user@ip-172-31-90-216 ~]$ docker images
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping": dial unix /var/run/docker.sock: conne
ct: permission denied
```

sudo usermod -aG docker $USER



```
[root@ip-172-31-90-216 ec2-user]# sudo usermod -aG docker $USER
```

2. Make sure that the DockerFile syntax is correct, especially the quotes(")
3. Port 8080 is already allocated.

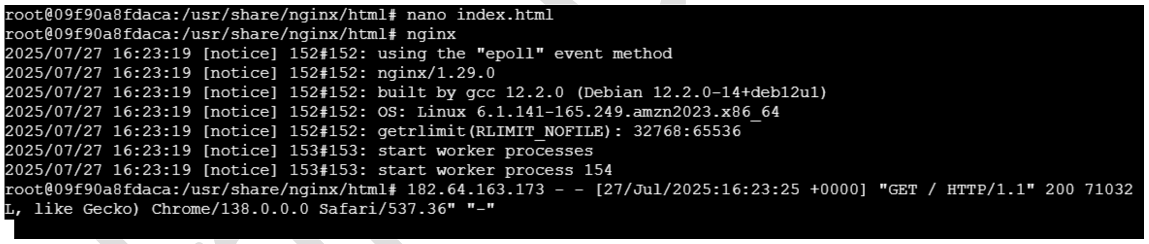


```
[ec2-user@ip-172-31-90-216 ~]$ docker run -it -p 8080:80  adcola13/my-static-website /bin/bash
docker: Error response from daemon: driver failed programming external connectivity on endpoint festive_b
haskara (1e31132654d64296c69883405270ab478b439f28579997b62a0e112720abcf6e): Bind for 0.0.0.0:8080 failed:
 port is already allocated.
[ec2-user@ip-172-31-90-216 ~]$
```

Solution: Stop the older container that is mapped to 8080.

Docker stop <container ID or Name>

4. While running the container in "Interaction" mode, you may need to install nginx if the website is still not working after completing all the required steps.



```
root@09f90a8fdaca:/usr/share/nginx/html# nano index.html
root@09f90a8fdaca:/usr/share/nginx/html# nginx
2025/07/27 16:23:19 [notice] 152#152: using the "epoll" event method
2025/07/27 16:23:19 [notice] 152#152: nginx/1.29.0
2025/07/27 16:23:19 [notice] 152#152: built by gcc 12.2.0 (Debian 12.2.0-14+deb12u1)
2025/07/27 16:23:19 [notice] 152#152: OS: Linux 6.1.141-165.249.amzn2023.x86_64
2025/07/27 16:23:19 [notice] 152#152: getrlimit(RLIMIT_NOFILE): 32768:65536
2025/07/27 16:23:19 [notice] 153#153: start worker processes
2025/07/27 16:23:19 [notice] 153#153: start worker process 154
root@09f90a8fdaca:/usr/share/nginx/html# 182.64.163.173 - - [27/Jul/2025:16:23:25 +0000] "GET / HTTP/1.1" 200 71032
L, like Gecko) Chrome/138.0.0.0 Safari/537.36" "-"
```

5. To change a project file while inside the container using the BASH shell.

Solution: Use the nano command.

If nano is not present, then install it inside the BASH of the container.

apt update

apt install nano -y