

# Web Development Advanced Concepts: Docker Primer

# Virtualization

## Using containers with Docker

Just Overview,  
we go step-by-step

- ❑ Download and install Docker

- ❑ Pull the latest version of the nginx image from Docker Hub:

```
docker pull nginx:latest
```

- ❑ Run a container and map port 80 from the container to port 8080 on your host:

```
docker run -d -p 8080:80 --name my-nginx nginx:latest
```

- ❑ -d runs the container in detached mode (in the background).

- ❑ -p 8080:80 maps port 80 inside the container to port 8080 on your host.

- ❑ --name my-nginx assigns a name to the container for easier management.

- ❑ nginx:latest specifies the image and tag to use.

- ❑ Check that it runs:                    `docker ps / docker ps -a`

- ❑ Open browser:                        `http://localhost:8080`

- ❑ View logs:                            `docker logs my-nginx`

- ❑ Access container:                  `docker exec -it my-nginx /bin/bash`

- ❑ Stop container:                    `docker stop my-nginx`

- ❑ Remove container:                `docker rm my-nginx`

# Virtualization

## Using containers with Docker

- Download and install Docker and macOS and Windows

The screenshot shows a web browser window with the Docker Desktop homepage. The address bar displays the URL <https://www.docker.com/products/docker-desktop/>. The page features a dark header with the Docker logo and navigation links for Products, Developers, Pricing, Support, Blog, and Company. A prominent call-to-action button at the bottom reads "Download Docker Desktop". The main content area includes a large Docker logo icon and a bold headline: "Increase productivity and efficiency to reduce time to deployment". Below the headline, a descriptive paragraph explains Docker Desktop's purpose: "Docker Desktop enhances your development experience by offering a powerful, user-friendly platform for container management. Fully integrated with your development tools, it simplifies container deployment and accelerates your workflow efficiency."

# Virtualization

## Using containers with Docker

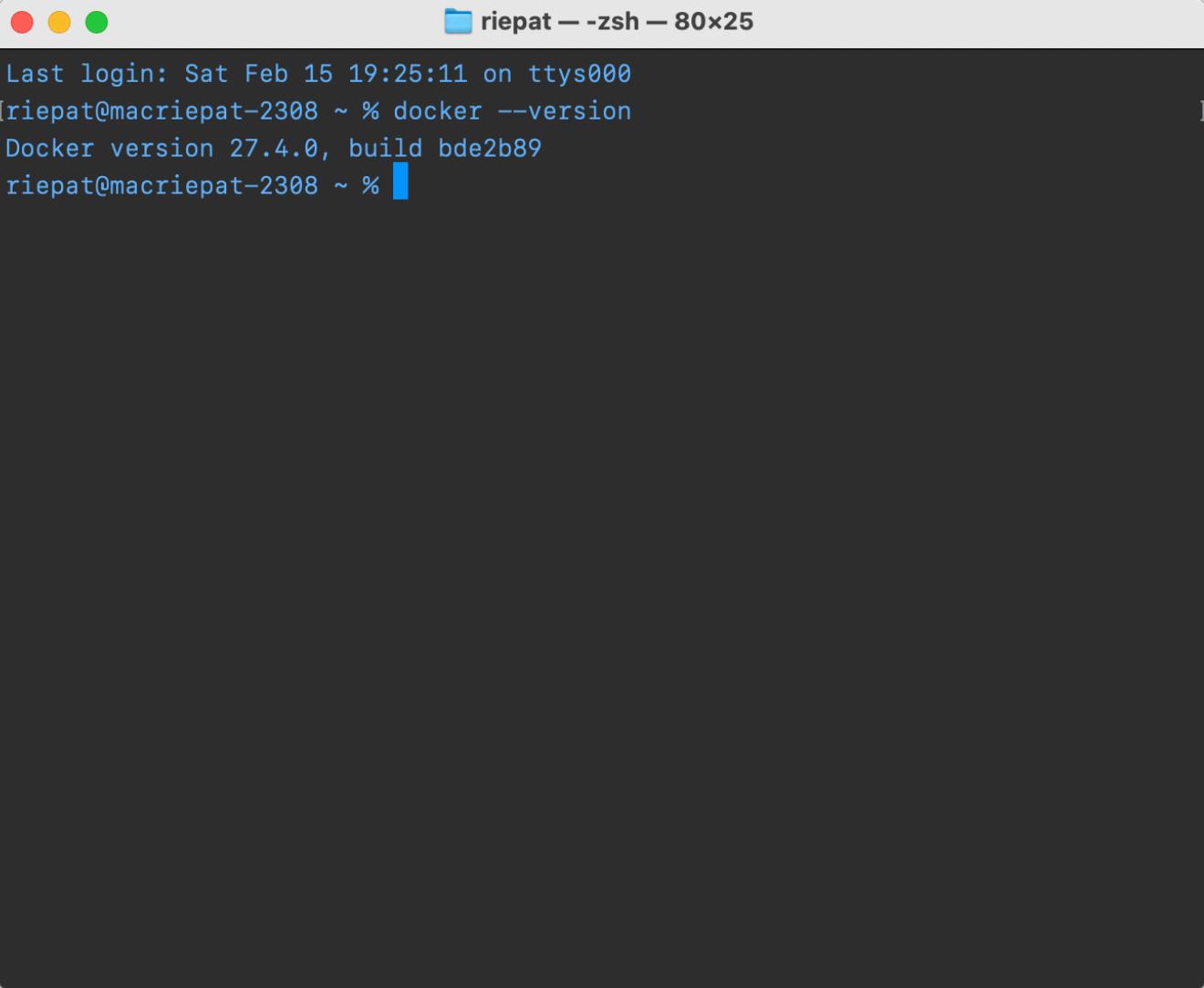
- Install Docker in Linux

```
sudo apt-get update  
sudo apt-get install -y docker.io  
sudo systemctl start docker  
sudo systemctl enable docker
```

# Virtualization

## Using containers with Docker

Check installed version



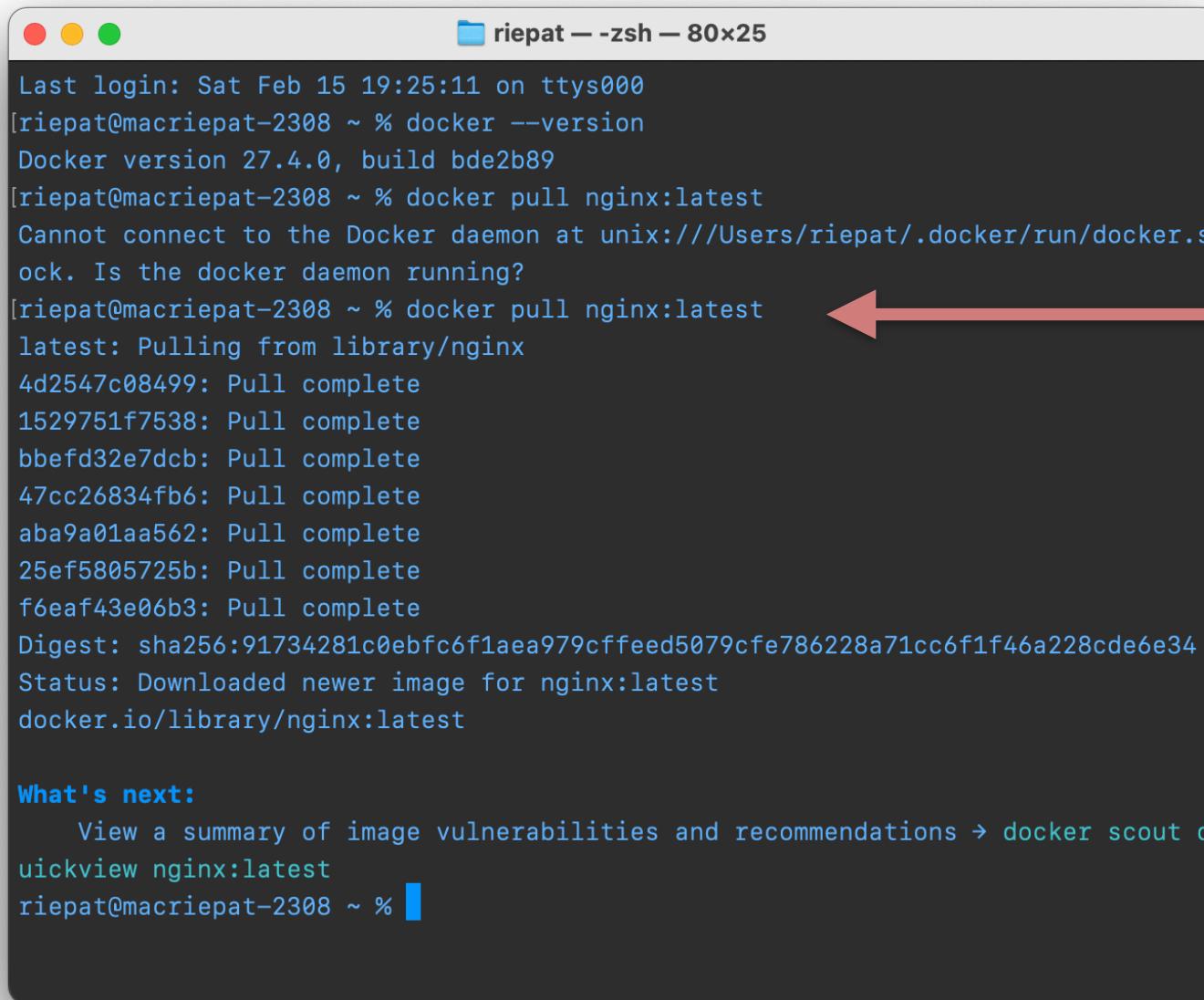
```
Last login: Sat Feb 15 19:25:11 on ttys000
[riepat@macriepat-2308 ~ % docker --version
Docker version 27.4.0, build bde2b89
riepat@macriepat-2308 ~ % ]
```

docker pull nginx:latest

# Virtualization

## Using containers with Docker

### ☐ Pull an image



```
Last login: Sat Feb 15 19:25:11 on ttys000
[riepat@macriepat-2308 ~ % docker --version
Docker version 27.4.0, build bde2b89
[riepat@macriepat-2308 ~ % docker pull nginx:latest
Cannot connect to the Docker daemon at unix:///Users/riepat/.docker/run/docker.s
ock. Is the docker daemon running?
[riepat@macriepat-2308 ~ % docker pull nginx:latest
latest: Pulling from library/nginx
4d2547c08499: Pull complete
1529751f7538: Pull complete
bbefd32e7dcb: Pull complete
47cc26834fb6: Pull complete
aba9a01aa562: Pull complete
25ef5805725b: Pull complete
f6eaf43e06b3: Pull complete
Digest: sha256:91734281c0ebfc6f1aea979cffeed5079cf786228a71cc6f1f46a228cde6e34
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest

What's next:
  View a summary of image vulnerabilities and recommendations → docker scout q
uickview nginx:latest
riepat@macriepat-2308 ~ %
```

docker pull nginx:latest

# Virtualization

## Using containers with Docker

☐ Create and run container from image

```
Last login: Sat Feb 15 19:25:11 on ttys000
[riepat@macriepat-2308 ~ % docker --version
Docker version 27.4.0, build bde2b89
[riepat@macriepat-2308 ~ % docker pull nginx:latest
Cannot connect to the Docker daemon at unix:///Users/riepat/.docker/run/docker.s
ock. Is the docker daemon running?
[riepat@macriepat-2308 ~ % docker pull nginx:latest
latest: Pulling from library/nginx
4d2547c08499: Pull complete
1529751f7538: Pull complete
bbefd32e7dcb: Pull complete
47cc26834fb6: Pull complete
aba9a01aa562: Pull complete
25ef5805725b: Pull complete
f6eaf43e06b3: Pull complete
Digest: sha256:91734281c0ebfc6f1aea979cffeed5079cf786228a71cc6f1f46a228cde6e34
Status: Downloaded newer image for nginx:latest
docker.io/library/nginx:latest

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout q
    uickview nginx:latest
[riepat@macriepat-2308 ~ % docker run -d -p 8080:80 --name my-nginx nginx:latest ] ←
efbc33fdcdf43671a8ee8172cc691feebd536b9b693f3374e2d58889b12ad5b0
riepat@macriepat-2308 ~ %
```

```
docker run -d -p 8080:80 --name my-nginx nginx:latest
```

# Virtualization

Optional

## Using containers with Docker

```
[riepat@macrie pat-2308 ~ % docker ps
CONTAINER ID   IMAGE      COMMAND      CREATED      STATUS      PORTS      NAMES
0533425d7c6c   nginx:latest "/docker-entrypoint..."   About a minute ago   Up About a minute   0.0.0.0:8080->80/tcp   my-nginx
riepat@macrie pat-2308 ~ %
```

docker ps

# Virtualization

## Using containers with Docker

☐ Alternative check in Docker Desktop

The screenshot shows the Docker Desktop application window. The left sidebar has a 'Containers' tab selected, followed by 'Images', 'Volumes', 'Builds', 'Docker Hub', 'Docker Scout', and 'Extensions'. A red arrow points to the 'my-nginx' container in the main list.

	Name	Container ID	Image	Port(s)	Actions
<input type="checkbox"/>	custom-nginx-cc	8221e5e56d5d	custom-nginx:latest	8080:80	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input type="checkbox"/>	pgsc2	2ddc29ffe27d	pgs:latest	5432:5432	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input type="checkbox"/>	pgsc1	a28a5d14225a	pgs:latest	5432:5432	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input checked="" type="checkbox"/>	my-nginx	0533425d7c6c	nginx:latest	8080:80 ↗	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>

Showing 4 items

**Walkthroughs**

- Multi-container applications  
8 mins
- Containerize your application  
3 mins

[View more in the Learning center](#)

Engine running | RAM 0.59 GB CPU 0.10% Disk: 4.51 GB used (limit 58.37 GB) > [New version available](#)

# Virtualization

Using containers with Docker

Open browser

The screenshot shows a Mac OS X desktop environment. In the top right corner, there is a window titled "riepat — zsh — 80x25" containing a command-line interface. In the top left corner, there is a window titled "Welcome to nginx!" which displays the "Welcome to nginx!" page. A red arrow points from the URL bar of the browser window to the terminal window, indicating the connection between the two.

lock Is the docker daemon running?

Welcome to nginx!

localhost:8080

# Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](http://nginx.org).  
Commercial support is available at [nginx.com](http://nginx.com).

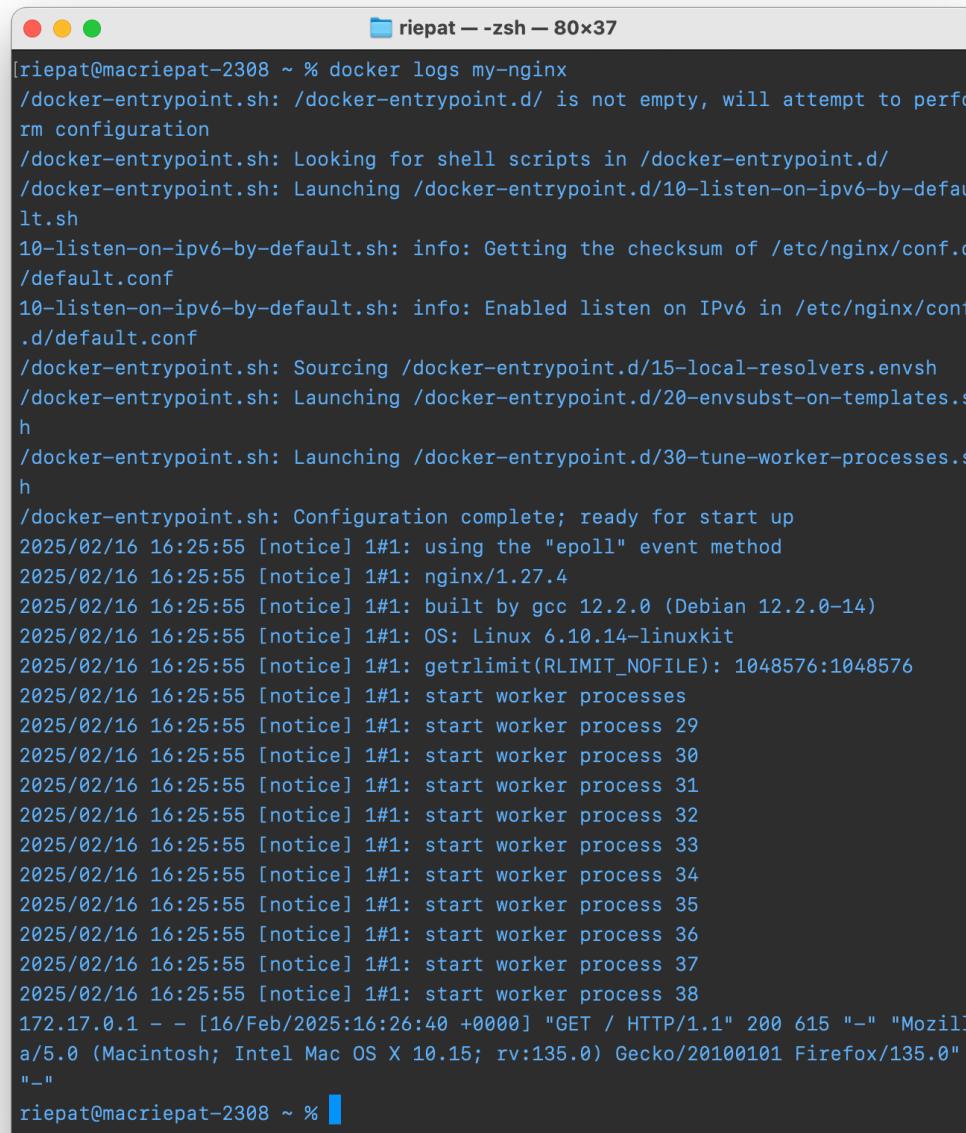
*Thank you for using nginx.*

```
[riepat@macriepat-2308 ~ % docker run -d -p 8080:80 --name my-nginx nginx:latest ]  
efbc33fdcdf43671a8ee8172cc691feebd536b9b693f3374e2d58889b12ad5b0  
[riepat@macriepat-2308 ~ % docker ps ]  
CONTAINER ID IMAGE COMMAND CREATED STATUS  
PORTS NAMES  
efbc33fdcdf4 nginx:latest "/docker-entrypoint..." 3 minutes ago Up 3 minu  
tes 0.0.0.0:8080->80/tcp my-nginx  
riepat@macriepat-2308 ~ %
```

# Virtualization

## Using containers with Docker

View logs



The screenshot shows a terminal window titled "riepat — zsh — 80x37". The window displays the output of the command "docker logs my-nginx". The log output is as follows:

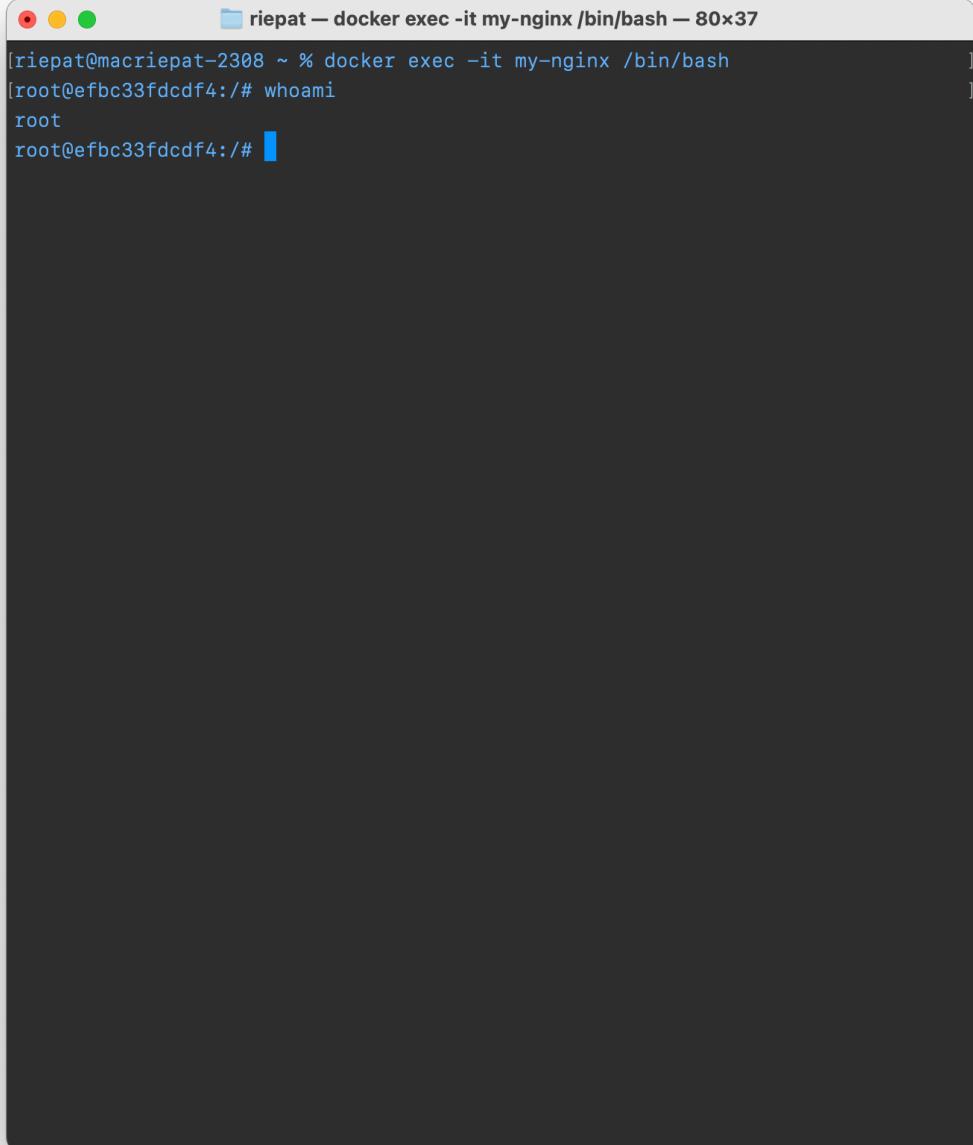
```
[riepat@macriepat-2308 ~ % docker logs my-nginx
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perfo
rm configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-defau
lt.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
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.s
h
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.s
h
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/02/16 16:25:55 [notice] 1#1: using the "epoll" event method
2025/02/16 16:25:55 [notice] 1#1: nginx/1.27.4
2025/02/16 16:25:55 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2025/02/16 16:25:55 [notice] 1#1: OS: Linux 6.10.14-linuxkit
2025/02/16 16:25:55 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/02/16 16:25:55 [notice] 1#1: start worker processes
2025/02/16 16:25:55 [notice] 1#1: start worker process 29
2025/02/16 16:25:55 [notice] 1#1: start worker process 30
2025/02/16 16:25:55 [notice] 1#1: start worker process 31
2025/02/16 16:25:55 [notice] 1#1: start worker process 32
2025/02/16 16:25:55 [notice] 1#1: start worker process 33
2025/02/16 16:25:55 [notice] 1#1: start worker process 34
2025/02/16 16:25:55 [notice] 1#1: start worker process 35
2025/02/16 16:25:55 [notice] 1#1: start worker process 36
2025/02/16 16:25:55 [notice] 1#1: start worker process 37
2025/02/16 16:25:55 [notice] 1#1: start worker process 38
172.17.0.1 - - [16/Feb/2025:16:26:40 +0000] "GET / HTTP/1.1" 200 615 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:135.0) Gecko/20100101 Firefox/135.0"
"-"
riepat@macriepat-2308 ~ %
```

docker logs my-nginx

# Virtualization

## Using containers with Docker

Access container



The screenshot shows a macOS terminal window with a dark background. The title bar reads "riepat — docker exec -it my-nginx /bin/bash — 80x37". The terminal content is as follows:

```
[riepat@macriepat-2308 ~ % docker exec -it my-nginx /bin/bash
[root@efbc33fdcdf4:/# whoami
root
root@efbc33fdcdf4:/# ]]
```

At the bottom of the terminal window, there is a blue vertical scroll bar.

`docker exec -it my-nginx /bin/bash`

# Virtualization

## Using containers with Docker

Stop container:

```
[riepat@macriebat-2308 ~ % docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0533425d7c6c nginx:latest "/docker-entrypoint...." About a minute ago Up About a minute 0.0.0.0:8080->80/tcp my-nginx
[riepat@macriebat-2308 ~ % docker stop my-nginx
my-nginx
[riepat@macriebat-2308 ~ % docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
riepat@macriebat-2308 ~ %
```

docker stop my-nginx

# Virtualization

## Using containers with Docker

Show stopped container:

```
[riepat@macriepat-2308 ~ % docker ps -a
CONTAINER ID   IMAGE      COMMAND           CREATED        STATUS          PORTS     NAMES
0533425d7c6c   nginx:latest "/docker-entrypoint..." 4 minutes ago  Exited (0) 2 minutes ago
8221e5e56d5d   custom-nginx  "/docker-entrypoint..." 3 hours ago   Exited (0) 31 minutes ago
2ddc29ffe27d   pgs:latest   "docker-entrypoint.s..." 11 months ago  Exited (0) 6 weeks ago
a28a5d14225a   pgs:latest   "docker-entrypoint.s..." 11 months ago  Exited (0) 11 months ago
riepat@macriepat-2308 ~ %
]  ↗
```

docker stop my-nginx

# Virtualization

## Using containers with Docker

Show stopped container:

```
[riepat@macriebat-2308 ~ % docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0533425d7c6c nginx:latest "/docker-entrypoint...." 4 minutes ago Exited (0) 2 minutes ago my-nginx
8221e5e56d5d custom-nginx "/docker-entrypoint...." 3 hours ago Exited (0) 31 minutes ago custom-nginx-container
2ddc29ffe27d pgs:latest "docker-entrypoint.s..." 11 months ago Exited (0) 6 weeks ago pgsc2
a28a5d14225a pgs:latest "docker-entrypoint.s..." 11 months ago Exited (0) 11 months ago pgsc1
[riepat@macriebat-2308 ~ % docker rm my-nginx
my-nginx
[riepat@macriebat-2308 ~ % docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
8221e5e56d5d custom-nginx "/docker-entrypoint...." 3 hours ago Exited (0) 32 minutes ago custom-nginx-container
2ddc29ffe27d pgs:latest "docker-entrypoint.s..." 11 months ago Exited (0) 6 weeks ago pgsc2
a28a5d14225a pgs:latest "docker-entrypoint.s..." 11 months ago Exited (0) 11 months ago pgsc1
riepat@macriebat-2308 ~ % ]
```

docker rm my-nginx

# Virtualization

## Using containers with Docker

❑ Container is gone, but the pulled image is still there

```
riePAT@macriePAT-2308 ~ % docker rm my-nginx
my-nginx
[riePAT@macriePAT-2308 ~ % docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
8221e5e56d5d custom-nginx "/docker-entrypoint..." 3 hours ago Exited (0) 32 minutes ago custom-nginx-container
2ddc29ffe27d pgs:latest "docker-entrypoint.s..." 11 months ago Exited (0) 6 weeks ago pgsc2
a28a5d14225a pgs:latest "docker-entrypoint.s..." 11 months ago Exited (0) 11 months ago pgsc1
[riePAT@macriePAT-2308 ~ % docker image ls
REPOSITORY TAG IMAGE ID CREATED SIZE
custom-nginx latest 13db8e1d9f5a 3 hours ago 197MB
nginx 9b1b7be1ffa6 11 days ago 197MB
docker/desktop-kubernetes kubernetes-v1.30.5-cni-v1.4.0-critools-v1.29.0-cri-dockerd-v0.3.11-1-debian a7b8c38c701e 4 months ago 419MB
registry.k8s.io/kube-apiserver v1.30.5 2bf7f63bc5e4 5 months ago 112MB
registry.k8s.io/kube-controller-manager v1.30.5 e1be44cf89df 5 months ago 107MB
registry.k8s.io/kube-scheduler v1.30.5 b6db73bf7694 5 months ago 68.5MB
registry.k8s.io/kube-proxy v1.30.5 57f247cd1b56 5 months ago 87.9MB
registry.k8s.io/coredns/coredns v1.11.3 2f6c962e7b83 6 months ago 60.2MB
jkpgcity latest ecef03278336 11 months ago 140MB
stores-img2 latest 6e92c89749a5 11 months ago 140MB
stores-img latest d58b9f1af73d 11 months ago 140MB
registry.k8s.io/etcda 3.5.12-0 014faa467e29 12 months ago 139MB
alpine latest ace17d5d883e 12 months ago 7.73MB
docker/desktop-kubernetes kubernetes-v1.29.1-cni-v1.4.0-critools-v1.29.0-cri-dockerd-v0.3.8-1-debian e953cb83dd7e 13 months ago 422MB
registry.k8s.io/kube-apiserver v1.29.1 f3b81ff188c6 13 months ago 123MB
registry.k8s.io/kube-controller-manager v1.29.1 8715bb0e3bc2 13 months ago 118MB
registry.k8s.io/kube-proxy v1.29.1 b125ba1d1878 13 months ago 85.4MB
registry.k8s.io/kube-scheduler v1.29.1 140ecfd0789f 13 months ago 58MB
registry.k8s.io/etcda 3.5.10-0 79f8d13ae8b8 15 months ago 136MB
registry.k8s.io/coredns/coredns v1.11.1 2437cf762177 18 months ago 57.4MB
docker/desktop-vpnkit-controller dc331cb22850be0cdd97c84a9cfecaf44a1afb6e 3750dfec169f 21 months ago 35MB
registry.k8s.io/pause 3.9 829e9de338bd 2 years ago 514kB
pgs latest 7dbafaf65348 3 years ago 351MB
docker/desktop-storage-provisioner v2.0 c027a58fa0bb 3 years ago 39.8MB
riePAT@macriePAT-2308 ~ %
```

docker image ls

# Virtualization

## Using containers with Docker

Show stopped container:

The screenshot shows the Docker Desktop application interface. The left sidebar has 'Images' selected. The main area displays local images with 24 items listed. A red arrow points to the checkbox column header in the table.

<input type="checkbox"/>	Name	Tag	Image ID	Created	Size	Actions
<input type="checkbox"/>	registry.k8s	v1.29.1	140ecfd0789f	1 year ago	58.02 MB	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input type="checkbox"/>	registry.k8s	3.5.10-0	79f8d13ae8b8	1 year ago	136.46 MB	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input type="checkbox"/>	registry.k8s	v1.11.1	2437cf762177	2 years ago	57.38 MB	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input type="checkbox"/>	docker/des	dc331cb22	3750dfec169f	2 years ago	34.99 MB	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input type="checkbox"/>	registry.k8s	3.9	829e9de338bd	2 years ago	514 KB	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input type="checkbox"/>	pgs	latest	7dbafaf65348	3 years ago	350.75 MB	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input type="checkbox"/>	docker/des	v2.0	c027a58fa0bb	4 years ago	39.81 MB	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>
<input type="checkbox"/>	nginx	latest	9b1b7be1ffa6	12 days ago	197.28 MB	<a href="#">▶</a> <a href="#">⋮</a> <a href="#">trash</a>

Showing 24 items

Engine running | RAM 0.59 GB CPU 0.20% Disk: 4.51 GB used (limit 58.37 GB) | [New version available](#)

# Virtualization

## Using containers with Docker

- Container is gone, but the pulled image is still there

```
[riepat@macriepat-2308 ~ % docker rm my-nginx
my-nginx
[riepat@macriepat-2308 ~ % docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
8221e5e56d5d custom-nginx "/docker-entrypoint..." 3 hours ago Exited (0) 32 minutes ago custom-nginx-container
2ddc29ffe27d pgs:latest "docker-entrypoint.s..." 11 months ago Exited (0) 6 weeks ago pgsc2
a28a5d14225a pgs:latest "docker-entrypoint..." 11 months ago Exited (0) 11 months ago pgsc1
[riepat@macriepat-2308 ~ % docker image ls
REPOSITORY TAG IMAGE ID CREATED SIZE
custom-nginx latest 13db8e1d9f5a 3 hours ago 197MB
nginx          9b1b7be1ffa6 11 days ago 197MB
docker/desktop-kubernetes kubernetes-v1.30.5-cni-v1.4.0-critools-v1.29.0-cri-dockerd-v0.3.11-1-debian a7b8c38c701e 4 months ago 419MB
registry.k8s.io/kube-apiserver v1.30.5 2bf63bc5e4 5 months ago 112MB
registry.k8s.io/kube-controller-manager v1.30.5 e1be44cf89df 5 months ago 107MB
registry.k8s.io/kube-scheduler v1.30.5 b6db73bf7694 5 months ago 60.5MB
registry.k8s.io/kube-proxy v1.30.5 57f247cd1b56 5 months ago 87.9MB
registry.k8s.io/coredns/coredns v1.11.3 2f6c962e7b83 6 months ago 60.2MB
jkpgcity      latest ecef03278336 11 months ago 140MB
stores-img2    latest 6e92c89749a5 11 months ago 140MB
stores-img     latest d58b9f1af73d 11 months ago 140MB
registry.k8s.io/etcfd 3.5.12-0 014faa467e29 12 months ago 139MB
alpine         latest ace17d5d883e 12 months ago 7.73MB
docker/desktop-kubernetes kubernetes-v1.29.1-cni-v1.4.0-critools-v1.29.0-cri-dockerd-v0.3.8-1-debian e953cb83dd7e 13 months ago 422MB
registry.k8s.io/kube-apiserver v1.29.1 f3b81ff188c6 13 months ago 123MB
registry.k8s.io/kube-controller-manager v1.29.1 8715bb0e3bc2 13 months ago 118MB
registry.k8s.io/kube-proxy v1.29.1 b125ba1d1878 13 months ago 85.4MB
registry.k8s.io/kube-scheduler v1.29.1 140ecfd0789f 13 months ago 58MB
registry.k8s.io/etcfd 3.5.10-0 79f8d13ae8b8 15 months ago 136MB
registry.k8s.io/coredns/coredns v1.11.1 2437cf762177 18 months ago 57.4MB
docker/desktop-vpnkit-controller dc331cb22850be0cdd97c84a9cfecaf44a1afb6e 3750dfec169f 21 months ago 35MB
registry.k8s.io/pause        3.9   829e9de338bd 2 years ago 514kB
pgs           latest 7dbafaf65348 3 years ago 351MB
docker/desktop-storage-provisioner v2.0   c027a58fa0bb 3 years ago 39.8MB
[riepat@macriepat-2308 ~ %
```

docker image ls

- Image can be deleted by

docker image rm nginx

# Virtualization

## Container setup with Dockerfile

```
# Dockerfile
# Use the official Nginx image as the base image
FROM nginx:latest

# Copy the index.html file into Nginx's default directory
COPY index.html /usr/share/nginx/html/index.html

# Expose port 80 for the web server
EXPOSE 80
```

```
<!-- index.html -->
<!DOCTYPE html>
<html>
  <head>
    <title>My Nginx Site</title>
  </head>
  <body>
    <h1>Hello, Nginx from Docker!</h1>
    <p>This page is served by a custom Docker image.</p>
  </body>
</html>
```

Just Overview,  
we go step-by-step

# Virtualization

## Container setup with Dockerfile

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

# Copy the index.html file to the /usr/share/nginx/html directory
COPY index.html /usr/share/nginx/html

# Expose port 80
EXPOSE 80
```

```
<!-- index.html -->
<!DOCTYPE html>
<html>
  <head>
    <title>The Dockerfile Way</title>
  </head>
  <body>
    <h1>Hello World!</h1>
    <p>This page is generated by a Docker container.</p>
  </body>
</html>
```

docker-test — zsh — 80x24

# Virtualization

## Container setup with Dockerfile

```
# Dockerfile
# Use the official Nginx image as the base image
FROM nginx:latest

# Copy the index.html file into Nginx's default directory
COPY index.html /usr/share/nginx/html/index.html

# Expose port 80 for the web server
EXPOSE 80
```

```
<!-- index.html -->
<!DOCTYPE html>
<html>
  <head>
    <title>My Nginx Site</title>
  </head>
  <body>
    <h1>Hello, Nginx from Docker!</h1>
    <p>This page is served by a custom Docker image.</p>
  </body>
</html>
```

# Virtualization

## Container setup with Dockerfile

The screenshot shows the Visual Studio Code interface with the following elements:

- Explorer View:** Shows a folder named "DOCKER-TEST" containing "Dockerfile" and "index.html".
- Terminal Tab:** Contains the command "docker build -t custom-nginx .". A red arrow points from the text "Create the docker image" in the code editor to this command.
- Code Editor:** Displays the "index.html" file content, which includes an HTML page with a title "My Nginx Site" and a body containing "Hello, Nginx from Docker!" and "This page is served by a custom Docker image".
- Status Bar:** Shows file statistics ("0 △ 0") and system information ("Ln 1, Col 1 Spaces: 4 UTF-8 LF {} HTML").
- Bottom Status Bar:** Shows the URL "https://docs.docker.com/get-started/docker-concepts/building-images/writing-a-dockerfile/".

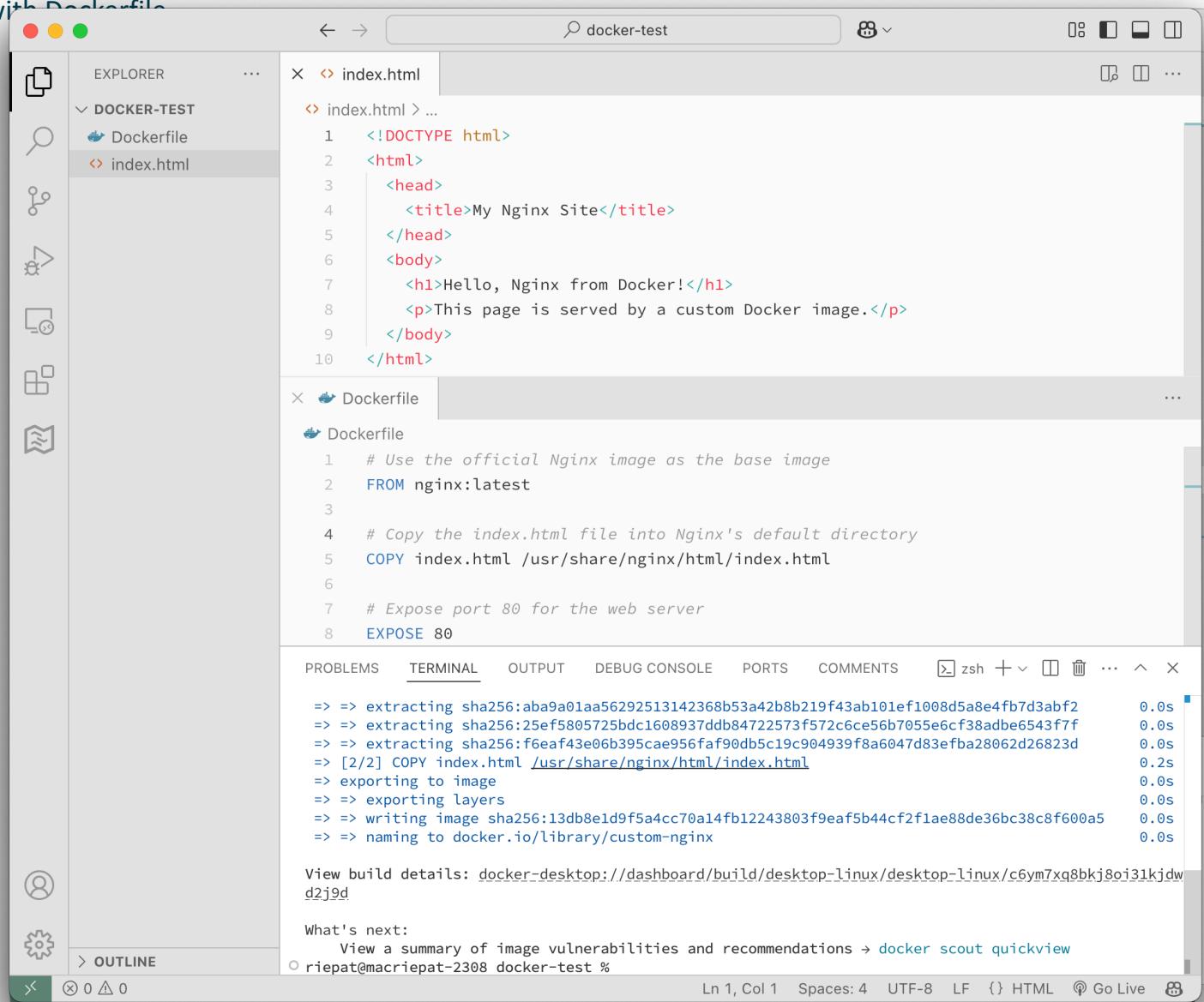
**Image name**

```
<!DOCTYPE html>
<html>
<head>
<title>My Nginx Site</title>
</head>
<body>
<h1>Hello, Nginx from Docker!</h1>
<p>This page is served by a custom Docker image.</p>
# In terminal
# 1. Create the docker image
docker build -t custom-nginx .

FROM nginx:latest
# Copy the index.html file into Nginx's default directory
COPY index.html /usr/share/nginx/html/index.html
# Expose port 80 for the web server
EXPOSE 80
```

# Virtualization

## Container setup with Dockerfile



The screenshot shows the Visual Studio Code interface with a Docker extension. The left sidebar has icons for Explorer, Search, Open, and Outline. The Explorer view shows a folder named 'DOCKER-TEST' containing 'Dockerfile' and 'index.html'. The 'index.html' file is open in the main editor, displaying an HTML document with a title 'My Nginx Site' and a body containing 'Hello, Nginx from Docker!' and 'This page is served by a custom Docker image.'. The 'Dockerfile' is also open, showing the following Dockerfile content:

```
# Use the official Nginx image as the base image
FROM nginx:latest

# Copy the index.html file into Nginx's default directory
COPY index.html /usr/share/nginx/html/index.html

# Expose port 80 for the web server
EXPOSE 80
```

The bottom status bar shows the command line: `riepat@macriepat-2308 docker-test %`. The terminal tab is active, showing the build logs:

```
=> => extracting sha256:aba9a01aa56292513142368b53a42b8b219f43ab101ef1008d5a8e4fb7d3abf2 0.0s
=> => extracting sha256:25ef5805725bcd1608937ddb84722573f572c6ce56b7055e6cf38adbe6543f7f 0.0s
=> => extracting sha256:f6eaf43e06b395cae956faf90db5c19c904939f8a6047d83efba28062d26823d 0.0s
=> [2/2] COPY index.html /usr/share/nginx/html/index.html 0.2s
=> exporting to image
=> exporting layers
=> => writing image sha256:13db8e1d9f5a4cc70a14fb12243803f9eaf5b44cf2f1ae88de36bc38c8f600a5 0.0s
=> => naming to docker.io/library/custom-nginx 0.0s
```

Below the logs, it says 'View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/c6ym7xq8bkj8oij31kjdw d2j9d'.

The status bar also shows the current position: 'Ln 1, Col 1'.

# Virtualization

## Container setup with Dockerfile

Host Container port      Container name      Image name

# 2. Create container and run it!

```
docker run -d -p 8080:80 --name custom-nginx-container custom-nginx
```

index.html

```
<!DOCTYPE html>
<html>
<head>
<title>My Nginx Site</title>
</head>
<body>
<h1>Hello Nginx from Docker!</h1>
<p>This page is served by a custom Docker image.</p>
</body>
</html>
```

```
4 # Copy the index.html file into Nginx's default directory
5 COPY index.html /usr/share/nginx/html/index.html
6
7 # Expose port 80 for the web server
8 EXPOSE 80
```

PROBLEMS TERMINAL OUTPUT DEBUG CONSOLE PORTS COMMENTS zsh + ×

```
=> [2/2] COPY index.html /usr/share/nginx/html/index.html
=> exporting to image
=> => exporting layers
=> => writing image sha256:13db8e1d9f5a4cc70a14fb12243803f9eaf5b44cf2f1ae88de36bc38c8f600a5
=> => naming to docker.io/library/custom-nginx
```

View build details: [docker\\_desktop://dashboard/build/desktop-linux/desktop-linux/c6ym7xq8bkj8oi31kjdw\\_d2j9d](#)

What's next:

- View a summary of image vulnerabilities and recommendations → [docker scout quickview](#)

riepat@macriepat-2308 docker-test % docker run -d -p 8080:80 --name custom-nginx-container custom-nginx

8221e5e56d5d56cd6aeb8fededa8a79ba8bd0f72dfa613bb755dad56e47b0092

riepat@macriepat-2308 docker-test %

Ln 1, Col 1 Spaces: 4 UTF-8 LF {} HTML ⚡ Go Live

# Virtualization

## Container setup with Dockerfile

The screenshot shows the Docker Desktop application running in a window. On the left is the Explorer sidebar with a 'DOCKER-TEST' folder containing 'Dockerfile' and 'index.html'. The main area displays the contents of 'index.html' and the Dockerfile.

**index.html:**

```
<!DOCTYPE html>
<html>
<head>
<title>My Nginx Site</title>
</head>
<body>
<h1>Hello, Nginx from Docker!</h1>
<p>This page is served by a custom Docker image.</p>
</body>
</html>
```

**Dockerfile:**

```
FROM nginx:latest
# Copy the index.html file into Nginx's default directory
COPY index.html /usr/share/nginx/html/index.html
# Expose port 80 for the web server
EXPOSE 80
```

The Dockerfile has been run, and the resulting container information is shown in the bottom right:

- View build details: [docker-desktop://dashboard/build/desktop-linux/desktop-linux/c6ym7xq8bkj8oi31kjdw2j9d](https://docker-desktop://dashboard/build/desktop-linux/desktop-linux/c6ym7xq8bkj8oi31kjdw2j9d)
- What's next:
  - View a summary of [linux/c6ym7xq8bkj8oi31kjdw2j9d](https://docker-desktop://dashboard/build/desktop-linux/desktop-linux/c6ym7xq8bkj8oi31kjdw2j9d) (cmd + click)
- Terminal output:
  - riepat@macriepat-2308 docker-test % docker run -d -p 8080:80 --name custom-nginx-container custom-nginx
  - riepat@macriepat-2308 docker-test % docker ps
- Table showing running containers:

COUNTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
NAMES					
8221e5e56d5d56cd6aeb8feda8a79ba8bd0f72dfda613bb755dad56e47b0092	custom-nginx	"/docker-entrypoint...."	41 seconds ago	Up 40 seconds	0.0.0.0:8080->80/tcp
riepat@macriepat-2308	custom-nginx-container				

# Virtualization

## Container setup with Dockerfile

The screenshot shows a Mac desktop environment with three main windows:

- Code Editor (VS Code):** The title bar says "docker-test". The Explorer sidebar shows a folder named "DOCKER-TEST" containing "Dockerfile" and "index.html". The "index.html" file is open in the editor, displaying the following content:

```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
<h1>Hello, Nginx from Docker!</h1>
<p>This page is served by a custom Docker image.</p>
</body>
</html>
```
- Browser:** The title bar says "My Nginx Site". The address bar shows "localhost:8080". The page content is identical to the one in the code editor.
- Terminal:** The title bar says "dockertest". The command entered was "docker run -d -p 8080:80 --name custom-nginx-container custom-nginx". The output shows the container has been created and is running, with port 8080 mapped to 80. A table lists the container's details:

COLUMN	IMAGE	COMMAND	CREATED	STATUS	PORTS
CONTAINER ID	custom-nginx	/docker-entrypoint....	41 seconds ago	Up 40 seconds	0.0.0.0:8080->80/tcp
NAMES	custom-nginx-container				

# Virtualization

## Container setup with Dockerfile

```
# In terminal

# Container can be removed with
docker rm container-name

# Check images
docker image ls

# Images can be removed with
docker image rm container-name
```

# Project

## Hint

Hint for your project: Dockerfile for Node needs more setup!

```
# Pull a image with node already integrated
FROM node:20-alpine

# Define a directory to use inside the container
WORKDIR /usr/src/app

# Copy the content of our working directory into
# the container's working directory
COPY . .

# Install necessary requirements
RUN npm install

# Expose port 3000 for the web server
EXPOSE 3000

# Define the command for starting our web application
# inside the container (often app.js)
CMD [ "node", "server.js"]
```

Just a hint,  
don't do it now!!