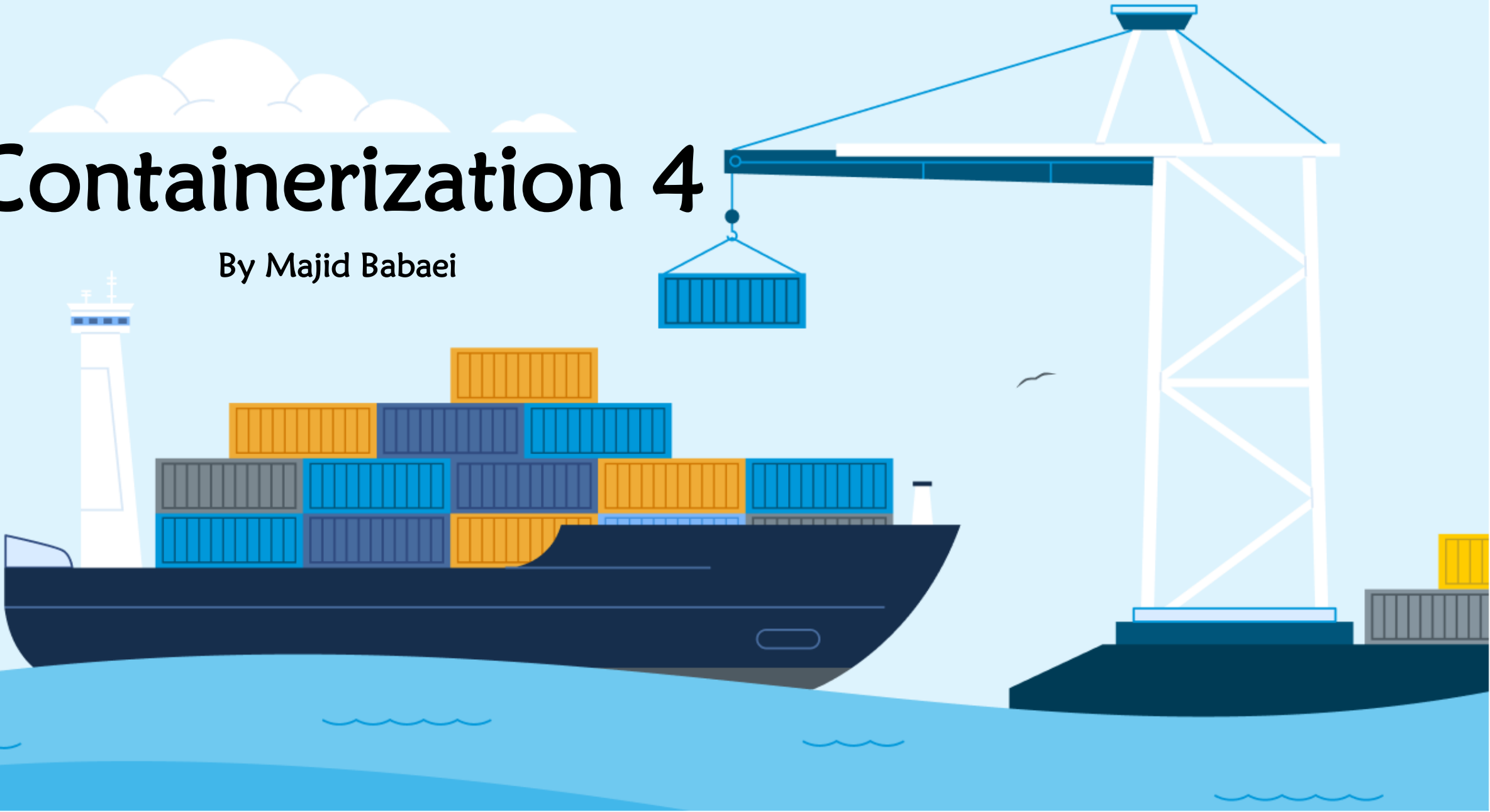


# Containerization 4

By Majid Babaei



```

1 FROM node:18.14-alpine
2 RUN addgroup app && adduser -S -G app app
3 USER app
4 WORKDIR /app
5 COPY . .
6 RUN npm install
7 ENV API_URL=http://api.myapp.com/
8 EXPOSE 3000
9 CMD ["npm", "run", "start"]


```

```

PS C:\Users\drbab> docker history react-app
IMAGE          CREATED          CREATED BY          SIZE
9bb849a78c66  19 minutes ago  CMD ["/bin/sh" "-c" "npm run start"]  0B
<missing>      19 minutes ago  EXPOSE map[3000/tcp:{}]              0B
<missing>      19 minutes ago  ENV API_URL=http://api.myapp.com/     0B
<missing>      19 minutes ago  RUN /bin/sh -c npm install # buildkit  395MB
<missing>      21 minutes ago  COPY . . # buildkit                   257kB
<missing>      36 minutes ago  WORKDIR /app                          0B
<missing>      36 minutes ago  USER app                             0B
<missing>      36 minutes ago  RUN /bin/sh -c addgroup app && adduser -S -G... 4.87kB
<missing>      4 months ago   /bin/sh -c #(nop) CMD ["node"]        0B
<missing>      4 months ago   /bin/sh -c #(nop) ENTRYPOINT ["docker-entry... 0B
<missing>      4 months ago   /bin/sh -c #(nop) COPY file:4d192565a7220e13... 388B
<missing>      4 months ago   /bin/sh -c apk add --no-cache --virtual .bui... 7.78MB
<missing>      4 months ago   /bin/sh -c #(nop) ENV YARN_VERSION=1.22.19 0B
<missing>      4 months ago   /bin/sh -c addgroup -g 1000 node && addu... 160MB
<missing>      4 months ago   /bin/sh -c #(nop) ENV NODE_VERSION=18.14.2 0B
<missing>      4 months ago   /bin/sh -c #(nop) CMD ["/bin/sh"]      0B
<missing>      4 months ago   /bin/sh -c #(nop) ADD file:40887ab7c06977737... 7.05MB
PS C:\Users\drbab>

```

 .dockerignore U  dockerfile U X

 dockerfile > ...

```

1 FROM node:18.14-alpine
2 RUN addgroup app && adduser -S -G app app
3 USER app
4 WORKDIR /app
5 COPY package*.json .
6 RUN npm install
7 COPY . .
8 ENV API_URL=http://api.myapp.com/
9 EXPOSE 3000
10 CMD ["npm", "run", "start"]

```

```

=> => transferring context: 53B 0.0s
=> [internal] load build definition from dockerfile 0.0s
=> => transferring dockerfile: 254B 0.0s
=> [internal] load metadata for docker.io/library/node:18.14-alpine 0.3s
=> [1/6] FROM docker.io/library/node:18.14-alpine@sha256:f8a51c36b0be743 0.0s
=> [internal] load build context 0.0s
=> => transferring context: 7.91kB 0.0s
=> CACHED [2/6] RUN addgroup app && adduser -S -G app app 0.0s
=> CACHED [3/6] WORKDIR /app 0.0s
=> CACHED [4/6] COPY package*.json . 0.0s
=> CACHED [5/6] RUN npm install 0.0s
=> [6/6] COPY . . 0.0s
=> exporting to image 0.0s
=> => exporting layers 0.0s
=> => writing image sha256:8a72cc28ca3e807046e99243b1dd83280ff6445476156 0.0s

```

# Removing Images

- When we run `> docker images`, we might see several images with **no name and no tag**
- They are called **dangling images** meaning loose images
- These are essentially **layers that have no relationship with a tagged image**
- As we are changing our dockefile and rebuilding our image, docker was creating these layers.
- As you work with docker, you will see these dangling images are popping up!
- But we can get rid of them!

```
PS C:\Users\drbab> docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
react-app	latest	8a72cc28ca3e	3 hours ago	570MB
<none>	<none>	2026e972ebe1	3 hours ago	570MB
<none>	<none>	9bb849a78c66	4 hours ago	570MB
<none>	<none>	caa4b852b3ed	4 hours ago	570MB
<none>	<none>	1e5e04b2040e	5 hours ago	471MB
<none>	<none>	5db785c0e23f	6 hours ago	471MB
<none>	<none>	1fc7031f47cd	6 hours ago	471MB
<none>	<none>	990f417f0368	9 hours ago	176MB
<none>	<none>	5436891e9b61	20 hours ago	466MB
alpine	latest	c1aabb73d233	3 weeks ago	7.33MB
ubuntu	latest	99284ca6cea0	4 weeks ago	77.8MB
<none>	<none>	29e2357271c8	4 months ago	175MB

# Removing Images

- We can use prune command

```
PS C:\Users\drbab> docker image prune
WARNING! This will remove all dangling images.
Are you sure you want to continue? [y/N] y
Deleted Images:
deleted: sha256:9bb849a78c66bea9dff4b5a896c97eab402f2cabd6a19cd313cafb0d94b657ea
deleted: sha256:2026e972ebe17cb2f45944a266b18c8b5e7d262f727fce2e3cbd1898d6ff758a

Total reclaimed space: 0B
```

- But there might be still some old images running by containers
- Let's get all the containers

```
PS C:\Users\drbab> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
0c520433b1e9	caa4b852b3ed	"docker-entrypoint.s..."	4 hours ago	Exited (1) 4 hours ago
72989cbb2730	1e5e04b2040e	"docker-entrypoint.s..."	5 hours ago	Exited (1) 4 hours ago
d1d9feea4d22	1e5e04b2040e	"docker-entrypoint.s..."	5 hours ago	Exited (0) 5 hours ago
4c2616d8d666	alpine:latest	"/bin/sh"	5 hours ago	Exited (0) 5 hours ago
d03572ed1c74	5db785c0e23f	"docker-entrypoint.s..."	6 hours ago	Exited (0) 5 hours ago
c2330ee20c79	1fc7031f47cd	"docker-entrypoint.s..."	7 hours ago	Exited (0) 6 hours ago
6b2a9dde2613	990f417f0368	"docker-entrypoint.s..."	9 hours ago	Exited (0) 7 hours ago
d458b9e992d6	5436891e9b61	"docker-entrypoint.s..."	20 hours ago	Exited (0) 9 hours ago
bb8bcab79c78	29e2357271c8	"docker-entrypoint.s..."	20 hours ago	Exited (0) 20 hours ago
2ad639fe769c	29e2357271c8	"docker-entrypoint.s..."	20 hours ago	Exited (1) 20 hours ago
b98cbce1c02a	29e2357271c8	"docker-entrypoint.s..."	20 hours ago	Exited (130) 20 hours ago
e736c1d88ab1	29e2357271c8	"docker-entrypoint.s..."	20 hours ago	Exited (1) 20 hours ago
e2b975cafd69	29e2357271c8	"docker-entrypoint.s..."	20 hours ago	Exited (1) 20 hours ago
bc6f3a476db0	29e2357271c8	"docker-entrypoint.s..."	20 hours ago	Exited (0) 20 hours ago
c0f84659ef0e	ubuntu	"/bin/bash"	31 hours ago	Up 31 hours
n/bash"	4 days ago	Up 4 days		mystifying_swirles
a4c48823bdbb	ubuntu	"/bin/bash"	6 days ago	Exited (0) 4 days ago
b0bfdb7f3279	ubuntu:latest	"/bin/bash"	6 days ago	Exited (0) 6 days ago
d3fd8644bdf8	ubuntu:latest	"/bin/bash"	6 days ago	Exited (0) 6 days ago



# Removing Images

- We can get rid of these stopped containers

```
PS C:\Users\drbab> docker container prune
WARNING! This will remove all stopped containers.
Are you sure you want to continue? [y/N] y
Deleted Containers:
0c520433b1e91a54ae1a442155dea3bab6d82b33e404229b4f6b10ca1e616257
72989cbb2730d70a4b69ac6cb0a598804d3f522030003b41c5d31beb0b377117
d1d9feea4d22b3c9304ba53c9512423d0a8bb47fb069e930bb3699b38b16287c
4c2616d8d666b4bb20e9ba7256a4f7f77117299c4cccca3a613442a37dff236a
d03572ed1c7427618e089fe740fcdca936b3b189236c29cc8293e564940b6783e
c2330ee20c792c8058d308827b349c15357f6768399b44b294b54c7e105bfcf0
6b2a9dde26130868cbfe250ae2c3aa7002a3dc3c00d6b6c3423a709116b70c86
d458b9e992d66c6109ebf7b02cc5d7e6c5c612dcff5795da2027c32de7e4cd39
bb8bcab79c78d896d91e3d6b7cdca147f9439906f5b49de309df8114b88e1225
2ad639fe769c249ff8f3807d22c87e38b8f8a950945f18f802db27fd5bf61da7
b98cbce1c02ac7115193cafad7035c6850a42f1f69fc88df9a361cd5af19da79
e736c1d88ab1aae8abd75e1834465130cd4f28496cd9dfe627ee17480e51098d
e2b975cafd69e2168f1b6e5301171cd9c54d5999283035389ff1a4afd01f5e17
bc6f3a476db0243812821868bf97d9605e00cbfc5fa9ca2344eceafbe8e0e873
a4c48823bdbb6cbb12801645cdebcea716b3d8cc7bc1e23d56435514c6dae15e
b0bfdb7f327923dc134f2d20d639375d271abd0c3c22ed4ce2706ee6ff5e8841
d3fd8644bdf84de4bc226eacfb277b079e6d3fb4db01a5dee182c06a07ee04b3

Total reclaimed space: 97.31MB
```

# Removing Images

- Now we can prune all the dangling images

```
PS C:\Users\drbab> docker image prune
WARNING! This will remove all dangling images.
Are you sure you want to continue? [y/N] y
Deleted Images:
deleted: sha256:caa4b852b3edc7f7ba7115f2c33cf328d6a1d8725db79dd6ecb8345bfc8a3df2
deleted: sha256:29e2357271c8e86ba0b3fc183c3b2be6597539bea1e881218b62324311e5de73
deleted: sha256:5436891e9b61a9f14abbcfa4c0401ae8e799611996e7b5f345f6e8ebda80c376
deleted: sha256:5db785c0e23faacc60d34283a0d0d4e820d0c6434c5e21d6621c2d9d2e5f3c7c
deleted: sha256:1fc7031f47cd44bf9be9739cb2f22f38721bbea1086aa3ea4c249b102697fbfd
deleted: sha256:990f417f0368dbea2169c16d29da10ab5a60579a4e000707981ac25d531f5b96
deleted: sha256:1e5e04b2040e65d14d6edee9b7c8853b9ad38a5b353cdb35b1df439fc7d41ada

Total reclaimed space: 0B
```

- Now we can get the list of all images

```
PS C:\Users\drbab> docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
react-app	latest	8a72cc28ca3e	4 hours ago	570MB
alpine	latest	c1aabb73d233	3 weeks ago	7.33MB
ubuntu	latest	99284ca6cea0	4 weeks ago	77.8MB

# Tagging Images

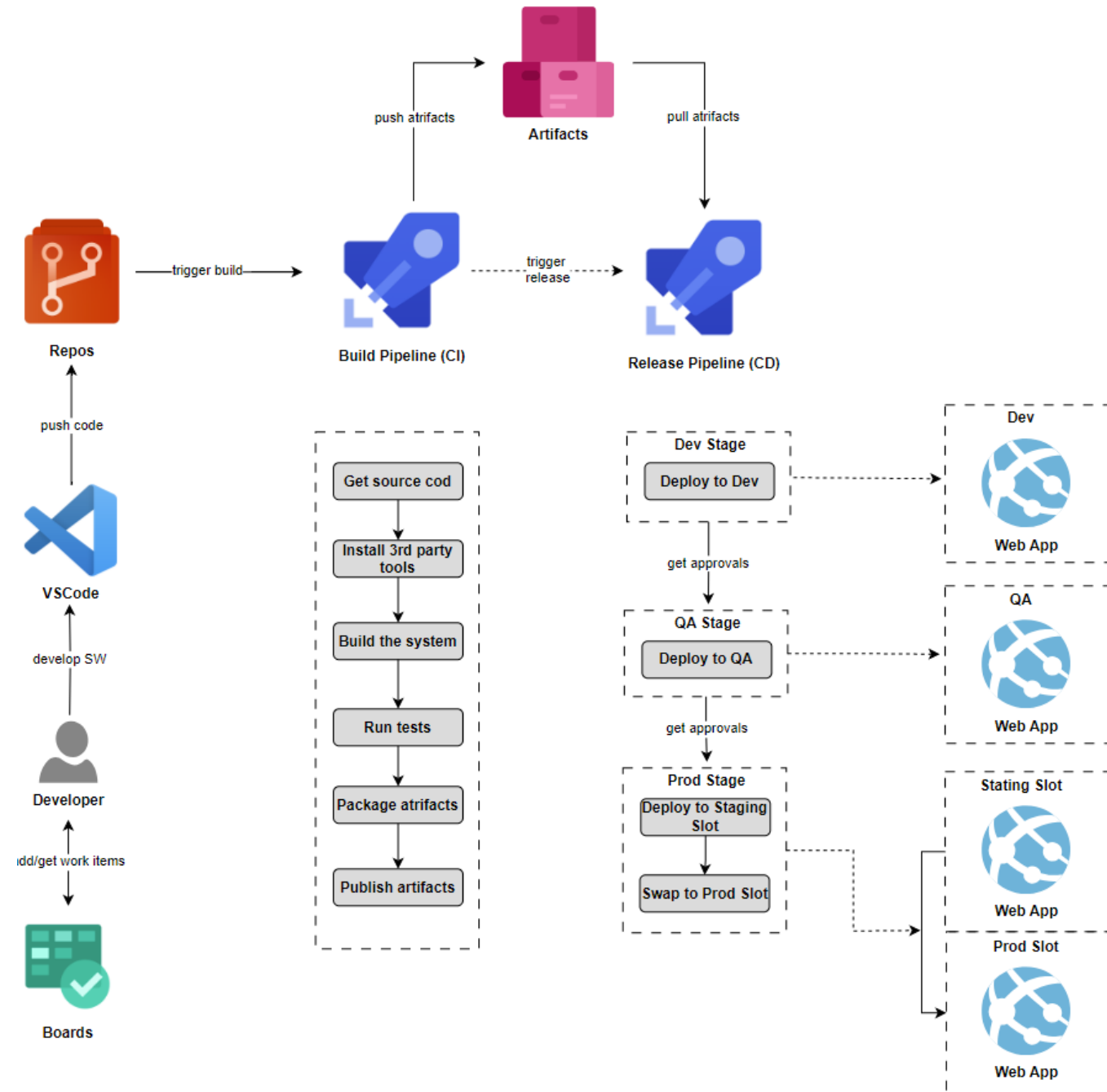
- Whenever we build an image or pull it from a dockerhub, by default docker uses the *latest* tag
- The *latest* tag is just a label! It doesn't necessarily mean it is the latest version of your image. If you don't tag your images properly, latest can point to an old image.
- Using *latest* tag is fine in development, but in production can cause confusion.
- If something goes wrong with one of your client running your app, you cannot easily troubleshoot issues. Because latest is not very meaningful and do not refer to any special version of your app!
- You should always use explicit tags in each environment (test, staging or production)

# How can we tag an image?

- There are two ways:
  - Tag an image while building the image
  - Tan an image after building the image
- To name a tag there are different approaches:
  - Some teams use *code name* for each version of the product, e.g., *buster*, everyone knows what buster is!
    - Then they will use this: **docker build -t react-app:buster**
  - Other teams prefer *semantic versioning*, e.g., 3.10.2 (common among teams that don't release frequently)
    - So they can use: **docker build -t react-app:3.10.2**
  - For teams that build too often use more natural way!
    - They usually use a number for each version: **docker build -t react-app:76**
    - Often this will be handled by CI/CD tools



- Checkout the latest code from the Repo
- Build an image
- Automatically tag it with the proper number



*Note: An image can have multiple tags!*

```
> ✓ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
react-app	1	f4b1daa5babc	14 minutes ago	297MB
react-app	latest	f4b1daa5babc	14 minutes ago	297MB
ubuntu	latest	4dd97cefde62	12 days ago	72.9MB
alpine	latest	28f6e2705743	3 weeks ago	5.61MB

We can tag an image after building it!

```
docker image tag react-app:latest react-app:1
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
react-app	1	f4b1daa5babc	15 minutes ago	297MB
react-app	latest	f4b1daa5babc	15 minutes ago	297MB
ubuntu	latest	4dd97cefde62	12 days ago	72.9MB
alpine	latest	28f6e2705743	3 weeks ago	5.61MB

# The latest tag can get out of order!

1. Back to the source code, make a tiny change (it simulates a change in the app code)
2. Build a new image

```
> ✓ docker build -t react-app:2 .  
[+] Building 1.2s (11/11) FINISHED  
=> [internal] load build definition from Dockerfile  
=> => transferring dockerfile: 37B  
=> [internal] load .dockerignore  
=> => transferring context: 34B
```

3. Let's look at the images

```
> ✓ docker images
```

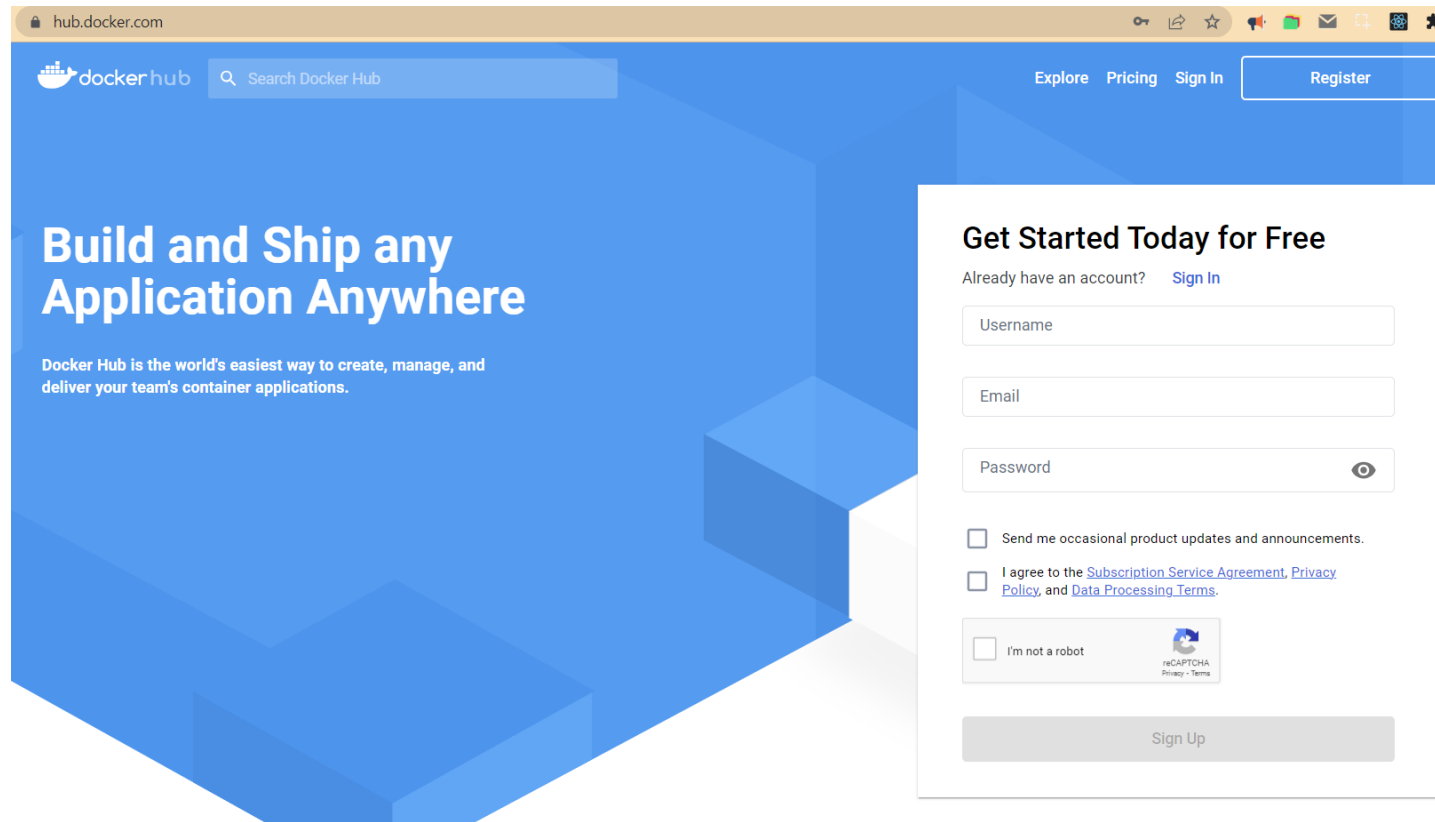
REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
react-app	2	b06073bda4c6	4 seconds ago	297MB
react-app	1	f4b1daa5babc	16 minutes ago	297MB
react-app	latest	f4b1daa5babc	16 minutes ago	297MB
ubuntu	latest	4dd97cefde62	12 days ago	72.9MB
alpine	latest	28f6e2705743	3 weeks ago	5.61MB

*The latest tag don't necessary reference to the latest image, you have to explicitly apply it to the latest image!*

```
docker image tag b06 react-app:latest
```

# Sharing Images with others!

- Head over to: <https://hub.docker.com/>
- Register, it is free!



The screenshot shows the Docker Hub website with a registration modal open. The website header includes the Docker Hub logo, a search bar, and navigation links for Explore, Pricing, Sign In, and Register. The main content area features the text "Build and Ship any Application Anywhere" and a description of Docker Hub. The registration modal, titled "Get Started Today for Free", contains a link for existing users to "Sign In", input fields for Username, Email, and Password, and checkboxes for product updates, terms agreement, and a reCAPTCHA verification.

hub.docker.com

docker hub Search Docker Hub

Explore Pricing Sign In Register

**Build and Ship any Application Anywhere**

Docker Hub is the world's easiest way to create, manage, and deliver your team's container applications.

**Get Started Today for Free**

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


Username

Email

Password

☐ Send me occasional product updates and announcements.

☐ I agree to the [Subscription Service Agreement](#), [Privacy Policy](#), and [Data Processing Terms](#).

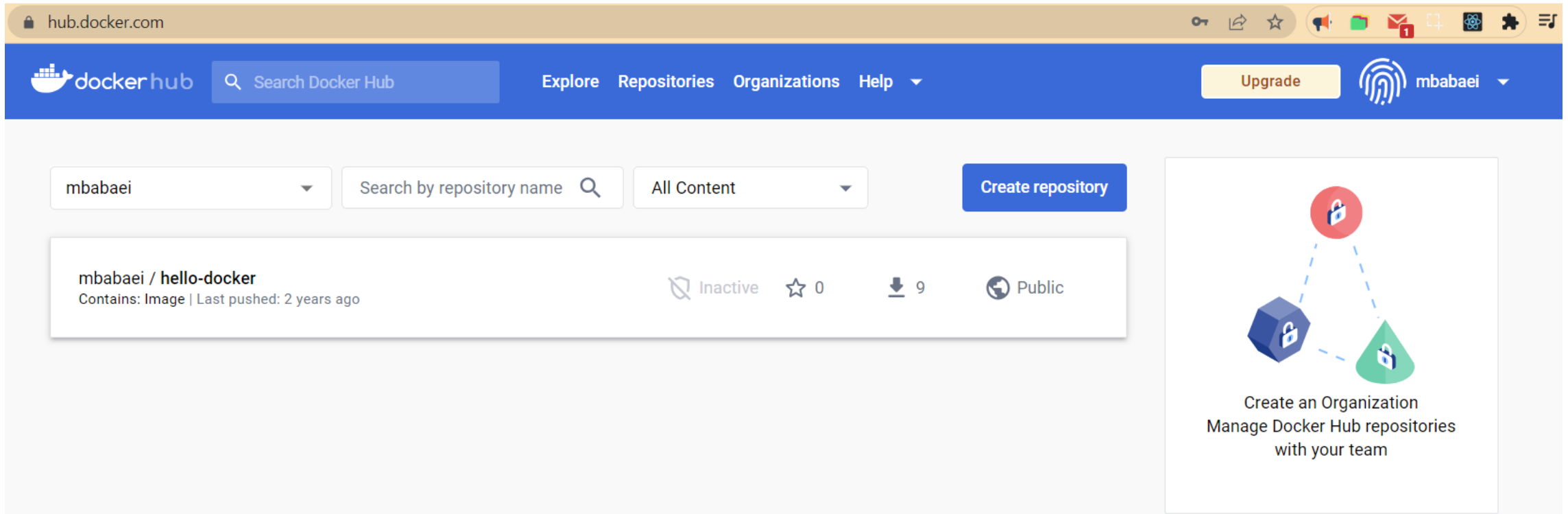
☐ I'm not a robot 

Sign Up



# Sharing Images with others!

- Use Create Repository
- In repository you can have multiple images with different tags



The screenshot displays the Docker Hub web interface. The top navigation bar includes the Docker Hub logo, a search bar, and links for Explore, Repositories, Organizations, and Help. A user profile for 'mbabaei' is visible in the top right corner. Below the navigation bar, the main content area shows a repository for 'mbabaei / hello-docker'. The repository details include 'Contains: Image | Last pushed: 2 years ago', 'Inactive' status, 0 stars, 9 downloads, and 'Public' visibility. A sidebar on the right contains a graphic with three Docker icons and the text 'Create an Organization Manage Docker Hub repositories with your team'.

hub.docker.com

docker hub Search Docker Hub Explore Repositories Organizations Help Upgrade mbabaei

mbabaei Search by repository name All Content Create repository

mbabaei / **hello-docker**  
Contains: Image | Last pushed: 2 years ago

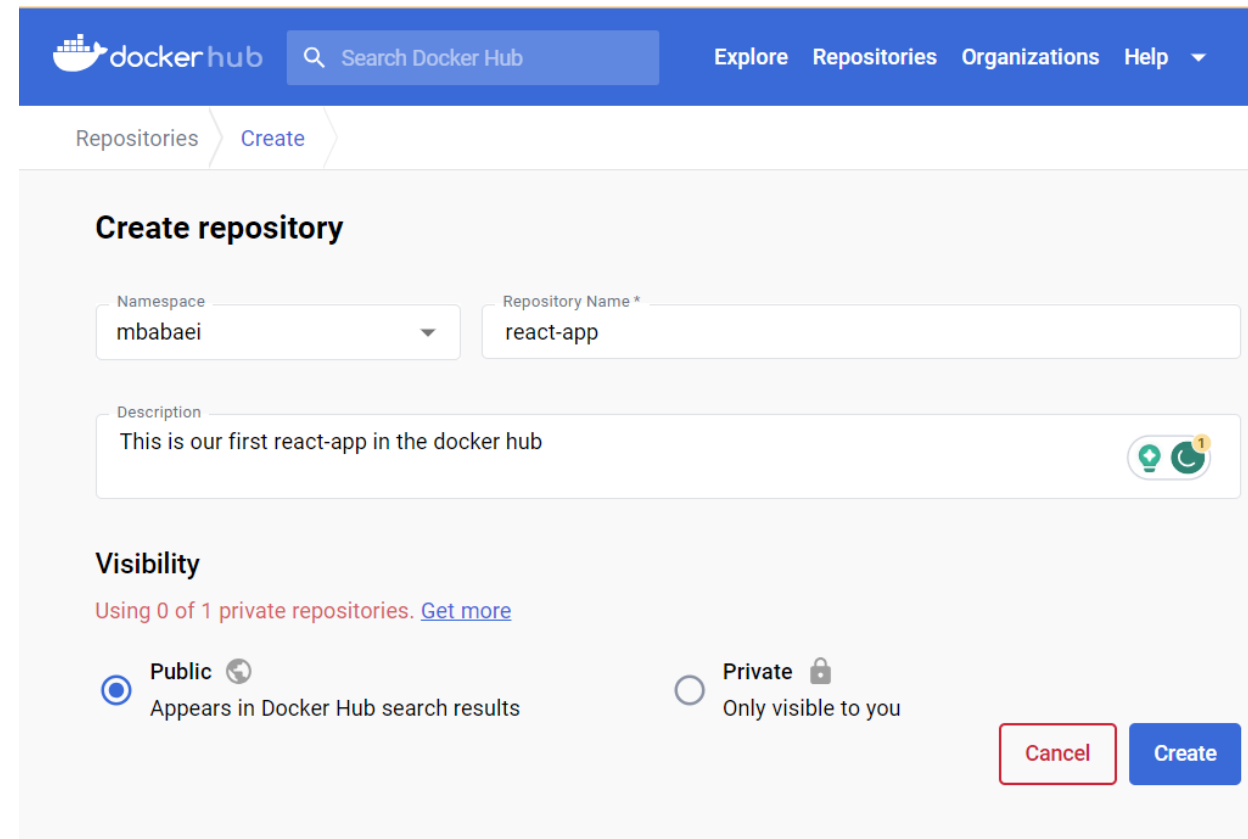
Inactive 0 9 Public

Create an Organization  
Manage Docker Hub repositories  
with your team

# Sharing Images with others!

- The name of the repo is: mbabaei/react-app
- To push our image to this repo we need to give it a tag with the same name

- Optionally we can connect our account to a GitHub account, so every time we do a push, dockerhub will pull the latest code and build a new image!




The screenshot shows the Docker Hub 'Create repository' page. The top navigation bar is blue with the Docker Hub logo, a search bar, and links for 'Explore', 'Repositories', 'Organizations', and 'Help'. Below the navigation bar, there are two tabs: 'Repositories' and 'Create', with 'Create' being the active tab. The main form is titled 'Create repository' and contains the following fields:

- Namespace:** A dropdown menu with 'mbabaei' selected.
- Repository Name \*:** A text input field containing 'react-app'.
- Description:** A text area containing 'This is our first react-app in the docker hub'.
- Visibility:** Two radio button options: 'Public' (selected) and 'Private'.


Below the 'Public' option, it says 'Appears in Docker Hub search results'. Below the 'Private' option, it says 'Only visible to you'. At the bottom right, there are two buttons: 'Cancel' (outlined in red) and 'Create' (solid blue).

# Sharing Images with others!



Explore Repositories Organizations Help ▾


Upgrade



 mbabaei ▾

mbabaei > Repositories > react-app > General

Using 0 of 1 private repositories. [Get more](#)

General Tags Builds Collaborators Webhooks Settings

 **mbabaei / react-app**

**Description**  
This is our first react-app in the docker hub   
 Last pushed: a few seconds ago

**Docker commands** [Public View](#)  
To push a new tag to this repository,  

```
docker push mbabaei/react-app:tagname
```

# Sharing Images with others!

- We give the image the proper tag: `docker image tag 8a72 mbabaei/react-app:1`

```
PS C:\Windows\System32> docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
react-app     latest    8a72cc28ca3e   5 hours ago   570MB
alpine        latest    c1aabb73d233   3 weeks ago   7.33MB
ubuntu        latest    99284ca6cea0   4 weeks ago   77.8MB
PS C:\Windows\System32> docker image tag 8a72 mbabaei/react-app:1
PS C:\Windows\System32> docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
mbabaei/react-app  1         8a72cc28ca3e   5 hours ago   570MB
react-app     latest    8a72cc28ca3e   5 hours ago   570MB
alpine        latest    c1aabb73d233   3 weeks ago   7.33MB
ubuntu        latest    99284ca6cea0   4 weeks ago   77.8MB
```

- Note: these two tags are pointing to the same image!
- Now we are ready to push it to our repo
- First we need to login:

```
PS C:\Users\drbab> docker login
Authenticating with existing credentials...
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more
```

## Docker commands

[Public View](#)

To push a new tag to this repository,

```
docker push mbabaei/react-app:tagname
```

```
PS C:\Windows\System32> docker login
Authenticating with existing credentials...
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/
PS C:\Windows\System32> docker push mbabaei/react-app:1
The push refers to repository [docker.io/mbabaei/react-app]
9f24f3beae97: Pushed
aba30e0dbf47: Pushing [=====>] 80.76MB/394.6MB
51b137c85d5a: Pushed
506d1d6b5867: Pushed
95377179def5: Pushed
dc923cea9549: Pushing [=====>] 3.584kB
d26cfa2c5d93: Pushing [=====>] 7.819MB
ec7aaa5c3b9b: Pushing [==>] 6.649MB/160MB
7cd52847ad77: Pushing [=====>] 4.428MB/7.05MB
```

- Now docker is pushing all layers in our image to the repository



EXPLORER

REACT-APP

> node\_modules

> public

> src

.dockerignore U

.gitignore

dockerfile U

package.json

README.md M

... .dockerignore U dockerfile U README.md M X

README.md > # yet another line!

1 # yet another line!

2

3 # new line added to this file!

4

5 # Getting Started with Create React App

6

7 This project was bootstrapped with [Create React App](https://github.com/facebook/create-react-app).

8

9 ## Available Scripts

10

11 In the project directory, you can run:

12

13 ### `npm start`

14

15 Runs the app in the development mode.\

16 Open [http://localhost:3000](http://localhost:3000) to view it in your browser.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

powershell + - X

● PS C:\Users\drbab\OneDrive - McGill University\@Courses\ECSE 437 Software Delivery - Fall'23\Docker\sample-react-app\react-app> docker build -t mbabaei/react-app:2 .

[+] Building 0.3s (11/11) FINISHED docker:default

=> [internal] load .dockerignore 0.0s

=> => transferring context: 53B 0.0s

=> [internal] load build definition from dockerfile 0.0s

=> => transferring dockerfile: 254B 0.0s

=> [internal] load metadata for docker.io/library/node:18.14-alpine 0.3s


> OUTLINE

> TIMELINE


- Let's push it with the new tag: **docker push mbabaei/react-app:2**

```
PS C:\Users\drbab\OneDrive - McGill University\@Courses\ECSE 437 Software Delivery - Fall'23\Docker\sample-react-app\react-app> docker push mbabaei/react-app:2
The push refers to repository [docker.io/mbabaei/react-app]
1c00edce4f68: Pushed
aba30e0dbf47: Layer already exists
51b137c85d5a: Layer already exists
506d1d6b5867: Layer already exists
95377179def5: Layer already exists
dc923cea9549: Layer already exists
d26cfa2c5d93: Layer already exists
ec7aaa5c3b9b: Layer already exists
7cd52847ad77: Layer already exists
2: digest: sha256:0b3e50bddcf7739796c665cee51340aae10d67776e6904032ffdf35225c132f size: 2203
```

- As you can see we don't need to push several layers!
- The push this time is much faster.

[Explore](#) [Repositories](#) [Organizations](#) [Help](#) [Upgrade](#)mbabaei [mbabaei](#) [Repositories](#) [react-app](#) [Tags](#)Using 0 of 1 private repositories. [Get more](#)[General](#) [Tags](#) [Builds](#) [Collaborators](#) [Webhooks](#) [Settings](#)☐ Sort by Newest ▾   [Go to Advanced Image Management](#) [Delete](#)

TAG

[2](#)Last pushed 3 minutes ago by [mbabaei](#)`docker pull mbabaei/react-app:2` 

DIGEST

[0b3e50bddcf7](#)

OS/ARCH

linux/amd64


LAST PULL

---

COMPRESSED SIZE ⓘ

180.54 MB

TAG

[1](#)Last pushed 15 minutes ago by [mbabaei](#)`docker pull mbabaei/react-app:1` 

DIGEST

[08d3efbbd9d6](#)

OS/ARCH

linux/amd64

LAST PULL

---

COMPRESSED SIZE ⓘ

180.54 MB

Now that our image is on dockerhub we can put it on any machine that runs docker application!



Working with containers



- Starting and stopping containers
- Publishing ports
- Viewing logs
- Executing commands in containers
- Removing containers
- Persisting data using volumes

# What is a container?

- A container is an isolated group of processes that are restricted to a private filesystem (coming from the image) and a process namespace.
- While the host machine knows that the container is actually a process, the container thinks that it is a separate machine.
- kernel features that make this possible are:
  - **namespaces** = Namespaces are the feature that make the container look and feel like it is an entirely separate machine.
  - **cgroups** = A way to group processes together in the kernel and limit resources for that grouping.

```
PS C:\Users\drbab> docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

```
PS C:\Users\drbab> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
abfa40d9e5bd	react-app:latest	"docker-entrypoint.s..."	3 minutes ago	Exited (1) 8 seconds ago		funny_black
bcf13743e237	react-app:latest	"docker-entrypoint.s..."	8 minutes ago	Exited (1) 3 minutes ago		nifty_ishizaka
c0f84659ef0e	ubuntu	"/bin/bash"	2 days ago	Exited (0) 17 hours ago		zen_feynman
de2e27a6293e	ubuntu	"/bin/bash"	5 days ago	Exited (0) 17 hours ago		mystifying_swirles

```
PS C:\Users\drbab> docker run react-app:latest
```

```
> react-app@0.1.0 start
```

```
> react-scripts start
```

```
(node:26) [DEP_WEBPACK_DEV_SERVER_ON_AFTER_SETUP_MIDDLEWARE] DeprecationWarning: 'onAfterSetupMiddleware' option is deprecated. Please use 'onAfterSetupDevServer' instead. (Use `node --trace-deprecation ...` to show where the warning was created)
```

```
(node:26) [DEP_WEBPACK_DEV_SERVER_ON_BEFORE_SETUP_MIDDLEWARE] DeprecationWarning: 'onBeforeSetupMiddleware' option is deprecated. Please use 'onBeforeSetupDevServer' instead. Starting the development server...
```

One of your dependencies, babel-preset-react-app, is importing the "@babel/plugin-proposal-private-property-in-object" package without declaring it in its dependencies. This is currently working because "@babel/plugin-proposal-private-property-in-object" is already in your node\_modules folder for unrelated reasons, but it may break at any time.

babel-preset-react-app is part of the create-react-app project, which is not maintained anymore. It is thus unlikely that this bug will ever be fixed. Add "@babel/plugin-proposal-private-property-in-object" to your devDependencies to work around this error. This will make this message go away.

```
Compiled successfully!
```

```
You can now view react-app in the browser.
```

```
Local: http://localhost:3000
```

```
On Your Network: http://172.17.0.2:3000
```

You can run the container in the detached mode!

```
npm ERR! A complete log of this run can be found in:  
npm ERR!     /home/app/.npm/_logs/2023-07-10T15_45_03_714Z-debug-0.log
```

```
PS C:\Users\drbab> docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

```
PS C:\Users\drbab> docker run -d react-app:latest
```

```
16d85e941b39522415c88a6186eb74eceb738462ec6365c4ea80f5083bfb5b7e
```

```
PS C:\Users\drbab> docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
16d85e941b39	react-app:latest	"docker-entrypoint.s..."	2 seconds ago	Up 2 seconds	3000/tcp	quizzical_davinci

# Viewing Logs

- We have some containers running in the background, but we don't know what is going on inside these containers (*what if something goes wrong?!*)

```
PS C:\Users\drbab> docker run -d react-app:latest
10b9599c29d3f3f77c40654628a863ba109fa11955c4c72d3201d9c4bd0b2f34
PS C:\Users\drbab> docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
10b9599c29d3	react-app:latest	"docker-entrypoint.s..."	3 seconds ago	Up 2 seconds	3000/tcp	wonderful_jackson
16d85e941b39	react-app:latest	"docker-entrypoint.s..."	2 minutes ago	Up 2 minutes	3000/tcp	quizzical_davinci

```
PS C:\Users\drbab> docker logs --help

Usage:  docker logs [OPTIONS] CONTAINER

Fetch the logs of a container

Aliases:
  docker container logs, docker logs

Options:
  --details          Show extra details provided to logs
  -f, --follow        Follow log output
  --since string      Show logs since timestamp (e.g.
                     "2013-01-02T13:23:37Z") or relative (e.g. "42m"
                     for 42 minutes)
  -n, --tail string   Number of lines to show from the end of the logs
                     (default "all")
  -t, --timestamps    Show timestamps
  --until string       Show logs before a timestamp (e.g.
                     "2013-01-02T13:23:37Z") or relative (e.g. "42m"
                     for 42 minutes)
```



```
PS C:\Users\drbab> docker logs -f 10b9599c29d3
```

```
> react-app@0.1.0 start  
> react-scripts start
```

```
PS C:\Users\drbab> docker logs -n 5 10b9599c29d3
```

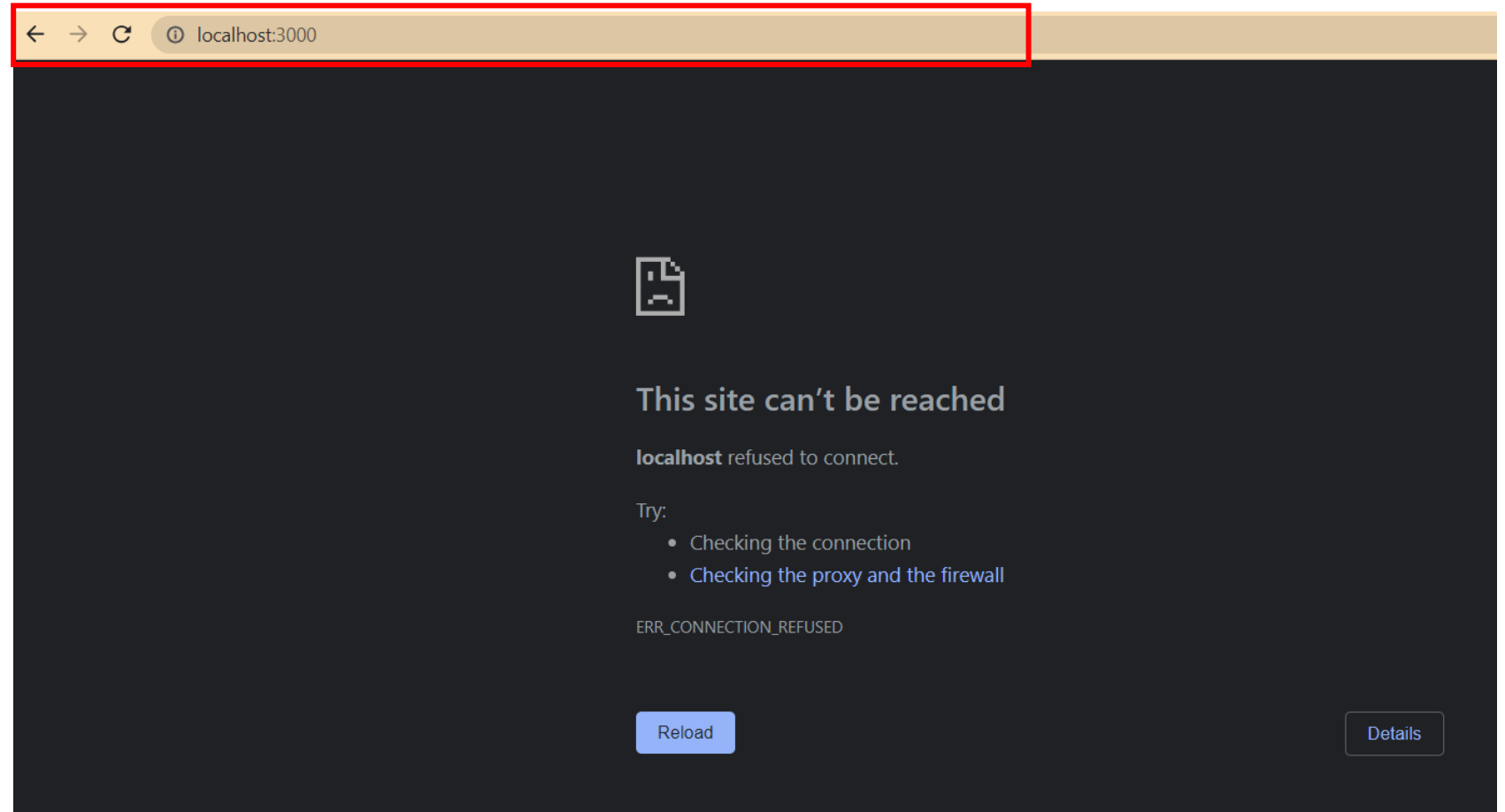
```
webpack compiled successfully  
Compiling...  
Compiled successfully!  
webpack compiled successfully
```

```
PS C:\Users\drbab> docker logs -n 5 -t 10b9599c29d3
```

```
2023-07-10T15:50:24.059650844Z  
2023-07-10T15:50:24.065201563Z webpack compiled successfully  
2023-07-10T15:50:24.118157534Z Compiling...  
2023-07-10T15:50:24.227126361Z Compiled successfully!  
2023-07-10T15:50:24.228997430Z webpack compiled successfully
```

# Publishing ports

- We have some containers running in the background.
- Each container is listening to port 3000
- But it is not mapped to any port in the host, so we cannot get access to the ports on the containers.



# Publishing ports

- We can use this pattern [host port]:[container port]

```
PS C:\Users\drbab> docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
10b9599c29d3	react-app:latest	"docker-entrypoint.s..."	9 minutes ago	Up 9 minutes	3000/tcp	wonderful_jackson
16d85e941b39	react-app:latest	"docker-entrypoint.s..."	11 minutes ago	Up 11 minutes	3000/tcp	quizzical_davinci

```
PS C:\Users\drbab> docker run -d -p 4000:3000 --name Accessible_Container react-app:latest
```

```
7c8366a44d3e4358095ac6e39020d104fc3da08b9ae2c81ca2b0a8b756e20248
```

```
PS C:\Users\drbab> docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
7c8366a44d3e	react-app:latest	"docker-entrypoint.s..."	About a minute ago	Up About a minute	0.0.0.0:4000->3000/tcp	Accessible_Container
10b9599c29d3	react-app:latest	"docker-entrypoint.s..."	14 minutes ago	Up 14 minutes	3000/tcp	wonderful_jackson
16d85e941b39	react-app:latest	"docker-entrypoint.s..."	17 minutes ago	Up 17 minutes	3000/tcp	quizzical_davinci





Edit `src/App.js` and save to reload.

[Learn React](#)

# Executing commands on a running container

- How can we interact with a running container without forcing it to stop and start again?
- We can use *exec* command!

```
PS C:\Users\drbab> docker exec Accessible_Container ls
README.md
dockerfile
node_modules
package-lock.json
package.json
public
src
```

- It shows the content of the app directory inside the running container
- We can maintain this interaction using *-it* in the *exec* command and using shell

```
PS C:\Users\drbab> docker exec -it Accessible_Container sh
/app $ whoami
app
/app $ ls
README.md      dockerfile      node_modules    package-lock.json  package.json      public          src
/app $
```

# Start vs. Run

- To start an existing stopped container, we can use *start* command
- When we use *run* command, we create a new container!

```
PS C:\Users\drbab> docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
7c8366a44d3e   react-app:latest "docker-entrypoint.s..." 15 minutes ago Up 15 minutes   0.0.0.0:4000->3000/tcp    Accessible_Container
10b9599c29d3   react-app:latest "docker-entrypoint.s..." 29 minutes ago Up 29 minutes   3000/tcp                 wonderful_jackson
16d85e941b39   react-app:latest "docker-entrypoint.s..." 32 minutes ago Up 32 minutes   3000/tcp                 quizzical_davinci
7b2d15ac065d   react-app:latest "docker-entrypoint.s..." 34 minutes ago Exited (1) 32 minutes ago friendly_leavitt
abfa40d9e5bd   react-app:latest "docker-entrypoint.s..." 38 minutes ago Exited (1) 35 minutes ago funny_black
bcf13743e237   react-app:latest "docker-entrypoint.s..." 43 minutes ago Exited (1) 38 minutes ago nifty_ishizaka
c0f84659ef0e   ubuntu        "/bin/bash"             2 days ago    Exited (0) 17 hours ago  zen_feynman
de2e27a6293e   ubuntu        "/bin/bash"             5 days ago    Exited (0) 17 hours ago  mystifying_swirles

PS C:\Users\drbab> docker start --help

Usage:  docker start [OPTIONS] CONTAINER [CONTAINER...]

Start one or more stopped containers

Aliases:
  docker container start, docker start

Options:
  -a, --attach          Attach STDOUT/STDERR and forward signals
  --detach-keys string  Override the key sequence for detaching a
                        container
  -i, --interactive     Attach container's STDIN

PS C:\Users\drbab> docker start -i c0f84659ef0e
root@c0f84659ef0e:/# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
root@c0f84659ef0e:/#
```



# Remove a container

- You can use `-f` to stop and remove a running container

```
PS C:\Users\drbab> docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
7c8366a44d3e	react-app:latest	"docker-entrypoint.s..."	20 minutes ago	Up 20 minutes	0.0.0.0:4000->3000/tcp	Accessible_Container
10b9599c29d3	react-app:latest	"docker-entrypoint.s..."	34 minutes ago	Up 34 minutes	3000/tcp	wonderful_jackson
16d85e941b39	react-app:latest	"docker-entrypoint.s..."	37 minutes ago	Up 37 minutes	3000/tcp	quizzical_davinci

```
PS C:\Users\drbab> docker rm 10b9599c29d3
```

Error response from daemon: You cannot remove a running container 10b9599c29d3f3f77c40654628a863ba109fa11955c4c72d3201d9c4bd0b2f34. Stop the container first.

```
PS C:\Users\drbab> docker rm -f 10b9599c29d3
```

```
10b9599c29d3
```

```
PS C:\Users\drbab> docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
7c8366a44d3e	react-app:latest	"docker-entrypoint.s..."	21 minutes ago	Up 21 minutes	0.0.0.0:4000->3000/tcp	Accessible_Container
16d85e941b39	react-app:latest	"docker-entrypoint.s..."	38 minutes ago	Up 38 minutes	3000/tcp	quizzical_davinci

- You can use `docker container prune` to get rid of all the stopped containers in one go!

# Container file system

- Each container has its own file system, hidden from other containers!

```
PS C:\Users\drbab> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
7c8366a44d3e   react-app:latest  "docker-entrypoint.s..." 38 minutes ago Up 38 minutes  0.0.0.0:4000->3000/tcp    Accessible_Container
16d85e941b39   react-app:latest  "docker-entrypoint.s..." 54 minutes ago Up 54 minutes  3000/tcp                 quizzical_davinci
PS C:\Users\drbab> docker exec -it Accessible_Container sh
/app $ ls
README.md      dockerfile     node_modules    package-lock.json package.json    public          src
/app $ echo "a new line" > data.txt
/app $ ls
README.md      data.txt       dockerfile      node_modules    package-lock.json package.json    public          src
/app $ cat data.txt
a new line
/app $ exit
PS C:\Users\drbab> docker exec -it quizzical_davinci sh
/app $ ls
README.md      dockerfile     node_modules    package-lock.json package.json    public          src
/app $
```

- If we delete a container, its file system will go with it! And we lose our data!
- We should not store our data in the file system of a container!
- How do you ensure data persistence in containers?

# Persisting data using Volumes

- A volume is a storage outside of a container
- It can be directly on the host or somewhere on the cloud.

```
PS C:\Users\drbab> docker volume
```

```
Usage:  docker volume COMMAND
```

```
Manage volumes
```

```
Commands:
```

create	Create a volume
inspect	Display detailed information on one or more volumes
ls	List volumes
prune	Remove all unused local volumes
rm	Remove one or more volumes

```
Run 'docker volume COMMAND --help' for more information on a command.
```

```
PS C:\Users\drbab> docker volume create app-data
```

```
app-data
```

```
PS C:\Users\drbab> docker volume inspect app-data
```

```
[
  {
    "CreatedAt": "2023-07-10T16:48:33Z",
    "Driver": "local",
    "Labels": null,
    "Mountpoint": "/var/lib/docker/volumes/app-data/_data",
    "Name": "app-data",
    "Options": null,
    "Scope": "local"
  }
]
```

Let's see how we can run a container  
and using volumes for persisting data.

# Persisting data using Volumes

```
PS C:\Users\drbab> docker run -d -p 4000:3000 --name Accessible_Container_v1 -v app-data:/app/data react-app:latest
f29056842e01b1a44caff8fbff6103efa88d190d04c3c0170bcc2eef971b4a36

PS C:\Users\drbab> docker exec -it Accessible_Container_v1 sh
/app $ ls
README.md      dockerfile      package-lock.json  public
data           node_modules    package.json       src
/app $ cd data/
/app/data $ echo "new line" > data.txt
sh: can't create data.txt: Permission denied
/app/data $
```

- Note: app-data and /app/data do not need to be exist before running this command
- But we are getting a permission issue!

```

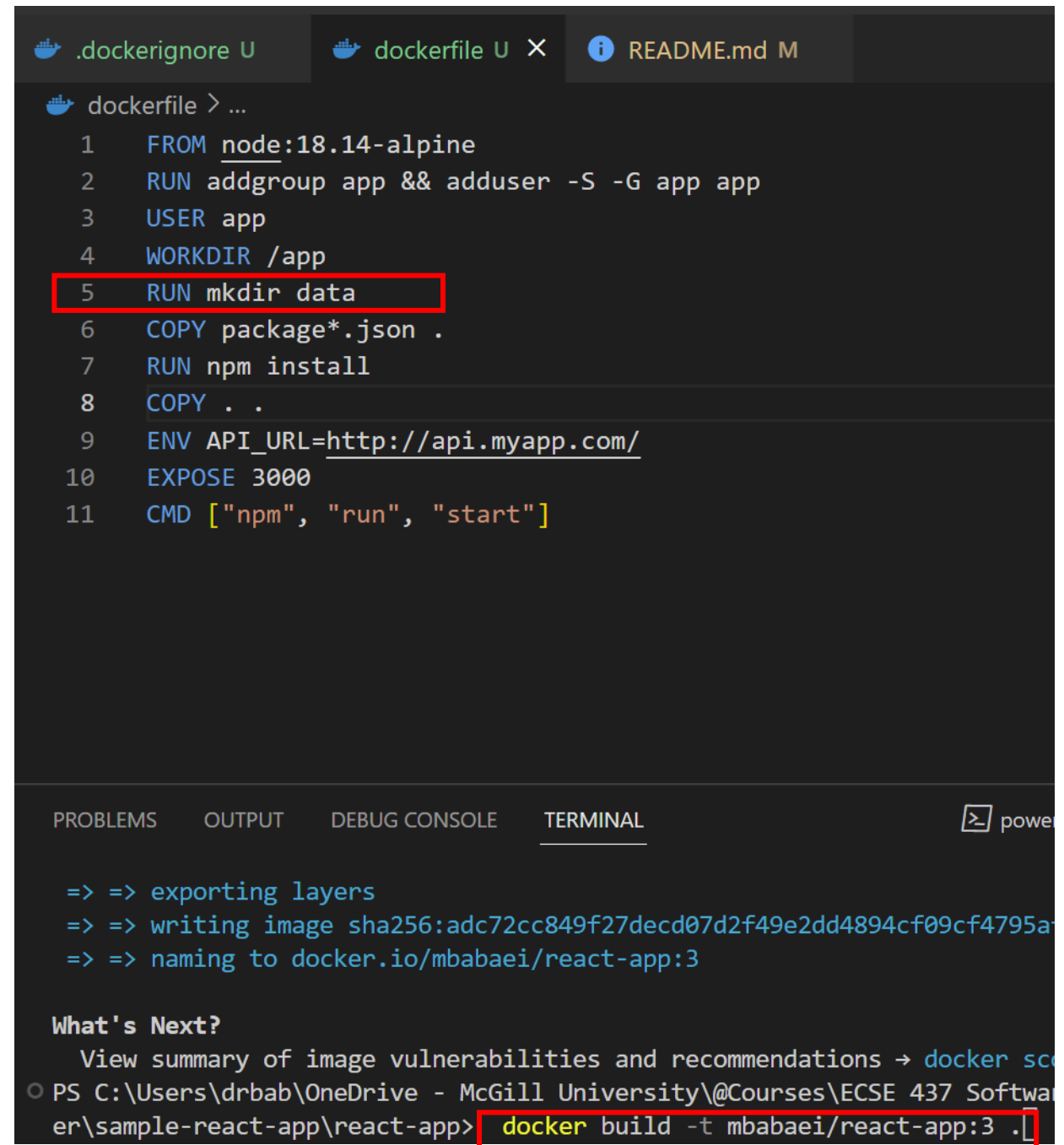
/app $ ls -al
total 700
drwxr-xr-x    1 app      app      4096 Jul 10 17:00 .
drwxr-xr-x    1 root    root     4096 Jul 10 17:00 ..
-rwxr-xr-x    1 root    root       13 Jul  9 12:35 .dockerignore
drwxr-xr-x    7 root    root     4096 Jul  9 17:53 .git
-rwxr-xr-x    1 root    root     310 Jul  8 15:39 .gitignore
-rwxr-xr-x    1 root    root    3391 Jul  9 17:52 README.md
drwxr-xr-x    2 root    root     4096 Jul 10 16:48 data
-rwxr-xr-x    1 root    root     215 Jul  9 17:46 dockerfile
drwxr-xr-x    1 app      app     4096 Jul 10 17:00 node_modules
-rw-r--r--    1 app      app   659998 Jul  9 17:50 package-lock.json
-rwxr-xr-x    1 root    root     812 Jul  8 15:39 package.json
drwxr-xr-x    2 root    root     4096 Jul  8 15:39 public
drwxr-xr-x    2 root    root     4096 Jul  8 15:39 src

```

- There is no write permission for the app user!
- Because this directory is created by the root user when we run the container.



- We should create the data directory using the app user
- Since we changed the dockerfile we need to rebuild the image



The screenshot shows a VS Code editor with three tabs: `.dockerignore U`, `dockerfile U`, and `README.md M`. The `dockerfile` tab is active, displaying the following content:

```
dockerfile > ...
1 FROM node:18.14-alpine
2 RUN addgroup app && adduser -S -G app app
3 USER app
4 WORKDIR /app
5 RUN mkdir data
6 COPY package*.json .
7 RUN npm install
8 COPY . .
9 ENV API_URL=http://api.myapp.com/
10 EXPOSE 3000
11 CMD ["npm", "run", "start"]
```

The line `5 RUN mkdir data` is highlighted with a red box.

Below the editor, the `TERMINAL` tab is active, showing the following output:

```
=> => exporting layers
=> => writing image sha256:adc72cc849f27decd07d2f49e2dd4894cf09cf4795a
=> => naming to docker.io/mbabaei/react-app:3
```

Below the terminal output, there is a section titled **What's Next?** with the text: "View summary of image vulnerabilities and recommendations → [docker scan](#)".

At the bottom of the terminal, the command `PS C:\Users\drbab\OneDrive - McGill University\@Courses\ECSE 437 Software\sample-react-app\react-app> docker build -t mbabaei/react-app:3 .` is shown, with `docker build -t mbabaei/react-app:3 .` highlighted by a red box.

This time data directory already exists in the /app owned by the user app

```
PS C:\Users\drbab> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
f29056842e01   react-app:latest "docker-entrypoint.s..." 13 minutes ago Up 13 minutes  0.0.0.0:4000->3000/tcp    Accessible_Container_v1
16d85e941b39   react-app:latest "docker-entrypoint.s..." About an hour ago Up About an hour  3000/tcp                  quizzical_davinci
PS C:\Users\drbab> docker rm -f Accessible_Container_v1
Accessible_Container_v1
PS C:\Users\drbab> docker run -d -p 4000:3000 --name Accessible_Container_v2 -v app-data:/app/data mbabaei/react-app:3
41ca682ccea769d3e67b422423010fdde08431a9dbe2dd198d61b92301699a0c
PS C:\Users\drbab> docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
41ca682ccea7   mbabaei/react-app:3 "docker-entrypoint.s..." 6 seconds ago  Up 5 seconds  0.0.0.0:4000->3000/tcp    Accessible_Container_v2
16d85e941b39   react-app:latest "docker-entrypoint.s..." About an hour ago Up About an hour  3000/tcp                  quizzical_davinci
```

- Let's check if we can modify the content of this directory!

```
PS C:\Users\drbab> docker exec -it Accessible_Container_v2 sh
/app $ ls
README.md          dockerfile          package-lock.json   public
data               node_modules        package.json         src
/app $ cd data
/app/data $ echo "new line added to this file!" > data.txt
/app/data $ cat data.txt
new line added to this file!
/app/data $
```

- Let's see what happens if we remove the container and then create a new container
- Do we still have access to the content of the data directory, or it is completely gone!

```
PS C:\Users\drbab> docker rm -f Accessible_Container_v2
Accessible_Container_v2
PS C:\Users\drbab> docker run -d -p 4000:3000 --name Accessible_Container_v3 -v app-data:/app/data mbabaei/react-app:3
ae17d05fe9a7063a07193a133f357513525df4ee67eaf1b2a826d62657973bad
PS C:\Users\drbab> docker exec -it Accessible_Container_v3 sh
/app $ ls
README.md          dockerfile          package-lock.json   public
data               node_modules        package.json         src
/app $ cd data/
/app/data $ ls
data.txt
/app/data $ cat data.txt
new line added to this file!
/app/data $
```

Volumes are the right way to persist data in dockerized applications, because they have different lifecycles from containers!

*We can share volumes among multiple containers 😊*

How can we use volumes to publish our application changes?

