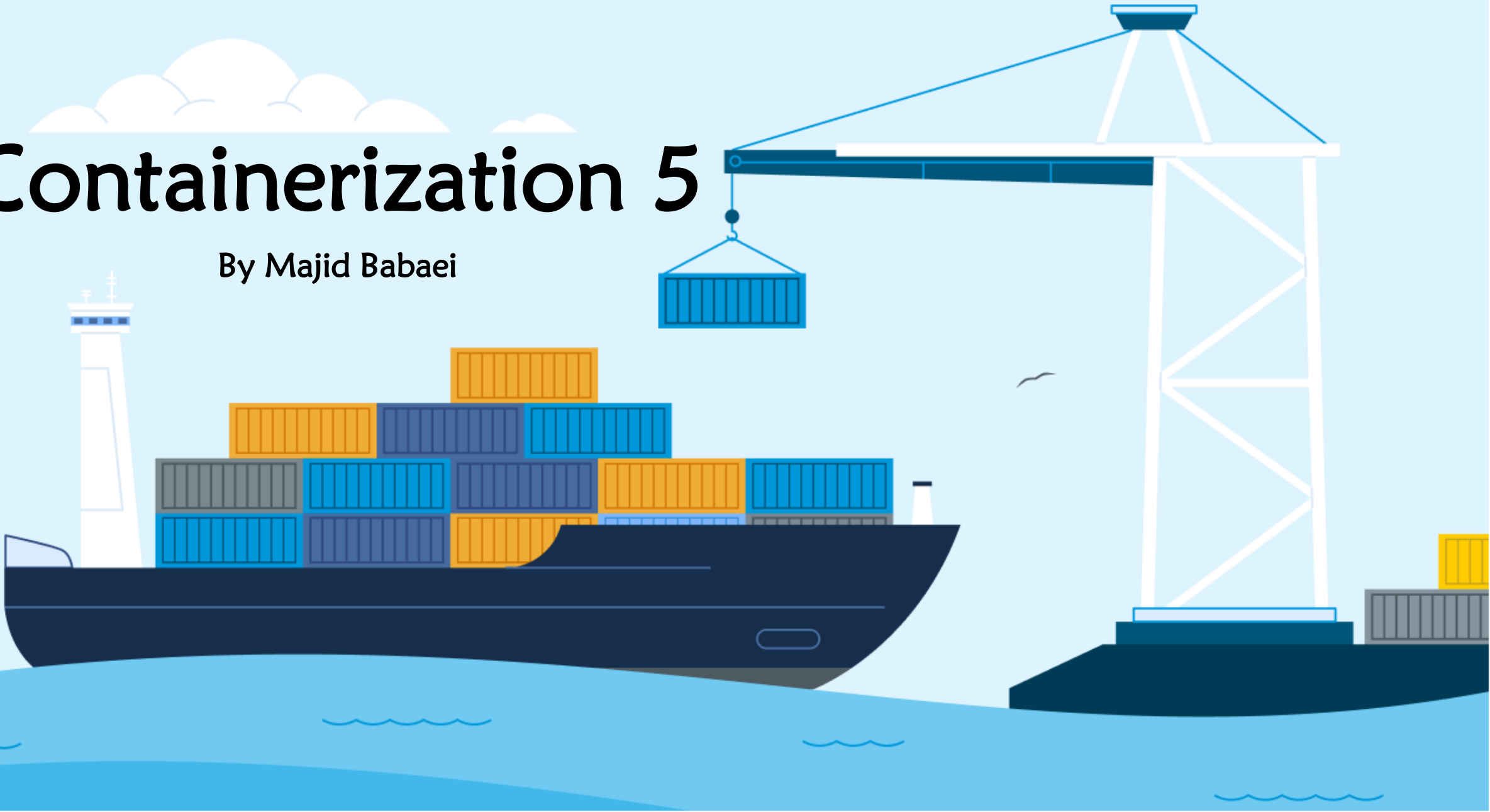






Containerization 5

By Majid Babaei



 .dockerignore U  dockerfile U X  README.md M

 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"]
```

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

Copying files between host and containers

- First let's copy a file from a container to the host
- `docker cp Accessible_Container_v3:/app/log.txt .`

```
PS C:\Users\drbab> docker ps
CONTAINER ID   IMAGE                COMMAND                  CREATED        STATUS        PORTS                NAMES
ae17d05fe9a7   mbabaei/react-app:3 "docker-entrypoint.s..." 9 minutes ago  Up 9 minutes  0.0.0.0:4000->3000/tcp Accessible_Container_v3
16d85e941b39   react-app:latest    "docker-entrypoint.s..." 2 hours ago    Up 2 hours    3000/tcp             quizzical_davinci
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 $ echo "a file in Accessible_Container_v3" > log.txt
/app $ exit
PS C:\Users\drbab> docker cp Accessible_Container_v3:/app/log.txt .
Successfully copied 2.05kB to C:\Users\drbab\.
```

Copying files between host and containers

- Now let's see how we can copy a file from host to a container
- `docker cp .\host.txt Accessible_Container_v3:/app`

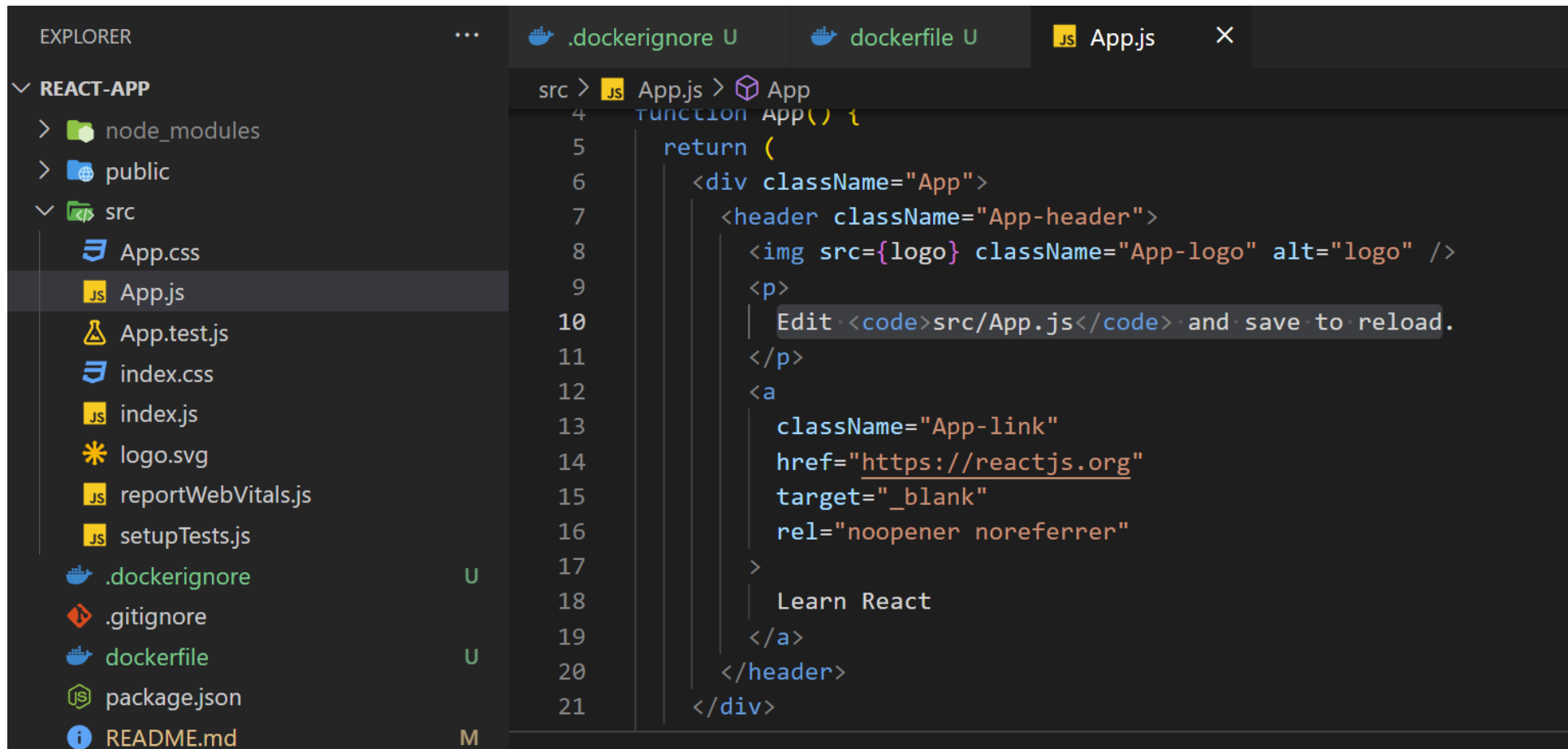
```
PS C:\Users\drbab> echo "a line from the host!" > host.txt
PS C:\Users\drbab> docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
ae17d05fe9a7	mbabaei/react-app:3	"docker-entrypoint.s..."	14 minutes ago	Up 14 minutes	0.0.0.0:4000->3000/tcp	Accessible_Container_v3
16d85e941b39	react-app:latest	"docker-entrypoint.s..."	2 hours ago	Up 2 hours	3000/tcp	quizzical_davinci

```
PS C:\Users\drbab> docker cp .\host.txt Accessible_Container_v3:/app
Successfully copied 2.05kB to Accessible_Container_v3:/app
PS C:\Users\drbab> docker exec -it Accessible_Container_v3 sh
/app $ ls
README.md          dockerfile          log.txt             package-lock.json   public
data               host.txt            node_modules        package.json         src
/app $ cat host.txt
a line from the host!
/app $
```

Sharing source code with a container

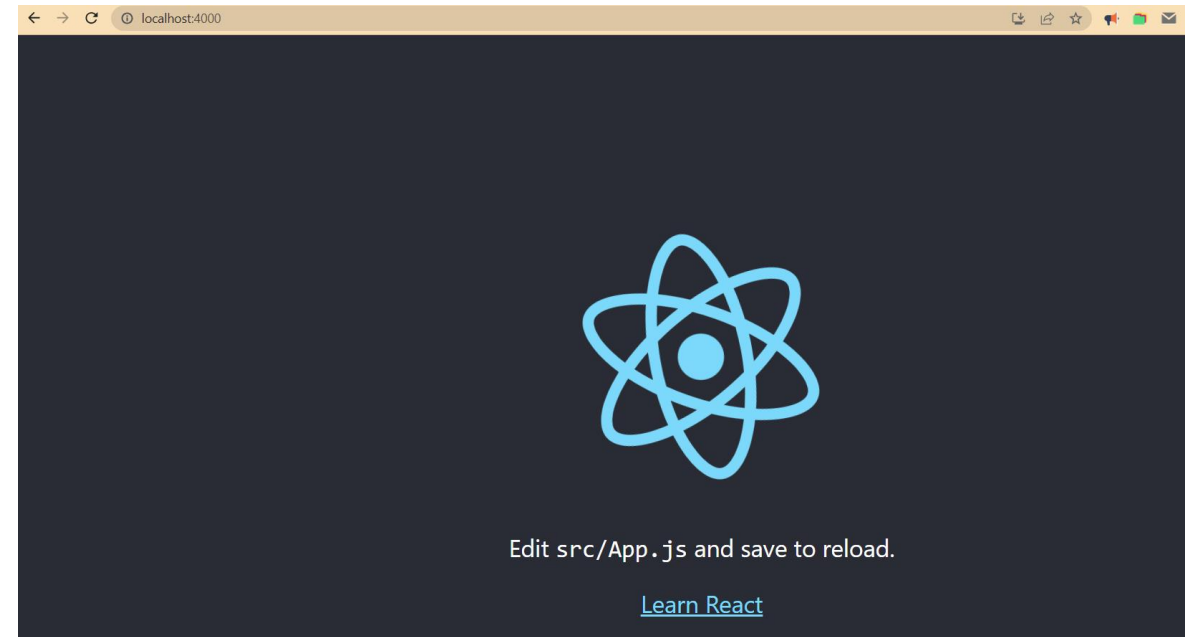
- How we can publish our application changes.
- Let's change this line in the App.js file and see what happens in the App running on port 4000.



The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left and the Editor window on the right. The Explorer sidebar displays the project structure for 'REACT-APP', including folders like 'node_modules', 'public', and 'src', and files like 'App.css', 'App.js', 'App.test.js', 'index.css', 'index.js', 'logo.svg', 'reportWebVitals.js', 'setupTests.js', '.dockerignore', '.gitignore', 'dockerfile', 'package.json', and 'README.md'. The Editor window shows the 'App.js' file, which contains a React component definition. The component has a header with a logo and a link to 'https://reactjs.org'. A text box highlights the line 'Edit <code>src/App.js</code> and save to reload.' in the code editor.

```
src > JS App.js > App
4 function App() {
5   return (
6     <div className="App">
7       <header className="App-header">
8         <img src={logo} className="App-logo" alt="logo" />
9         <p>
10          Edit <code>src/App.js</code> and save to reload.
11        </p>
12        <a
13          className="App-link"
14          href="https://reactjs.org"
15          target="_blank"
16          rel="noopener noreferrer"
17        >
18          Learn React
19        </a>
20      </header>
21    </div>
```

```
.dockerignore U  dockerfile U  JS App.js M X
src > JS App.js > [default]
1  import './App.css';
2
3  function App() {
4    return (
5      <div className="App">
6        <header className="App-header">
7          <p>Hello World!</p>
8        </header>
9      </div>
10   );
11 }
12
13 export default App;
14
```



But nothing changed!

Sharing source code with a container

- For production machine: we should always build a new image!
- For development machine:
 - We don't want to rebuild a new image for a tiny change in the code!
 - Manually copying file is time consuming!
 - We can create a mapping between a code directory on the host and one in the container

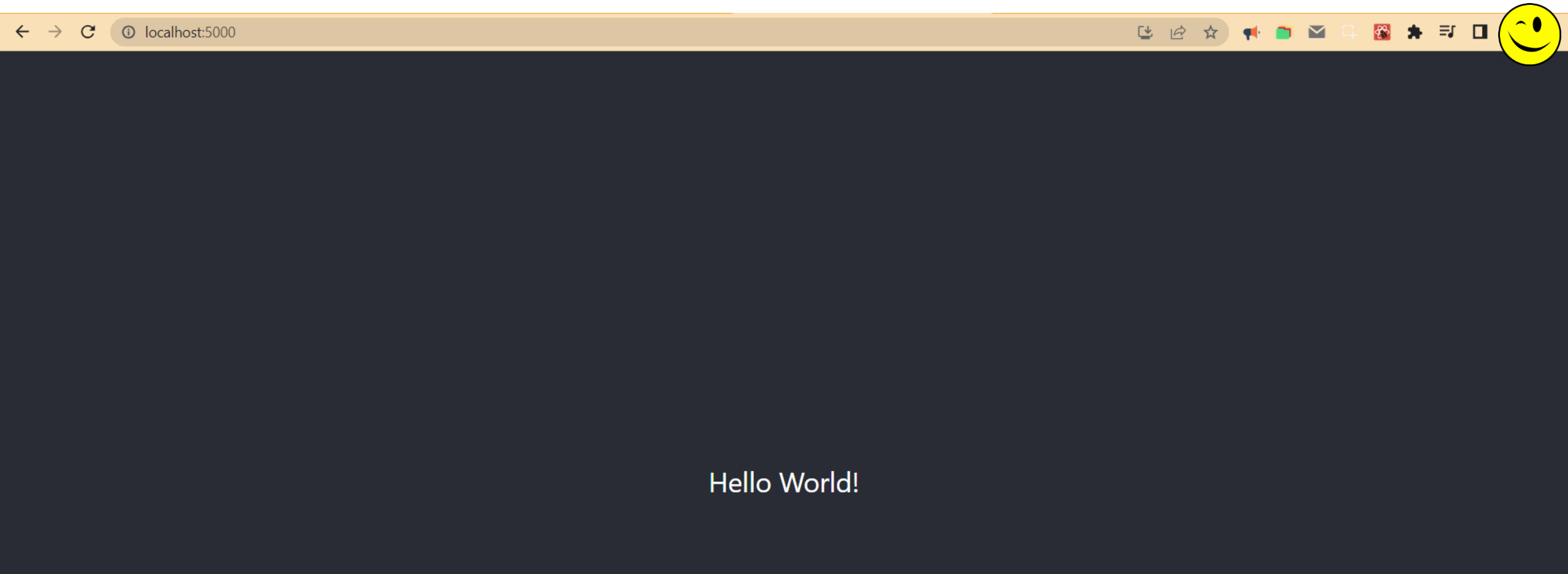
Sharing source code with a container

We can run:

- `docker run -d -p 4000:3000 --name Accessible_Container_v4 -v $(pwd):/app mbabaei/react-app:3`
- Note: `$(pwd)` is evaluated first!
- Note: for the windows machine first go to the directory where your code is located, then use “.” to refer to the current directory in the above command.
- Then we can get the live log from the running container:
- `docker logs -f Accessible_Container_v4`

Thanks to ReactJS hot
reloading feature!

Now our change on the code is immediately visible on the output!





Running Multi-container Applications

Get a clean workspace

- First, we need to remove all images and containers
- As you can see, we have several images and stopped containers
- We need to stop running containers first and then remove containers!

```
PS C:\Windows\System32> docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
mbabaei/react-app	3	adc72cc849f2	22 hours ago	570MB
mbabaei/react-app	2	505d4ff640ae	39 hours ago	570MB
mbabaei/react-app	1	8a72cc28ca3e	45 hours ago	570MB
react-app	latest	8a72cc28ca3e	45 hours ago	570MB
alpine	latest	c1aabb73d233	3 weeks ago	7.33MB
ubuntu	latest	99284ca6cea0	5 weeks ago	77.8MB

```
PS C:\Windows\System32> docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
62d0c1339597	mbabaei/react-app:3	"docker-entrypoint.s..."	21 hours ago	Exited (0) About a minute ago	0.0.0.0:5000->3000/tcp	Accessible_Container_v6
16d85e941b39	react-app:latest	"docker-entrypoint.s..."	23 hours ago	Exited (0) About a minute ago	3000/tcp	quizzical_davinci
7b2d15ac065d	react-app:latest	"docker-entrypoint.s..."	23 hours ago	Exited (1) 23 hours ago		friendly_leavitt
abfa40d9e5bd	react-app:latest	"docker-entrypoint.s..."	23 hours ago	Exited (1) 23 hours ago		funny_black
bcbf13743e237	react-app:latest	"docker-entrypoint.s..."	23 hours ago	Exited (1) 23 hours ago		nifty_ishizaka
c0f84659ef0e	ubuntu	"/bin/bash"	3 days ago	Exited (130) 22 hours ago		zen_feynman
de2e27a6293e	ubuntu	"/bin/bash"	6 days ago	Exited (0) 40 hours ago		mystifying_swirles

```
PS C:\Windows\System32>
```

Get a clean workspace

- How to get the list of all containers?

```
PS C:\Windows\System32> docker container ls
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS        NAMES
PS C:\Windows\System32> docker container ls -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS        NAMES
62d0c1339597   mbabaei/react-app:3   "docker-entrypoint.s..." 21 hours ago   Exited (0)    About a minute ago   0.0.0.0:5000->3000/tcp   Accessible_Container_v6
16d85e941b39   react-app:latest      "docker-entrypoint.s..." 23 hours ago   Exited (0)    About a minute ago   3000/tcp                quizzical_davinci
7b2d15ac065d   react-app:latest      "docker-entrypoint.s..." 23 hours ago   Exited (1)    23 hours ago                friendly_leavitt
abfa40d9e5bd   react-app:latest      "docker-entrypoint.s..." 23 hours ago   Exited (1)    23 hours ago                funny_black
bcf13743e237   react-app:latest      "docker-entrypoint.s..." 23 hours ago   Exited (1)    23 hours ago                nifty_ishizaka
c0f84659ef0e   ubuntu            "/bin/bash"              3 days ago     Exited (130)  22 hours ago                zen_feynman
de2e27a6293e   ubuntu            "/bin/bash"              6 days ago     Exited (0)    40 hours ago                mystifying_swirles
PS C:\Windows\System32> docker container ls -aq
62d0c1339597
16d85e941b39
7b2d15ac065d
abfa40d9e5bd
bcf13743e237
c0f84659ef0e
de2e27a6293e
```

- `docker container ls -aq`
- We can send this list as an argument to the remove command!

Get a clean workspace

- The list will be empty!

```
PS C:\Windows\System32> docker container rm -f $(docker container ls -aq)
62d0c1339597
16d85e941b39
7b2d15ac065d
abfa40d9e5bd
bcf13743e237
c0f84659ef0e
de2e27a6293e
PS C:\Windows\System32> docker container ls -aq
PS C:\Windows\System32>
```

- By the same token we can remove all images

docker image rm -f \$(docker image ls -aq)

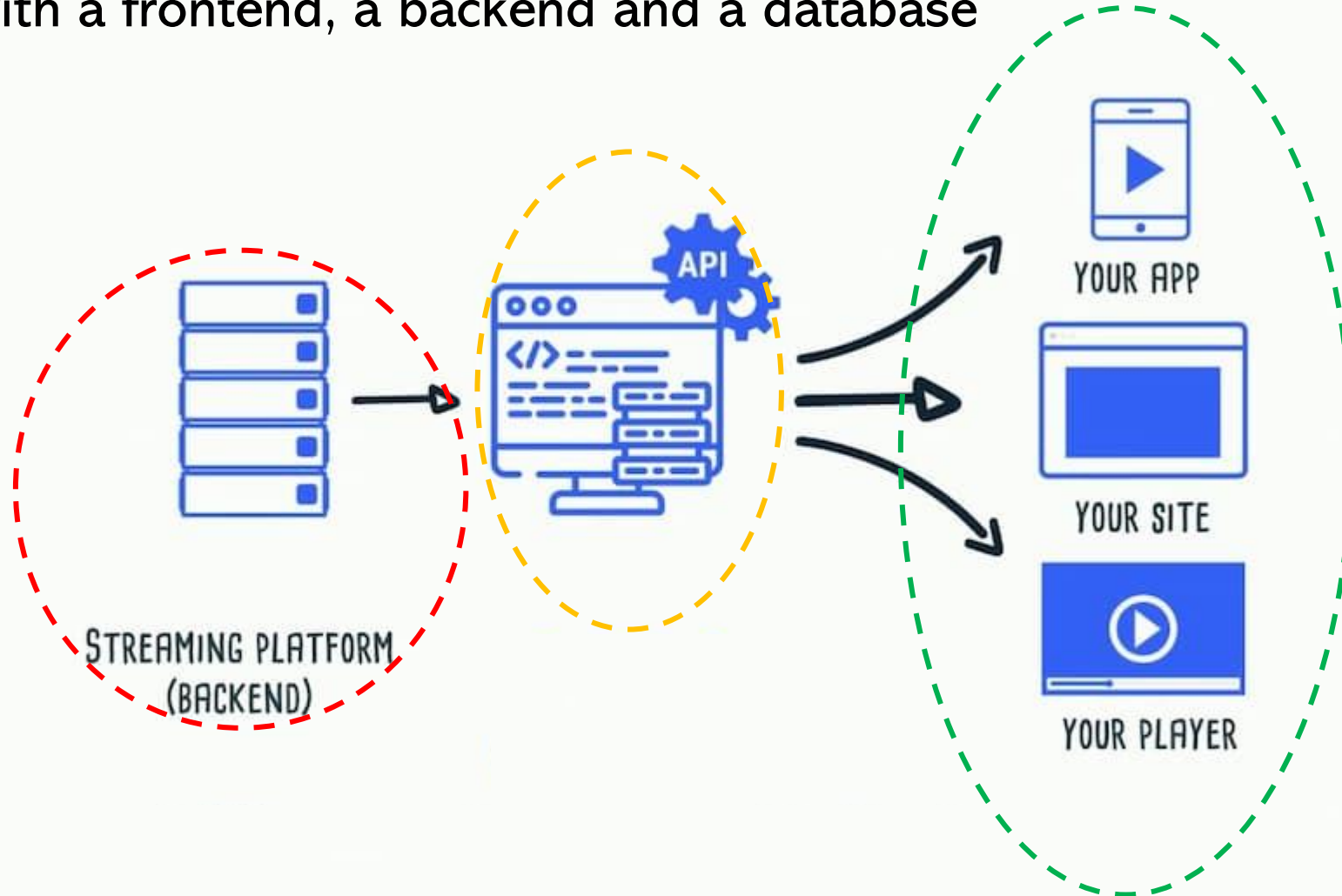
```
PS C:\Windows\System32> docker image rm -f $(docker image ls -aq)
Untagged: mbabaei/react-app:3
Deleted: sha256:adc72cc849f27decd07d2f49e2dd4894cf09cf4795afa42b92139501bd5b65bd
Untagged: mbabaei/react-app:2
Untagged: mbabaei/react-app@sha256:0b3e50bddcf7739796c665cee51340aae10d67776e6904032ffdf35225c132f
Deleted: sha256:505d4ff640ae22c1dd66670b7c07b768757d6afa87811f2eab949f410c87c97c
Untagged: react-app:latest
Untagged: mbabaei/react-app:1
Untagged: mbabaei/react-app@sha256:08d3efbbd9d643e278cd7b0c176436d0a9b282e3f9b714e30225338f5e14dece
Deleted: sha256:8a72cc28ca3e807046e99243b1dd83280ff64454761561430733bd81b9409bab
Untagged: alpine:latest
Untagged: alpine@sha256:82d1e9d7ed48a7523bdebc18cf6290bdb97b82302a8a9c27d4fe885949ea94d1
Deleted: sha256:c1aabb73d2339c5ebaa3681de2e9d9c18d57485045a4e311d9f8004bec208d67
Deleted: sha256:78a822fe2a2d2c84f3de4a403188c45f623017d6a4521d23047c9fbb0801794c
Untagged: ubuntu:latest
Untagged: ubuntu@sha256:6120be6a2b7ce665d0cbddc3ce6eae60fe94637c6a66985312d1f02f63cc0bcd
Deleted: sha256:99284ca6cea039c7784d1414608c6e846dd56830c2a13e1341be681c3ffcc8ac
Deleted: sha256:cdd7c73923174e45ea648d66996665c288e1b17a0f45efdbeca860f6dafdf731
```

Get a clean workspace

```
PS C:\Windows\System32> docker images
REPOSITORY    TAG        IMAGE ID      CREATED      SIZE
PS C:\Windows\System32> docker ps -a
CONTAINER ID   IMAGE      COMMAND      CREATED      STATUS      PORTS      NAMES
PS C:\Windows\System32>
```

A sample application

- An application with a frontend, a backend and a database



Frontend



Web App

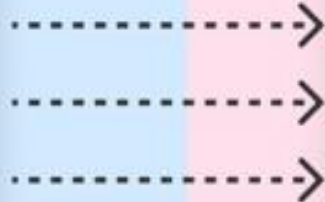
Backend

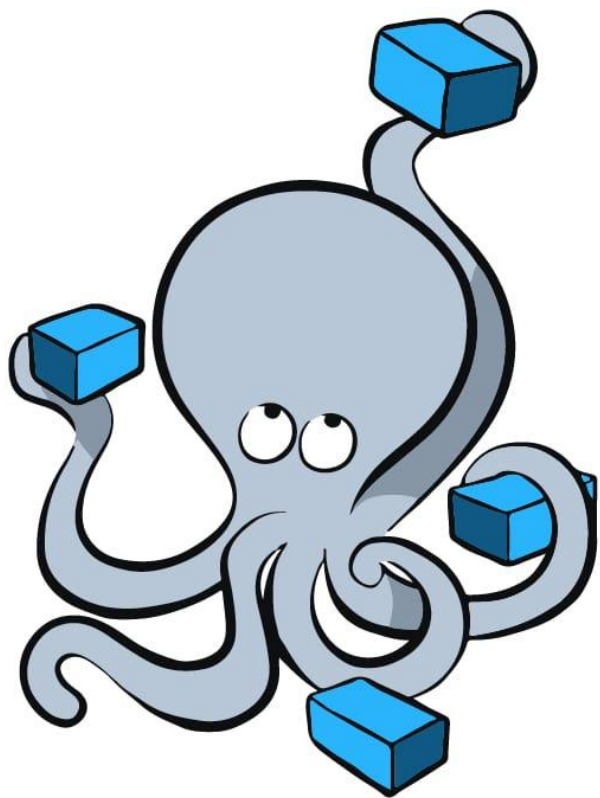


Server



Database





docker

Compose

<https://docs.docker.com/compose/install/>

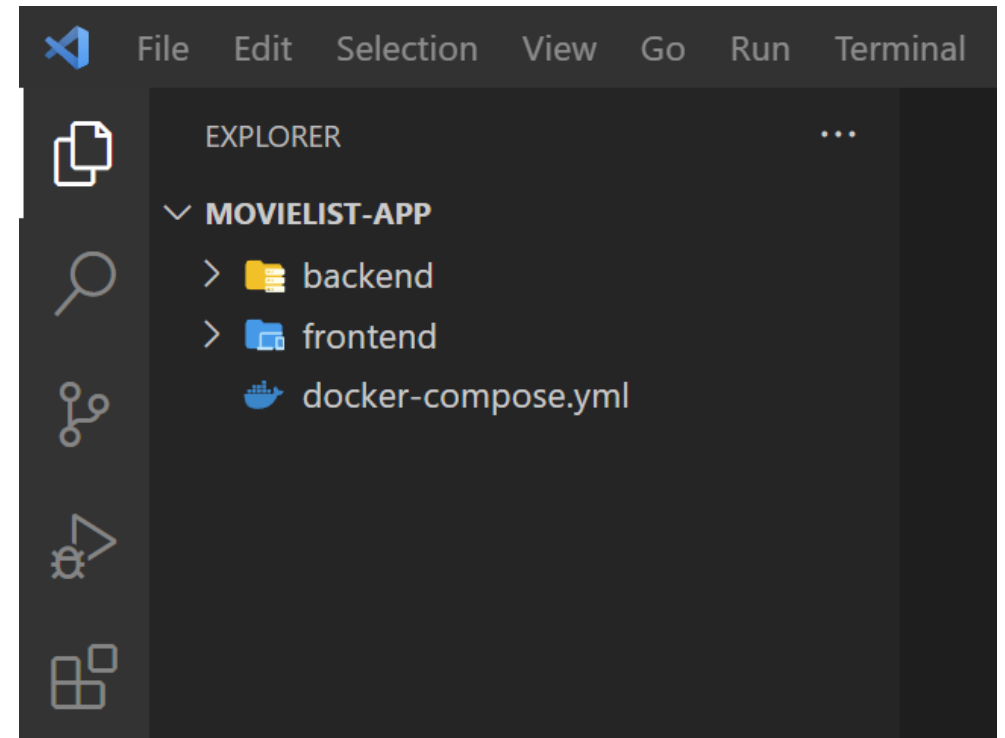
Check your docker-compose version

- *Docker-compose* is a tool built on top of docker engine
- It makes it very easy to start an application with multiple containers
- For example, an application with a container for its frontend, backend and database
- If you use windows or Mac machine you don't need to do any thing else!
- But if you use a Linux machine you need to install it from:
[*https://docs.docker.com/compose/install/*](https://docs.docker.com/compose/install/)

```
PS C:\Windows\System32> docker-compose --version
Docker Compose version v2.19.0
```

A sample application

- Inside movieList-app you can find this folder structure
- Backend: a basic node project, starts a webserver on port 3000
- Frontend: a react app that talks to the backend
- How would you run this application?
- Go to each directory
- Install all dependencies
- Running multiple terminals
- Install and run the database on your machine
- Populate the database manually
- Check if everything works as you expect!
- So many steps we need to follow, after getting the code from github.

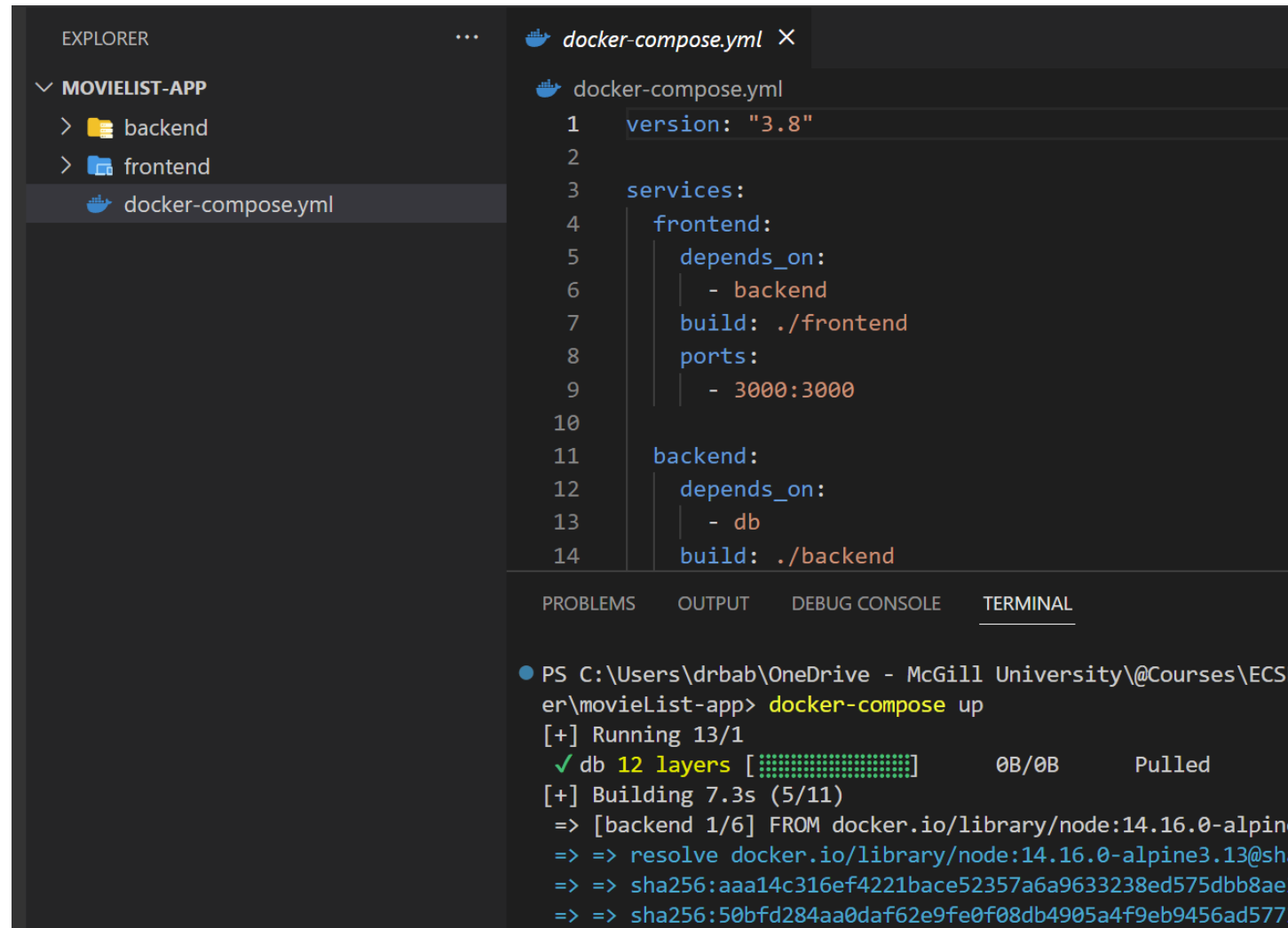


Or use docker-compose!

- We can simple do everything by just running the docker-compose
- Let docker handle rest!

- All you need to do is:
docker-compose up

- Download and run mongodb
- Install all the dependencies for our frontend and backend
- Starts the webserver
- Runs the automated tests
- Populate the database with data



The screenshot shows a Visual Studio Code editor with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'MOVIELIST-APP' with subfolders 'backend' and 'frontend', and a file 'docker-compose.yml'. The code editor shows the content of 'docker-compose.yml'.

```
version: "3.8"

services:
  frontend:
    depends_on:
      - backend
    build: ./frontend
    ports:
      - 3000:3000
  backend:
    depends_on:
      - db
    build: ./backend
```

Below the code editor, the 'TERMINAL' tab is active, showing the output of the command 'docker-compose up'.

```
PS C:\Users\drbab\OneDrive - McGill University\@Courses\ECS
er\movieList-app> docker-compose up
[+] Running 13/1
✓ db 12 layers [████████████████████████████████████████] 0B/0B Pulled
[+] Building 7.3s (5/11)
=> [backend 1/6] FROM docker.io/library/node:14.16.0-alpin
=> => resolve docker.io/library/node:14.16.0-alpine3.13@sh
=> => sha256:aaa14c316ef4221bace52357a6a9633238ed575dbb8ae
=> => sha256:50bfd284aa0daf62e9fe0f08db4905a4f9eb9456ad577
```

← → ↻ ⓘ localhost:3000

Add a new movie...

Avatar



Star Wars



Terminator



Titanic



JSON vs. YAML

- Json is a human readable language to representing data
- We can have an object or an array : {}
Let's say we want to represent a course
- A course can have properties: *name, price, etc.*
- In the object we can add one or more key-value pairs

```
{..} data.json > ...
1  {
2      "name": "ECSE437-Software Delivery",
3      "department": "ECE",
4      "tags": ["software engineering", "DevOps"],
5      "instructor": {
6          "first_name": "Majid",
7          "last_name": "Babaei"
8      },
9      "location": {
10         "building": "Rutherford Physics",
11         "class": "118",
12         "institute": "Mcgill University",
13         "city": "Montreal"
14     }
15 }
```

JSON vs. YAML

- Yaml is used for representing data
- It is less clutter than JSON
- It is easier to read!
- Add ---
- Use indentation to represent hierarchy
- Remove all quotations
- No need to use commas to sperate key-value pairs
- Remove [], instead use -

```
data.yaml
1  ---
2  name: ECSE437-Software Delivery
3  department: ECE
4  tags:
5    - software engineering
6    - DevOps
7  instructor:
8    first_name: Majid
9    last_name: Babaei
10 location:
11   building: Rutherford Physics
12   class: 118
13   institute: McGill University
14   city: Montreal
```

JSON vs. YAML

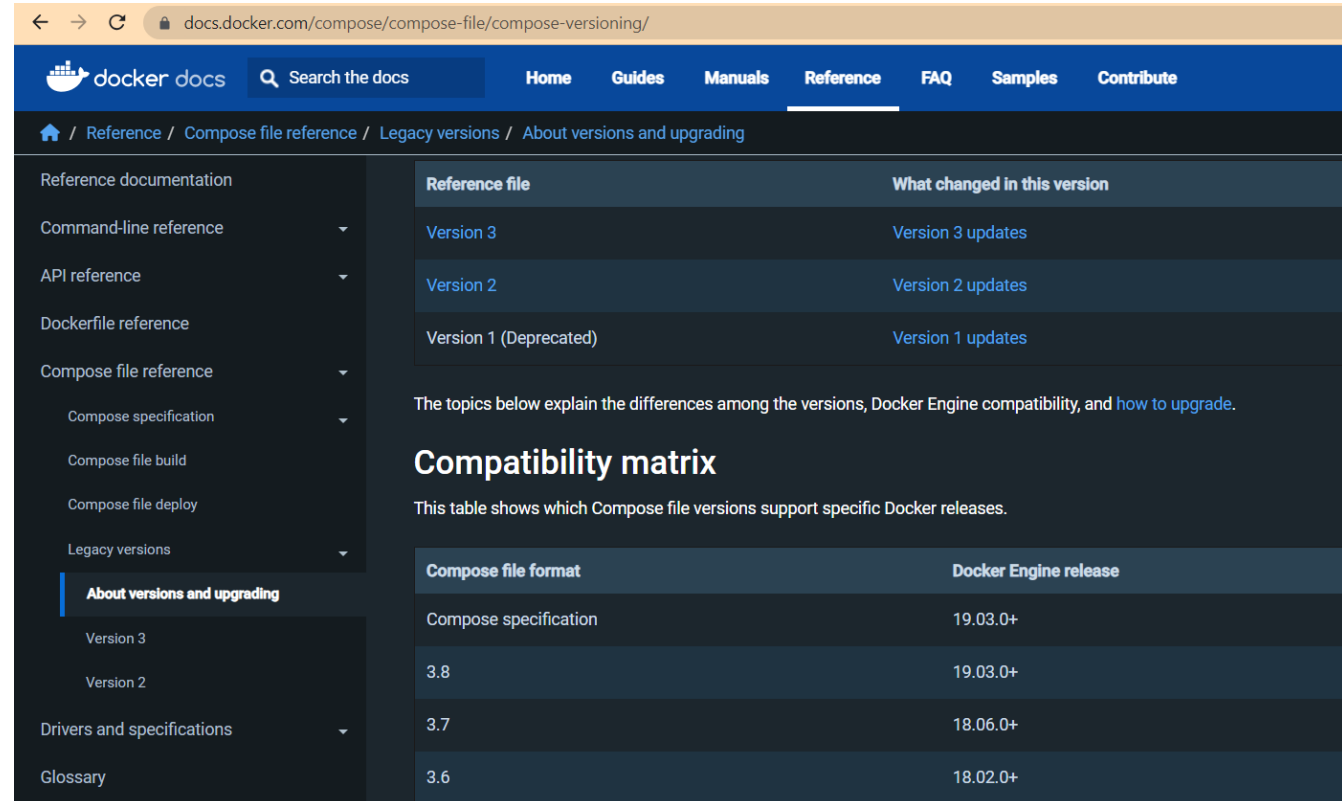
- Parsing JSON is faster than YAML
- YAML is more readable than JSON
- Use YAML for configuration files
- Use JSON for exchanging data between computers

```
data.yaml
1  ---
2  name: ECSE437-Software Delivery
3  department: ECE
4  tags:
5    - software engineering
6    - DevOps
7  instructor:
8    first_name: Majid
9    last_name: Babaei
10 location:
11   building: Rutherford Physics
12   class: 118
13   institute: McGill University
14   city: Montreal
```

```
{.} data.json > ...
1  {
2    "name": "ECSE437-Software Delivery",
3    "department": "ECE",
4    "tags": ["software engineering", "DevOps"],
5    "instructor": {
6      "first_name": "Majid",
7      "last_name": "Babaei"
8    },
9    "location": {
10     "building": "Rutherford Physics",
11     "class": "118",
12     "institute": "McGill University",
13     "city": "Montreal"
14   }
15 }
```


Create a compose file

- Create a file name `docker-compose.yaml`
- First specify the version property
 - <https://docs.docker.com/compose/compose-file/compose-versioning/>
- Define the services:
 - Frontend, Backend, etc.
 - Each service has its own dockerfile
 - We can pull an image from hub
- Tell docker how to build images for services



The screenshot shows the Docker documentation page for Compose file versioning. The page is titled "docs.docker.com/compose/compose-file/compose-versioning/" and features a navigation bar with links to Home, Guides, Manuals, Reference, FAQ, Samples, and Contribute. The left sidebar contains a list of reference documentation topics, with "About versions and upgrading" selected. The main content area is divided into two sections: "Reference file" and "Compatibility matrix".

Reference file	What changed in this version
Version 3	Version 3 updates
Version 2	Version 2 updates
Version 1 (Deprecated)	Version 1 updates

The topics below explain the differences among the versions, Docker Engine compatibility, and [how to upgrade](#).

Compatibility matrix

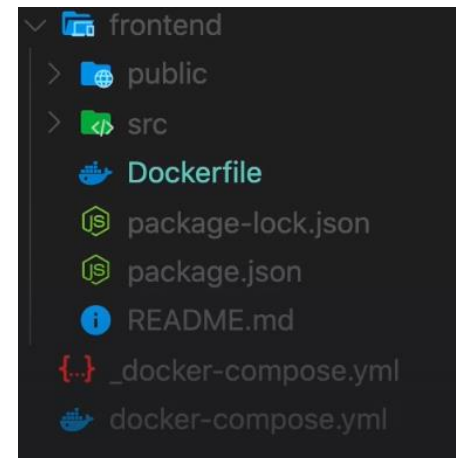
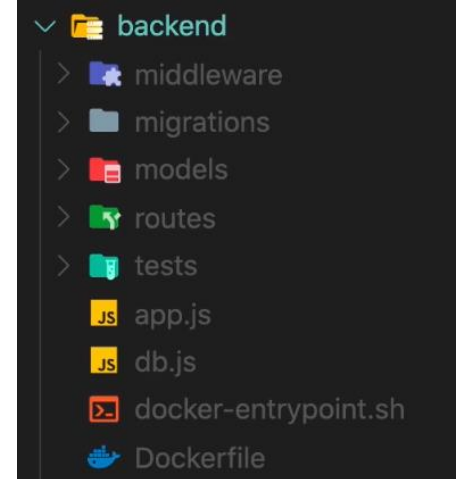
This table shows which Compose file versions support specific Docker releases.

Compose file format	Docker Engine release
Compose specification	19.03.0+
3.8	19.03.0+
3.7	18.06.0+
3.6	18.02.0+

Create a compose file

```
docker-compose.yml


1  version: "3.8"
2  services:
3      web:
4          build: ./frontend
5      api:
6          build: ./backend
7      db:
8          image: mongo:4.0-xenial
```



```
PS C:\Users\drbab> docker run -d -p 4000:3000 --name Accessible_Container_v2 -v app-data:/app/data mbabaei/react-app:3
41ca682ccea769d3e67b422423010fdde08431a9dbe2dd198d61b92301699a0c
```


frontend >  Dockerfile > ...

```
1 FROM node:14.16.0-alpine3.13
2
3 RUN addgroup app && adduser -S -G app app
4 USER app
5
6 WORKDIR /app
7 COPY package*.json ./
8 RUN npm install
9 COPY . .
10
11 EXPOSE 3000
12
13 CMD ["npm", "start"]
```

backend >  Dockerfile > ...

```
1 FROM node:14.16.0-alpine3.13
2
3 RUN addgroup app && adduser -S -G app app
4 USER app
5
6 WORKDIR /app
7 COPY package*.json ./
8 RUN npm install
9 COPY . .
10
11 EXPOSE 3001
12
13 CMD ["npm", "start"]
```

Create a compose file

```
 docker-compose.yml
1  version: "3.8"
2  ∨ services:
3  ∨    web:
4      build: ./frontend
5  ∨    ports:
6      - 3000:3000
7  ∨    api:
8      build: ./backend
9  ∨    ports:
10     - 3001:3001
11  ∨    db:
12     image: mongo:4.0-xenial
13  ∨    ports:
14     - 27017:27017
```

Create a compose file

- What other properties you can add:
- <https://docs.docker.com/compose/compose-file/compose-file-v3/>
- Build
- Ports
- Volume
- Environment

Compose file version 3 reference

Important

From July 2023 Compose V1 stopped receiving updates. It's also no longer available in new releases of Docker Desktop.

Compose V2 is included with all currently supported versions of Docker Desktop. For more information, see [Migrate to Compose V2](#).

Reference and guidelines

These topics describe version 3 of the Compose file format.

Compose and Docker compatibility matrix

There are several versions of the Compose file format – 1, 2, 2.x, and 3.x. The table below is a quick look. For full details on what each version includes and how to upgrade, see [About versions and upgrading](#).

This table shows which Compose file versions support specific Docker releases.

Compose file format	Docker Engine release
Compose specification	19.03.0+
3.8	19.03.0+
3.7	18.06.0+
3.6	18.02.0+
3.5	17.12.0+
3.4	17.09.0+

Reference and guidelines

Compose and Docker compatibility matrix

Compose file structure and examples

Service configuration reference

build

context

dockerfile

args

cache_from

labels

network

shm_size

target

cap_add, cap_drop

cgroup_parent

command

configs

Short syntax

Long syntax

container_name

credential_spec

Example gMSA configuration

depends_on

deploy

endpoint_mode

labels

Create a compose file

- Our api needs to know where database is!
- We can define an environment variable and define how to connect to the db
- Note: we can have multiple env. Variables so we should use a list
- `DB_URL="mongodb://db/movieList"`
- To connect to a mongodb it has to start with `"mongodb://"`
- Then we should tell the name of the host which is equal to the name we use for the database, in this case `"db"`
- Finally, the name of the database: `"movieList"`

```
api:
  build: ./backend
  ports:
    - 3001:3001
  environment:
    - DB_URL="mongodb://db/movieList"
```

Create a compose file

- We need to add a volume
- We don't want the mongodb write data to the temporally filesystem of the container
- To map data to a volume we need to know where mogodb stores its data in the container. We should use the documentation in: https://hub.docker.com/_/mongo

The Docker documentation is a good starting point for understanding the different storage options and variations, and there are multiple blogs and forum postings that discuss and give advice in this area. We will simply show the basic procedure here for the latter option above:

1. Create a data directory on a suitable volume on your host system, e.g. `/my/own/datadir`.
2. Start your `mongo` container like this:

```
$ docker run --name some-mongo -v /my/own/datadir:/data/db -d mongo
```

- mongodb stores its data inside: `/data/db`
- We need to define the volume in the compose file

Create a compose file

```
🚢 docker-compose.yml
1  version: "3.8"
2  services:
3    web:
4      build: ./frontend
5      ports:
6        - 3000:3000
7    api:
8      build: ./backend
9      ports:
10       - 3001:3001
11      environment:
12        - DB_URL="mongodb://db/movieList"
13    db:
14      image: mongo:4.0-xenial
15      ports:
16        - 27017:27017
17      volumes:
18        - movieList:data/db
19  volumes:
20    movieList:
```


Create a compose file

- Everything we have done with docker engine, e.g., building images, listing, starting a container, we can do using a docker-compose

```
C:\Users\Toronto>docker-compose
```

```
Usage:  docker compose [OPTIONS] COMMAND
```

```
Define and run multi-container applications with Docker.
```

```
Options:
```

<code>--ansi</code> string	Control when to print ANSI control characters ("never" "always" "auto") (default "auto")
<code>--compatibility</code>	Run compose in backward compatibility mode
<code>--dry-run</code>	Execute command in dry run mode
<code>--env-file</code> stringArray	Specify an alternate environment file.
<code>-f, --file</code> stringArray	Compose configuration files
<code>--parallel</code> int	Control max parallelism, -1 for unlimited (default -1)
<code>--profile</code> stringArray	Specify a profile to enable
<code>--progress</code> string	Set type of progress output (auto, tty, plain, quiet) (default "auto")
<code>--project-directory</code> string	Specify an alternate working directory (default: the path of the, first specified, Compose file)
<code>-p, --project-name</code> string	Project name

```
Commands:
```

<code>build</code>	Build or rebuild services
<code>config</code>	Parse, resolve and render compose file in canonical format
<code>cp</code>	Copy files/folders between a service container and the local filesystem
<code>create</code>	Creates containers for a service.
<code>down</code>	Stop and remove containers, networks
<code>events</code>	Receive real time events from containers.
<code>exec</code>	Execute a command in a running container.
<code>images</code>	List images used by the created containers
<code>kill</code>	Force stop service containers.
<code>logs</code>	View output from containers
<code>ls</code>	List running compose projects
<code>pause</code>	Pause services
<code>port</code>	Print the public port for a port binding.
<code>ps</code>	List containers
<code>pull</code>	Pull service images
<code>push</code>	Push service images
<code>restart</code>	Restart service containers
<code>rm</code>	Removes stopped service containers
<code>run</code>	Run a one-off command on a service.
<code>scale</code>	Scale services
<code>start</code>	Start services
<code>stop</code>	Stop services
<code>top</code>	Display the running processes
<code>unpause</code>	Unpause services
<code>up</code>	Create and start containers
<code>version</code>	Show the Docker Compose version information

- The difference is, these commands are applied in our app as a whole!
- They will impact multiple services.

```
C:\Users\Toronto>docker-compose build --help
```

```
Usage:  docker compose build [OPTIONS] [SERVICE...]
```

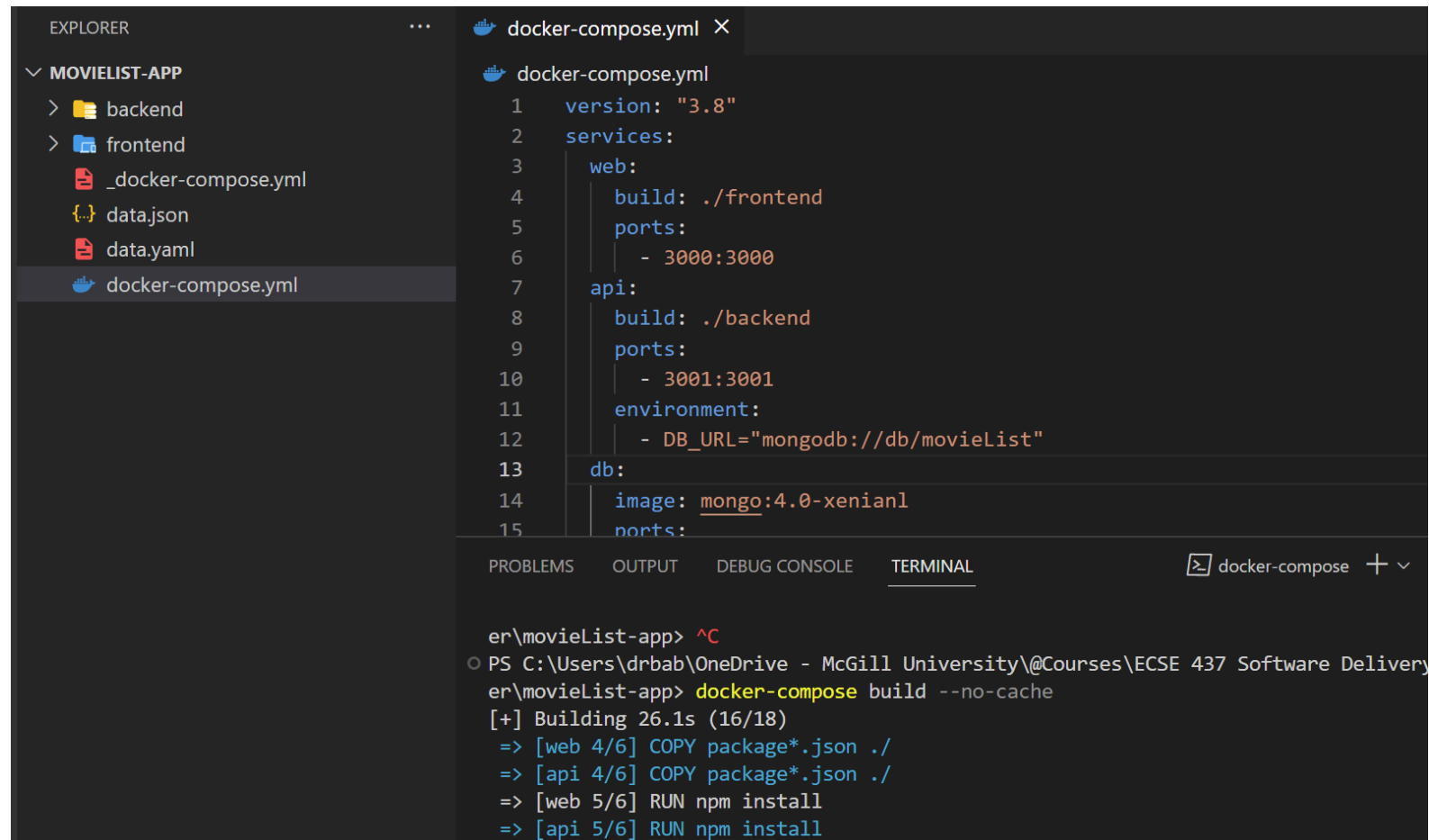
```
Build or rebuild services
```

```
Options:
```

<code>--build-arg</code> <code>stringArray</code>	Set build-time variables for services.
<code>--builder</code> <code>string</code>	Set builder to use.
<code>--dry-run</code>	Execute command in dry run mode
<code>-m, --memory</code> <code>bytes</code>	Set memory limit for the build container. Not supported by BuildKit.
<code>--no-cache</code>	Do not use cache when building the image
<code>--pull</code>	Always attempt to pull a newer version of the image.
<code>--push</code>	Push service images.
<code>-q, --quiet</code>	Don't print anything to STDOUT
<code>--ssh</code> <code>string</code>	Set SSH authentications used when building service images. (use 'default' for using your default SSH Agent)

Build the application

- docker-compose build



The screenshot shows the Visual Studio Code interface. On the left, the Explorer pane displays the project structure for 'MOVIELIST-APP', including 'backend', 'frontend', '_docker-compose.yml', 'data.json', 'data.yaml', and 'docker-compose.yml'. The 'docker-compose.yml' file is selected and its content is shown in the main editor. The file defines two services: 'web' and 'api'. The 'web' service builds from the 'frontend' directory and maps port 3000 to 3000. The 'api' service builds from the 'backend' directory, maps port 3001 to 3001, and sets the 'DB_URL' environment variable to 'mongodb://db/movieList'. A 'db' service is also defined, using the 'mongo:4.0-xenian1' image. At the bottom, the Terminal pane shows the command 'docker-compose build --no-cache' being executed, with progress output for building the 'web' and 'api' services.

```
EXPLORER
MOVIELIST-APP
  backend
  frontend
  _docker-compose.yml
  data.json
  data.yaml
  docker-compose.yml

docker-compose.yml
1 version: "3.8"
2 services:
3   web:
4     build: ./frontend
5     ports:
6       - 3000:3000
7   api:
8     build: ./backend
9     ports:
10      - 3001:3001
11     environment:
12       - DB_URL="mongodb://db/movieList"
13   db:
14     image: mongo:4.0-xenian1
15     ports:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
er\movieList-app> ^C
PS C:\Users\drbab\OneDrive - McGill University\@Courses\ECSE 437 Software Delivery
er\movieList-app> docker-compose build --no-cache
[+] Building 26.1s (16/18)
=> [web 4/6] COPY package*.json ./
=> [api 4/6] COPY package*.json ./
=> [web 5/6] RUN npm install
=> [api 5/6] RUN npm install
```

Start and stop an application

- `docker-compose up --build`
- `docker-compose up -d`
- Note: If you get the images before running them you don't see the db image
- Note: you can start multiple container from the same image (*nice feature for the scalability and availability of your services*)
- To stop the app: `docker-compose down`

```
PS C:\Windows\System32> docker images
REPOSITORY          TAG          IMAGE ID      CREATED        SIZE
movielist-app-web    latest       abd550312a9c  19 seconds ago 299MB
movielist-app-api    latest       614d3a8eda9d  33 seconds ago 184MB
PS C:\Windows\System32> docker images
REPOSITORY          TAG          IMAGE ID      CREATED        SIZE
movielist-app-web    latest       abd550312a9c  57 seconds ago 299MB
movielist-app-api    latest       614d3a8eda9d  About a minute ago 184MB
mongo                4.0.16-xenial 4ca2473194af  3 years ago    418MB
PS C:\Windows\System32> docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED        STATUS        PORTS                    NAMES
4a9db75f5835   movielist-app-web     "docker-entrypoint.s..." 5 minutes ago  Up 5 minutes  0.0.0.0:3000->3000/tcp    movielist-app-web-1
4d87e6991f49   mongo:4.0.16-xenial   "docker-entrypoint.s..." 5 minutes ago  Up 5 minutes  0.0.0.0:27017->27017/tcp  movielist-app-db-1
edad53966f5f   movielist-app-api     "docker-entrypoint.s..." 5 minutes ago  Up 5 minutes  0.0.0.0:3001->3001/tcp    movielist-app-api-1
PS C:\Windows\System32>
```

C:\Users\Toronto\Desktop\tmp\movielist-app>docker-compose up

[+] Running 13/13

db 12 layers [000000000000] 0B/0B Pulled

58690f9b18fc Pull complete
b51569e7c507 Pull complete
da8ef40b9eca Pull complete
fb15d46c38dc Pull complete
a0dc15b16822 Pull complete
b7a3e92f19af Pull complete
ed4a7b863fa1 Pull complete
a58b030aa8e4 Pull complete
6aa1ba699846 Pull complete
ebc52c729dca Pull complete
52e8c440d4d6 Pull complete
22b97876323d Pull complete

[+] Building 164.6s (16/18)

=> [backend internal] load build definition from Dockerfile

=> => transferring dockerfile: 217B

=> [backend internal] load .dockerignore

=> => transferring context: 53B

=> [frontend internal] load metadata for docker.io/library/node:14.16.0-alpine3.13

=> [backend auth] library/node:pull token for registry-1.docker.io

=> [frontend 1/6] FROM docker.io/library/node:14.16.0-alpine3.13@sha256:2c51dc462a02f15621e7486774d36d048a27225d581374b002b8477a79a59b4b

=> => resolve docker.io/library/node:14.16.0-alpine3.13@sha256:2c51dc462a02f15621e7486774d36d048a27225d581374b002b8477a79a59b4b

=> => sha256:aaa14c316ef4221bace52357a6a9633238ed575dbb8ae372df959d7939ce6366 1.16kB / 1.16kB

=> => sha256:50bfd284aa0daf62e9fe0f08db4905a4f9eb9456ad5773e259ae18d31ec44f6e 6.73kB / 6.73kB

=> => sha256:ca3cd42a7c9525f6ce3d64c1a70982613a8235f0cc057ec9244052921853ef15 2.81MB / 2.81MB

=> => sha256:2c51dc462a02f15621e7486774d36d048a27225d581374b002b8477a79a59b4b 1.43kB / 1.43kB

=> => sha256:cf8dc363e30fa6d6a24fafe0c593ea0447c0fbee2ec3f1a7e6c68fa8836d1556c 35.91MB / 35.91MB

=> => sha256:074ae49463c4f38ae0a7b325d47187e75bc59b07639d2e312d35b29aca1ee4e7 2.36MB / 2.36MB

=> => extracting sha256:ca3cd42a7c9525f6ce3d64c1a70982613a8235f0cc057ec9244052921853ef15

=> => sha256:48fd78e4b5321d188146bc79a6b38f93ad9bc0958939a8ecbde58aebd044c7b 282B / 282B

=> => extracting sha256:cf8dc363e30fa6d6a24fafe0c593ea0447c0fbee2ec3f1a7e6c68fa8836d1556c

=> => extracting sha256:074ae49463c4f38ae0a7b325d47187e75bc59b07639d2e312d35b29aca1ee4e7

=> => extracting sha256:48fd78e4b5321d188146bc79a6b38f93ad9bc0958939a8ecbde58aebd044c7b

=> [backend internal] load build context

=> => transferring context: 10.81kB

=> CACHED [frontend 2/6] RUN addgroup app && adduser -S -G app app

=> CACHED [frontend 3/6] WORKDIR /app

=> [backend 4/6] COPY package*.json ./

=> [backend 5/6] RUN npm install

=> [backend 6/6] COPY . .

=> [backend] exporting to image

=> => exporting layers

=> => writing image sha256:03663e74262fde864dc575a4fb8e1aed891ff9db927081bfa06f65b61e1afec7

=> => naming to docker.io/library/movielist-app-backend

=> [frontend internal] load build definition from Dockerfile

=> => transferring dockerfile: 216B

=> [frontend internal] load .dockerignore

=> => transferring context: 53B

=> [frontend internal] load build context

=> => transferring context: 22.67kB

=> [frontend 4/6] COPY package*.json ./

=> [frontend 5/6] RUN npm install

=> => # npm WARN deprecated @hapi/address@2.1.4: Moved to 'npm install @sideway/address'

=> => # npm WARN deprecated resolve-url@0.2.1: https://github.com/lydell/resolve-url#deprecated

=> => # npm WARN deprecated source-map-codec@1.4.8: Please use @jridgewell/source-map-codec instead

=> => # npm WARN deprecated w3c-hr-time@1.0.2: Use your platform's native performance.now() and performance.timeOrigin.

=> => # npm WARN deprecated core-js@2.6.12: core-js@<3.23.3 is no longer maintained and not recommended for usage due to the number of issues. Because of the V8 engine whims,

=> => # down up to 100x even if nothing is polyfilled. Some versions have web compatibility issues. Please, upgrade your dependencies to the actual version of core-js.


```
[+] Running 5/4
  0 Network movielist-app_default      Created
  0 Volume "movielist-app_movielist"   Created
  0 Container movielist-app-db-1        Created
  0 Container movielist-app-backend-1   Created
  0 Container movielist-app-frontend-1  Created
Attaching to movielist-app-backend-1, movielist-app-db-1, movielist-app-frontend-1
movielist-app-db-1 | 2023-10-29T19:56:34.170+0000 I CONTROL [main] Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify --sslDisabledProtocols 'none'
movielist-app-db-1 | 2023-10-29T19:56:34.180+0000 I CONTROL [initandlisten] MongoDB starting : pid=1 port=27017 dbpath=/data/db 64-bit host=40ddaa57f0b4
movielist-app-db-1 | 2023-10-29T19:56:34.180+0000 I CONTROL [initandlisten] db version v4.0.28
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I CONTROL [initandlisten] git version: af1a9dc12adcfa83cc19571cb3faba26eeddac92
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.2g 1 Mar 2016
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I CONTROL [initandlisten] allocator: tcmalloc
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I CONTROL [initandlisten] modules: none
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I CONTROL [initandlisten] build environment:
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I CONTROL [initandlisten] distmod: ubuntu1604
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I CONTROL [initandlisten] distarch: x86_64
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I CONTROL [initandlisten] target_arch: x86_64
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I CONTROL [initandlisten] options: { net: { bindIpAll: true } }
movielist-app-db-1 | 2023-10-29T19:56:34.181+0000 I STORAGE [initandlisten]
movielist-app-db-1 | 2023-10-29T19:56:34.182+0000 I STORAGE [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine
movielist-app-db-1 | 2023-10-29T19:56:34.182+0000 I STORAGE [initandlisten] ** See http://dochub.mongodb.org/core/prodnotes-filesystem
movielist-app-db-1 | 2023-10-29T19:56:34.182+0000 I STORAGE [initandlisten] wiredtiger_open config: create,cache_size=5826M,cache_overflow=(file_max=0M),session_max=20000,eviction=(threads_min=4,threads_max=4),log=(enabled=true,archive=true,path=journal,compressor=snappy),file_manager=(close_idle_time=100000),statistics_log=(wait=0),verbose=(recovery_progress),
movielist-app-backend-1 | Waiting for MongoDB to start...
movielist-app-db-1 | 2023-10-29T19:56:34.863+0000 I STORAGE [initandlisten] WiredTiger message [1698609394:863177][1:0x7f040c688a80], txn-recover: Set global recovery timestamp: 0
movielist-app-db-1 | 2023-10-29T19:56:34.936+0000 I RECOVERY [initandlisten] WiredTiger recoveryTimestamp. Ts: Timestamp(0, 0)
movielist-app-db-1 | 2023-10-29T19:56:35.017+0000 I STORAGE [initandlisten] Starting to check the table logging settings for existing WiredTiger tables
movielist-app-db-1 | 2023-10-29T19:56:35.055+0000 I CONTROL [initandlisten]
movielist-app-db-1 | 2023-10-29T19:56:35.055+0000 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
movielist-app-db-1 | 2023-10-29T19:56:35.055+0000 I CONTROL [initandlisten] ** Read and write access to data and configuration is unrestricted.
movielist-app-db-1 | 2023-10-29T19:56:35.055+0000 I CONTROL [initandlisten]
movielist-app-db-1 | 2023-10-29T19:56:35.055+0000 I CONTROL [initandlisten]
movielist-app-db-1 | 2023-10-29T19:56:35.056+0000 I CONTROL [initandlisten] ** WARNING: /sys/kernel/mm/transparent_hugepage/enabled is 'always'.
movielist-app-db-1 | 2023-10-29T19:56:35.056+0000 I CONTROL [initandlisten] ** We suggest setting it to 'never'
movielist-app-db-1 | 2023-10-29T19:56:35.056+0000 I STORAGE [initandlisten] createCollection: admin.system.version with provided UUID: 8c28dca4-34a6-4a1f-9efd-e08606922040
movielist-app-db-1 | 2023-10-29T19:56:35.123+0000 I COMMAND [initandlisten] setting featureCompatibilityVersion to 4.0
movielist-app-db-1 | 2023-10-29T19:56:35.123+0000 I STORAGE [initandlisten] Finished adjusting the table logging settings for existing WiredTiger tables
movielist-app-db-1 | 2023-10-29T19:56:35.123+0000 I STORAGE [initandlisten] createCollection: local.startup_log with generated UUID: 2488c47d-43db-44e3-aa8b-4753be2dfde8
movielist-app-db-1 | 2023-10-29T19:56:35.176+0000 I FTDC [initandlisten] Initializing full-time diagnostic data capture with directory '/data/db/diagnostic.data'
movielist-app-db-1 | 2023-10-29T19:56:35.178+0000 I STORAGE [LogicalSessionCacheRefresh] createCollection: config.system.sessions with generated UUID: 39534c76-71bc-447f-99e7-7190cf1474f0
movielist-app-db-1 | 2023-10-29T19:56:35.179+0000 I NETWORK [initandlisten] waiting for connections on port 27017
movielist-app-db-1 | 2023-10-29T19:56:35.255+0000 I INDEX [LogicalSessionCacheRefresh] build index on: config.system.sessions properties: { v: 2, key: { lastUse: 1 }, name: "lsidTTLIndex", ns: "config.system.sessions" }
movielist-app-db-1 | 2023-10-29T19:56:35.255+0000 I INDEX [LogicalSessionCacheRefresh] building index using bulk method; build may temporarily use up to 500 megabytes of RAM
movielist-app-db-1 | 2023-10-29T19:56:35.256+0000 I INDEX [LogicalSessionCacheRefresh] build index done. scanned 0 total records. 0 secs
movielist-app-frontend-1 |
movielist-app-frontend-1 | > movielist-frontend@0.1.0 start /app
movielist-app-frontend-1 | > react-scripts start
movielist-app-frontend-1 |
movielist-app-db-1 | 2023-10-29T19:56:35.506+0000 I NETWORK [listener] connection accepted from 172.18.0.3:46123 #1 (1 connection now open)
movielist-app-db-1 | 2023-10-29T19:56:35.506+0000 I NETWORK [conn1] end connection 172.18.0.3:46123 (0 connections now open)
movielist-app-backend-1 | Migrating the database...
movielist-app-backend-1 |
movielist-app-backend-1 | > movielist-backend@1.0.0 db:up /app
movielist-app-backend-1 | > migrate-mongo up
movielist-app-backend-1 |
movielist-app-db-1 | 2023-10-29T19:56:37.079+0000 I NETWORK [listener] connection accepted from 172.18.0.3:41700 #2 (1 connection now open)
movielist-app-db-1 | 2023-10-29T19:56:37.090+0000 I NETWORK [conn2] received client metadata from 172.18.0.3:41700 conn2: { driver: { name: "nodejs", version: "4.17.1" }, platform: "Node.js v14.16.0,
```

Docker Networking

- Docker-compose automatically create a network and put our container inside this network. So these containers can talk to each other
- Every docker installation has 3 network: bridge, host, and none

```
PS C:\Windows\System32> docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
b3444eda49e8        bridge              bridge              local
81ba1e01870d        host                host                local
c05a7757c1cc        movielist-app_default bridge              local
b054108dbe34        none                null                local
```

- The network “movielist-app_default” contains 3 containers: web, api, and db

Docker Networking

- Let's use this network and communicate between containers
- Note: we need to login as the root user otherwise we will get a permission error

```
docker exec -it -u root movielist-app-web-1 sh
```

```
PS C:\Windows\System32> docker exec -it -u root movielist-app-web-1 sh
/app # ping api
PING api (172.18.0.2): 56 data bytes
64 bytes from 172.18.0.2: seq=0 ttl=64 time=0.265 ms
64 bytes from 172.18.0.2: seq=1 ttl=64 time=0.088 ms
64 bytes from 172.18.0.2: seq=2 ttl=64 time=0.088 ms
64 bytes from 172.18.0.2: seq=3 ttl=64 time=0.083 ms
64 bytes from 172.18.0.2: seq=4 ttl=64 time=0.094 ms
64 bytes from 172.18.0.2: seq=5 ttl=64 time=0.091 ms
64 bytes from 172.18.0.2: seq=6 ttl=64 time=0.088 ms
```


CONTAINER

DNS RESOLVER

DNS SERVER



ip of api?

DNS SERVER



CONTAINER

DNS RESOLVER



172.21.0.3



CONTAINER



172.21.0.1

CONTAINER



172.21.0.2

CONTAINER



172.21.0.3

MongoDB Compass - localhost:27017/movielist.movies

Connect Edit View Collection Help

localhost:27017 Documents movielist.movies

My Queries Databases

Search

- admin
- config
- local
- movielist
 - changelog
 - movies

movielist.movies

Documents Aggregations Schema Indexes Validation

4 DOCUMENTS 1 INDEXES

Filter Type a query: { field: 'value' } or [Generate query](#)

Explain Reset Find Options

ADD DATA EXPORT DATA

1 - 4 of 4

<code>_id: ObjectId('653eb8f56839efc5dd6ee7f2')</code> <code>title: "Avatar"</code>
<code>_id: ObjectId('653eb8f56839efc5dd6ee7f3')</code> <code>title: "Star Wars"</code>
<code>_id: ObjectId('653eb8f56839efc5dd6ee7f4')</code> <code>title: "The Godfather"</code>

<https://www.mongodb.com/try/download/compass>

Viewing Logs

- We can use **docker-compose logs** to view logs from all running containers

```
C:\Users\Toronto\Desktop\tmp\movieList-app>docker-compose up -d
[+] Building 0.0s (0/0)
[+] Running 3/3
  Container movielist-app-db-1      Running
  Container movielist-app-backend-1 Running
  Container movielist-app-frontend-1 Started

C:\Users\Toronto\Desktop\tmp\movieList-app>docker-compose logs
movielist-app-backend-1 | Waiting for MongoDB to start...
movielist-app-frontend-1 |
movielist-app-frontend-1 | > movielist-frontend@0.1.0 start /app
movielist-app-frontend-1 | > react-scripts start
movielist-app-frontend-1 |
movielist-app-frontend-1 | @wds@: Project is running at http://172.18.0.4/
movielist-app-frontend-1 | @wds@: webpack output is served from
movielist-app-frontend-1 | @wds@: Content not from webpack is served from /app/public
movielist-app-frontend-1 | @wds@: 404s will fallback to /
movielist-app-frontend-1 | Starting the development server...
movielist-app-frontend-1 |
movielist-app-frontend-1 | One of your dependencies, babel-preset-react-app, is importing the
movielist-app-frontend-1 | "@babel/plugin-proposal-private-property-in-object" package without
movielist-app-backend-1 | Migrating the database...
movielist-app-backend-1 |
movielist-app-frontend-1 | declaring it in its dependencies. This is currently working because
movielist-app-frontend-1 | "@babel/plugin-proposal-private-property-in-object" is already in your
movielist-app-frontend-1 | node_modules folder for unrelated reasons, but it may break at any time.
movielist-app-frontend-1 |
movielist-app-frontend-1 | babel-preset-react-app is part of the create-react-app project, which
movielist-app-frontend-1 | is not maintained anymore. It is thus unlikely that this bug will
movielist-app-frontend-1 | ever be fixed. Add "@babel/plugin-proposal-private-property-in-object" to
movielist-app-frontend-1 | your devDependencies to work around this error. This will make this message
movielist-app-frontend-1 | go away.
movielist-app-frontend-1 |
movielist-app-frontend-1 | Compiled successfully!
movielist-app-db-1 | 2023-10-29T19:56:34.170+0000 I CONTROL [main] Automatically disabling TLS 1.0, to f
movielist-app-backend-1 | > movielist-backend@1.0.0 db:up /app
movielist-app-backend-1 |
movielist-app-frontend-1 | You can now view movielist-frontend in the browser.
movielist-app-frontend-1 |
movielist-app-frontend-1 | Local:          http://localhost:3000
movielist-app-frontend-1 | On Your Network: http://172.18.0.4:3000
movielist-app-frontend-1 |
movielist-app-frontend-1 | Note that the development build is not optimized.
movielist-app-frontend-1 | To create a production build, use npm run build.
movielist-app-backend-1 | > migrate-mongo up
```

```
C:\Users\Toronto\Desktop\tmp\movieList-app>docker-compose logs --help
```

Usage: docker compose logs [OPTIONS] [SERVICE...]

View output from containers

Options:

<code>--dry-run</code>	Execute command in dry run mode
<code>-f, --follow</code>	Follow log output.
<code>--no-color</code>	Produce monochrome output.
<code>--no-log-prefix</code>	Don't print prefix in logs.
<code>--since string</code>	Show logs since timestamp (e.g. 2013-01-02T13:23:37Z) or relative (e.g. 42m for 42 minutes)
<code>-n, --tail string</code>	Number of lines to show from the end of the logs for each container. (default "all")
<code>-t, --timestamps</code>	Show timestamps.
<code>--until string</code>	Show logs before a timestamp (e.g. 2013-01-02T13:23:37Z) or relative (e.g. 42m for 42 minutes)



How can we publish our application changes using docker-compose?

```
api_1 | [nodemon] starting `node index.js`  
db_1   | 2021-03-17T22:42:11.000+0000 I SHAR  
api_1 | Server started on port 3001...
```