Lab 2 Docker

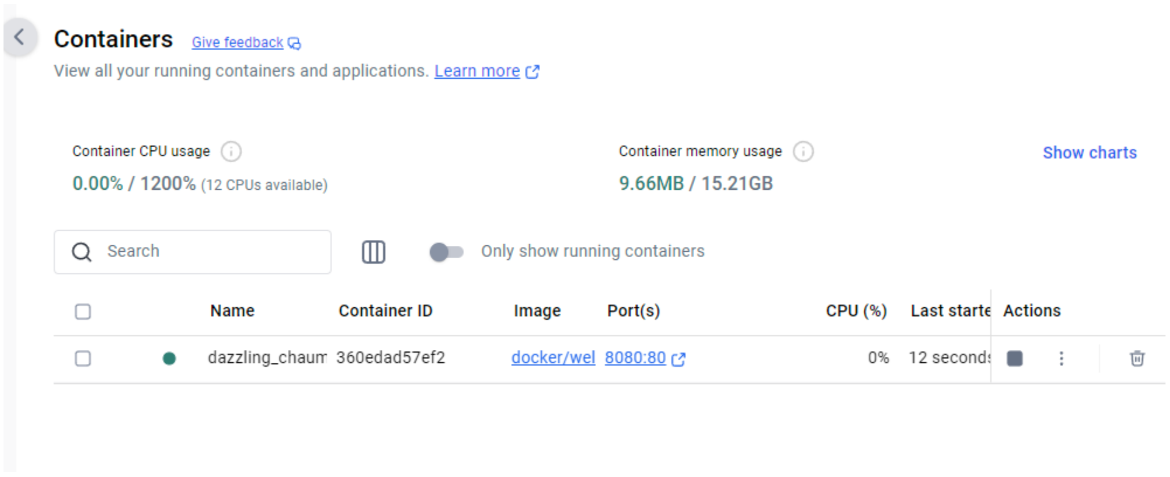
Dylan O'Donnell

05/03/2024

Publishing and exposing ports

I downloaded and installed docker, I then ran this command docker

"run -d -p 8080:80 docker/welcome-to-docker"



Containers Give feedback

View all your running containers and applications. Learn more

Container CPU usage
0.00% / 1200% (12 CPUs available)

Container memory usage
9.66MB / 15.21GB

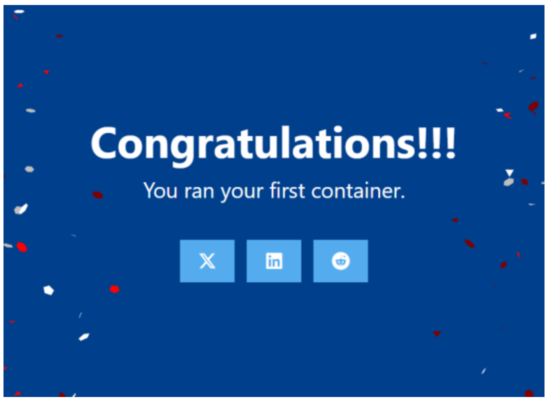
Show charts

Search

Only show running containers

		Name	Container ID	Image	Port(s)	CPU (%)	Last starte	Actions
☐	●	dazzling_chaum	360edad57ef2	docker/wel	8080:80	0%	12 seconds	

On localhost:8080



Congratulations!!!

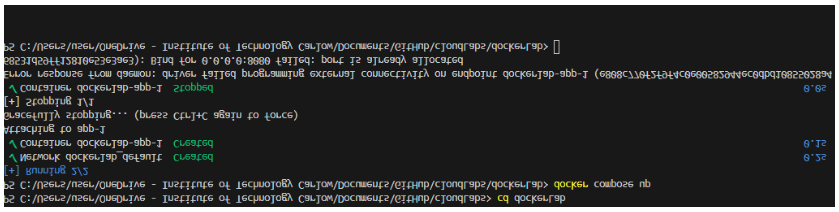
You ran your first container.

Use Docker Compose

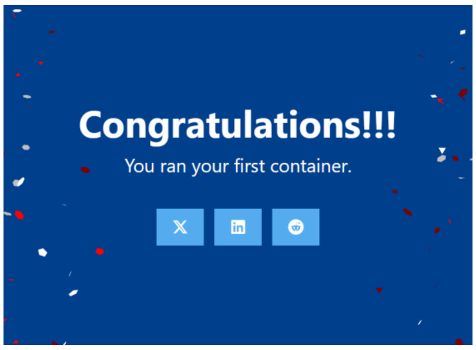
- Create a new directory and inside that directory, create a compose.yaml file with the following contents:

```
services:
 app:
  image: docker/welcome-to-docker
  ports:
   - 8080:80
```

Open your browser to http://localhost:8080.



```
PS C:/Users/user/OneDrive - Institute of Technology Carlow/Documents/GitHub/cloudLabs> cd dockerLab
PS C:/Users/user/OneDrive - Institute of Technology Carlow/Documents/GitHub/cloudLabs/dockerLab> docker compose up
[+] Running 2/2
 ✓ Network dockerlab_default  Created                                                                      0.2s
 ✓ Container dockerlab-app-1  Created                                                                      0.1s
Attaching to app-1
Gracefully stopping... (press Ctrl+C again to force)
[+] Stopping 1/1
 ✓ Container dockerlab-app-1  Stopped                                                                      0.0s
Error response from daemon: driver failed programming external connectivity on endpoint dockerlab-app-1 (6808c770f2f9f4c060082944e0dbd108220284e
e8231d29ff128106e363963): Bind for 0.0.0.0:8080 failed: port is already allocated
PS C:/Users/user/OneDrive - Institute of Technology Carlow/Documents/GitHub/cloudLabs/dockerLab> 
```



This worked the same way.

Overriding Container Defaults

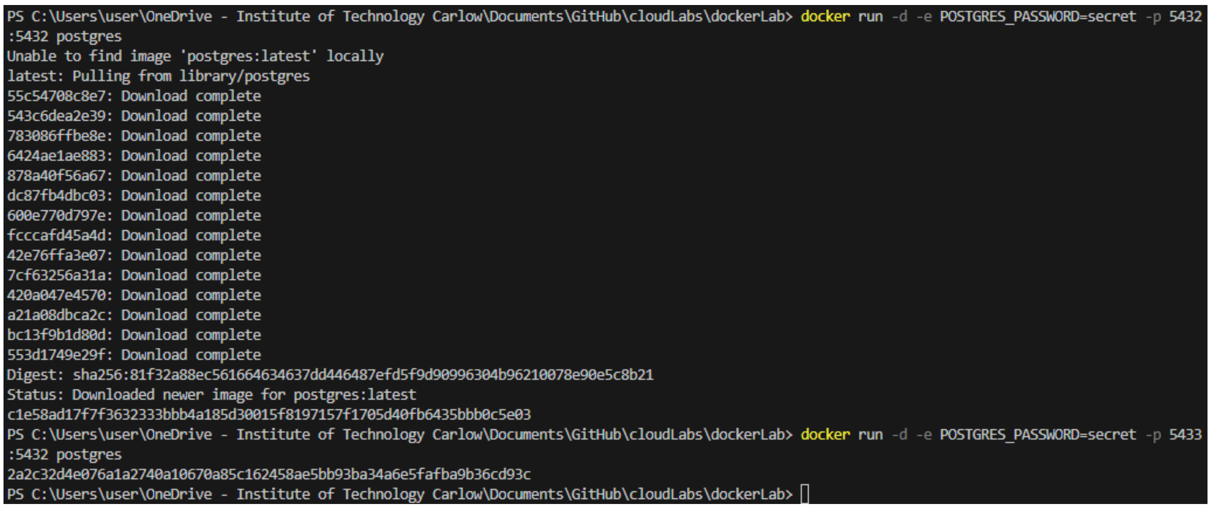
Run multiple instance of the Postgres database

- Start a container using the [Postgres image](https://hub.docker.com/_/postgres) with the following command:

$ docker run -d -e POSTGRES_PASSWORD=secret -p 5432:5432 postgres

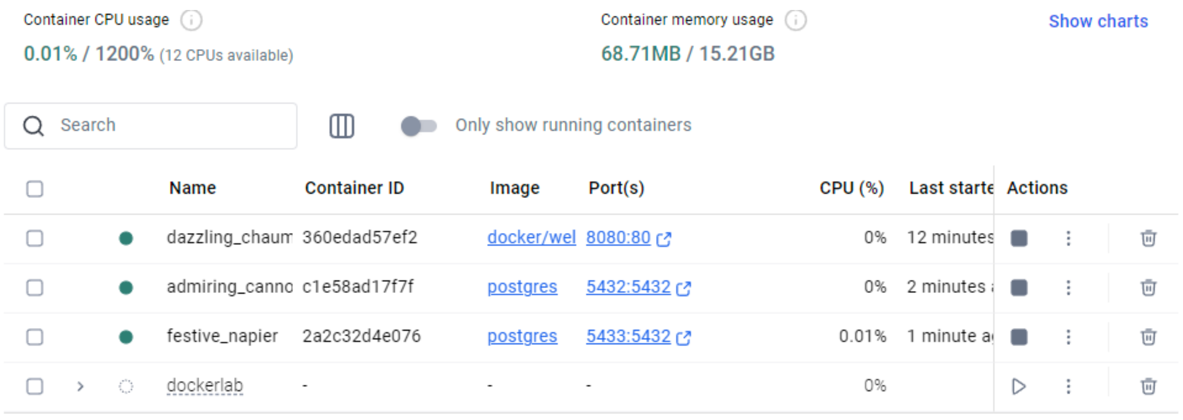
- Start a second Postgres container mapped to a different port.

$ docker run -d -e POSTGRES_PASSWORD=secret -p 5433:5432 postgres



```
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker run -d -e POSTGRES_PASSWORD=secret -p 5432
:5432 postgres
Unable to find image 'postgres:latest' locally
latest: Pulling from library/postgres
55c54708c8e7: Download complete
543c6dea2e39: Download complete
783086ffbe8e: Download complete
6424ae1ae883: Download complete
878a40f56a67: Download complete
dc87fb4dbc03: Download complete
600e770d797e: Download complete
fcccafd45a4d: Download complete
42e76ffa3e07: Download complete
7cf63256a31a: Download complete
420a047e4570: Download complete
a21a08dbca2c: Download complete
bc13f9b1d80d: Download complete
553d1749e29f: Download complete
Digest: sha256:81f32a88ec561664634637dd446487efd5f9d90996304b96210078e90e5c8b21
Status: Downloaded newer image for postgres:latest
c1e58ad17f7f3632333bbb4a185d30015f8197157f1705d40fb6435bbb0c5e03
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker run -d -e POSTGRES_PASSWORD=secret -p 5433
:5432 postgres
2a2c32d4e076a1a2740a10670a85c162458ae5bb93ba34a6e5fafba9b36cd93c
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> 
```

- Verify that both containers are running by going to the **Containers** view in the Docker Desktop Dashboard.


Container CPU usage
0.01% / 1200% (12 CPUs available)
Container memory usage
68.71MB / 15.21GB
Show charts
Search
Only show running containers

	Name	Container ID	Image	Port(s)	CPU (%)	Last starte	Actions
☐	dazzling_chaum	360edad57ef2	docker/wel	8080:80	0%	12 minutes	
☐	admiring_canno	c1e58ad17f7f	postgres	5432:5432	0%	2 minutes	
☐	festive_napier	2a2c32d4e076	postgres	5433:5432	0.01%	1 minute a	
☐	dockerlab	-	-	-	0%		

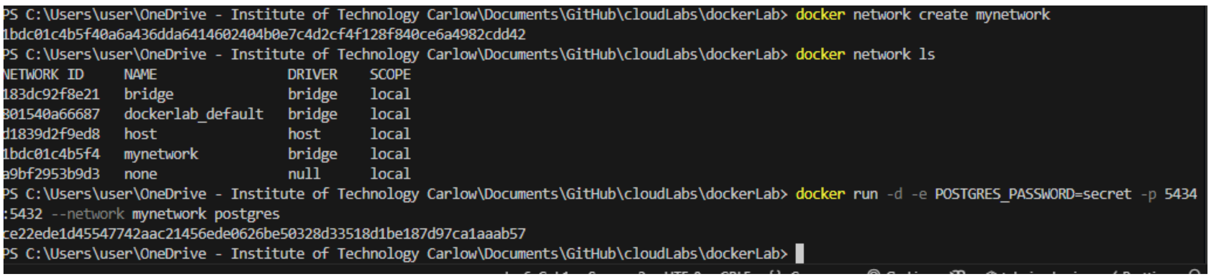
Run Postgres container in a controlled network

1. Create a new custom network by using the following command:
2. **$ docker network create mynetwork**
3. Verify the network by running the following command:
4. **$ docker network ls**

This command lists all networks, including the newly created "mynetwork".

5. Connect Postgres to the custom network by using the following command:

$ docker run -d -e POSTGRES_PASSWORD=secret -p 5434:5432 --network


```
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker network create mynetwork
1bdc01c4b5f40a6a436dda6414602404b0e7c4d2cf4f128f840ce6a4982cdd42
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker network ls
NETWORK ID     NAME                DRIVER    SCOPE
183dc92f8e21   bridge              bridge    local
301540a66687   dockerlab_default   bridge    local
d1839d2f9ed8   host                host      local
1bdc01c4b5f4   mynetwork           bridge    local
a9bf2953b9d3   none                null      local
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker run -d -e POSTGRES_PASSWORD=secret -p 5434
:5432 --network mynetwork postgres
ce22ede1d45547742aac21456ede0626be50328d33518d1be187d97ca1aaab57
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab>
```

Manage the resources

I ran this command

docker run -d -e POSTGRES_PASSWORD=secret --memory="512m" --cpus=".5" postgres

Override the default CMD and ENTRYPOINT in Docker Compose

Create a compose.yml file with the following content:

services:

postgres:

image: postgres

entrypoint: ["docker-entrypoint.sh", "postgres"]

command: ["-h", "localhost", "-p", "5432"]

environment:

POSTGRES_PASSWORD: secret

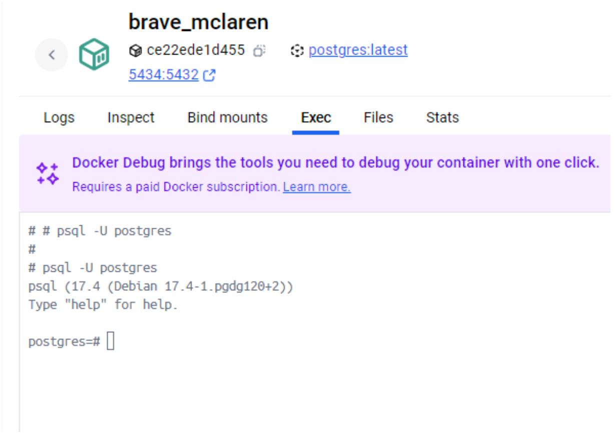
Bring up the service by running the following command:

$ docker compose up -d

This command starts the Postgres service defined in the Docker Compose file.

1. **Verify the authentication with Docker Desktop Dashboard.**

psql -U postgres


brave_mclaren
ce22ede1d455
postgres:latest
5434:5432
Logs
Inspect
Bind mounts
Exec
Files
Stats
Docker Debug brings the tools you need to debug your container with one click.
Requires a paid Docker subscription. Learn more.

```
# # psql -U postgres
#
# psql -U postgres
psql (17.4 (Debian 17.4-1.pgdg120+2))
Type "help" for help.

postgres=# 
```

Persisting container data

Start a container using the Postgres image with the following command:

docker run --name=db -e POSTGRES_PASSWORD=secret -d -v postgres_data:/var/lib/postgresql/data postgres

Connect to the database by using the following command:

docker exec -ti db psql -U postgres

Run this command

CREATE TABLE tasks (

id SERIAL PRIMARY KEY,

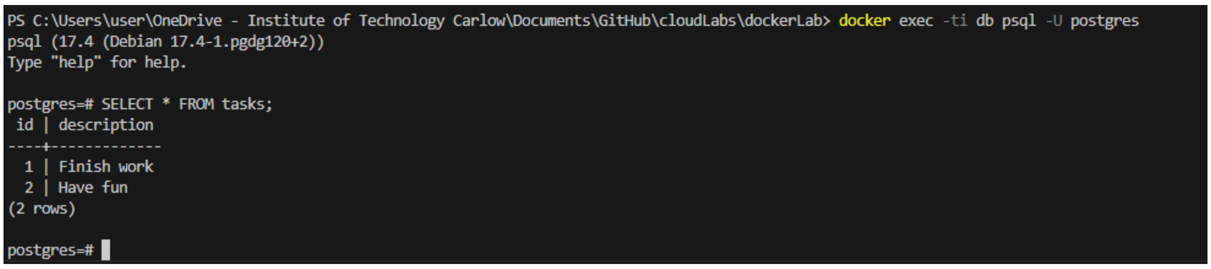
    description VARCHAR(100)

);

INSERT INTO tasks (description) VALUES ('Finish work'), ('Have fun');

Verify the data is in the database

SELECT * FROM tasks;



```
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker exec -ti db psql -U postgres
psql (17.4 (Debian 17.4-1.pgdg120+2))
Type "help" for help.

postgres=# SELECT * FROM tasks;
 id | description
----+-------------
  1 | Finish work
  2 | Have fun
(2 rows)

postgres=# 
```

Exit out of the PostgreSQL shell by running the following command:

\q

$ docker stop db

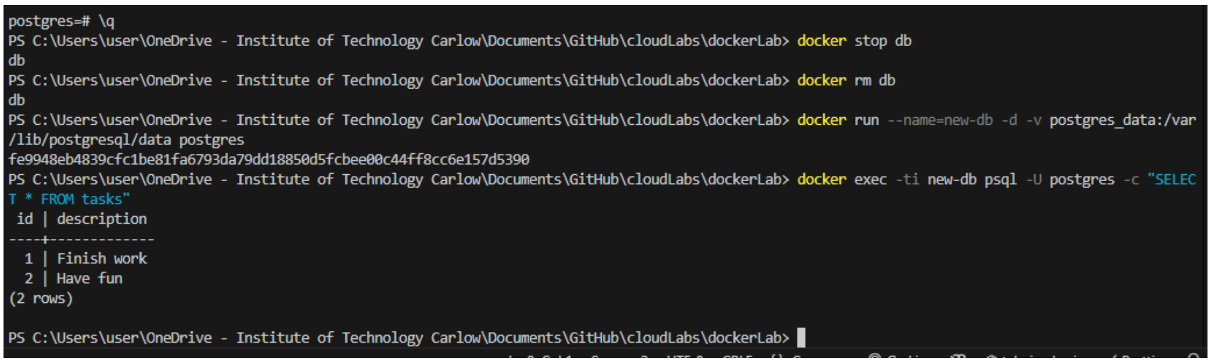
$ docker rm db

Start a new container by running the following command

$ docker run --name=new-db -d -v postgres_data:/var/lib/postgresql/data postgres

Verify the database still has the records by running the following command:

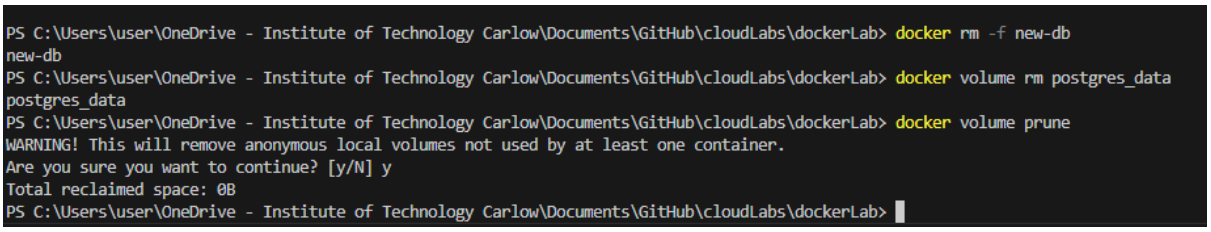
$ docker exec -ti new-db psql -U postgres -c "SELECT * FROM tasks



```
postgres=# \q
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker stop db
db
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker rm db
db
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker run --name=new-db -d -v postgres_data:/var
/lib/postgresql/data postgres
fe9948eb4839cfc1be81fa6793da79dd18850d5fcbee00c44ff8cc6e157d5390
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker exec -ti new-db psql -U postgres -c "SELEC
T * FROM tasks"
 id | description
----+-------------
  1 | Finish work
  2 | Have fun
(2 rows)

PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> 
```

Remove volumes



```
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker rm -f new-db
new-db
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker volume rm postgres_data
postgres_data
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> docker volume prune
WARNING! This will remove anonymous local volumes not used by at least one container.
Are you sure you want to continue? [y/N] y
Total reclaimed space: 0B
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab> 
```

Sharing local files with containers

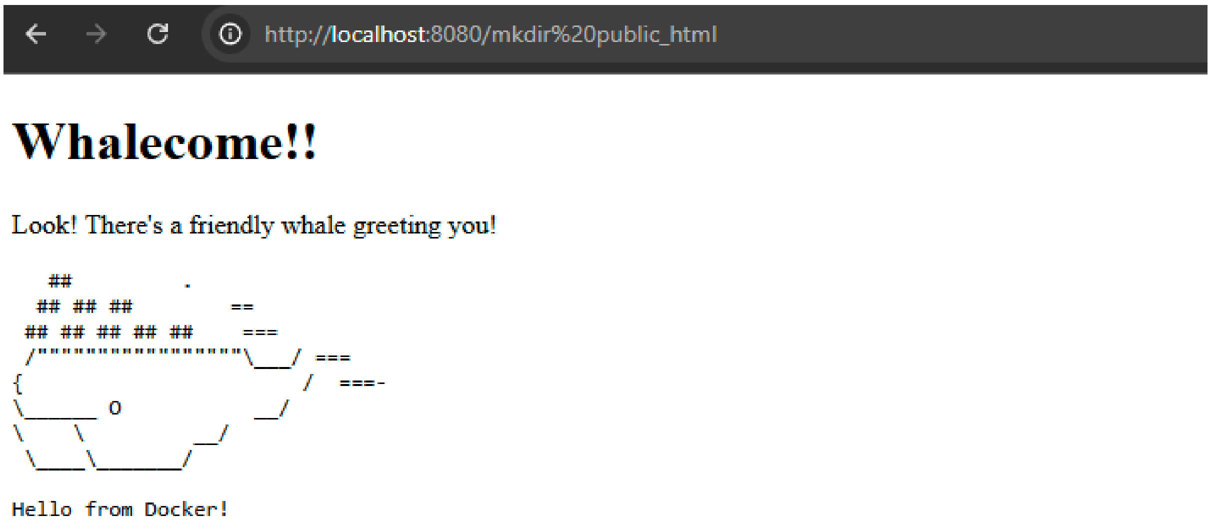
Use a bind mount

Delete the existing container by using the Docker Desktop Dashboard:

Create a new directory called public_html on your host system.

Insert into public_html ascii art of a whale.

Run the container



http://localhost:8080/mkdir%20public_html

Whalecome!!

Look! There's a friendly whale greeting you!

```
   ##         .
  ## ## ##        ==
 ## ## ## ## ##    ===
 /"""""""""""""""""\___/ ===
{                       /  ===-
\______ O           __/
\    \         __/
 \____\_______/
```

```
Hello from Docker!
```

Multi-container applications

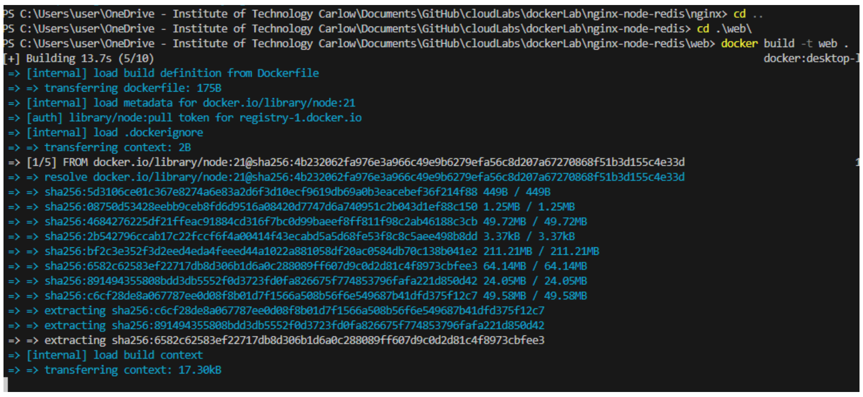
Clone to local

git clone https://github.com/dockersamples/nginx-node-redis

Build the images

1. Navigate into the nginx directory to build the image by running the following command:
2. **$ docker build -t nginx .**
3. Navigate into the web directory and run the following command to build the first web image:

   $ docker build -t web .



```
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab\nginx-node-redis\nginx> cd ..
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab\nginx-node-redis> cd .\web\
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab\nginx-node-redis\web> docker build -t web .
[+] Building 13.7s (5/10)                                                                                                        docker:desktop-l
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 175B
 => [internal] load metadata for docker.io/library/node:21
 => [auth] library/node:pull token for registry-1.docker.io
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [1/5] FROM docker.io/library/node:21@sha256:4b232062fa976e3a966c49e9b6279efa56c8d207a67270868f51b3d155c4e33d
 => => resolve docker.io/library/node:21@sha256:4b232062fa976e3a966c49e9b6279efa56c8d207a67270868f51b3d155c4e33d
 => => sha256:5d3106ce01c367e8274a6e83a2d6f3d10ecf9619db69a0b3eacebef36f214f88 449B / 449B
 => => sha256:08750d53428eebb9ceb8fd6d9516a08420d7747d6a740951c2b043d1ef88c150 1.25MB / 1.25MB
 => => sha256:4684276225df21ffeac91884cd316f7bc0d99baeef8ff811f98c2ab46188c3cb 49.72MB / 49.72MB
 => => sha256:2b542796ccab17c22fccf6f4a00414f43ecabd5a5d68fe53f8c8c5aee498b8dd 3.37kB / 3.37kB
 => => sha256:bf2c3e352f3d2eed4eda4feeed44a1022a881058df20ac0584db70c138b041e2 211.21MB / 211.21MB
 => => sha256:6582c62583ef22717db8d306b1d6a0c288089ff607d9c0d2d81c4f8973cbfee3 64.14MB / 64.14MB
 => => sha256:891494355808bdd3db5552f0d3723fd0fa826675f774853796fafa221d850d42 24.05MB / 24.05MB
 => => sha256:c6cf28de8a067787ee0d08f8b01d7f1566a508b56f6e549687b41dfd375f12c7 49.58MB / 49.58MB
 => => extracting sha256:c6cf28de8a067787ee0d08f8b01d7f1566a508b56f6e549687b41dfd375f12c7
 => => extracting sha256:891494355808bdd3db5552f0d3723fd0fa826675f774853796fafa221d850d42
 => => extracting sha256:6582c62583ef22717db8d306b1d6a0c288089ff607d9c0d2d81c4f8973cbfee3
 => [internal] load build context
 => => transferring context: 17.30kB
```

Run the containers

Create a network for container communication:

docker network create sample-app

Start the Redis container and attach it to the network:

docker run -d --name redis --network sample-app --network-alias redis redis

Start the first web container:

docker run -d --name web1 -h web1 --network sample-app --network-alias web1 web

Start the second web container:

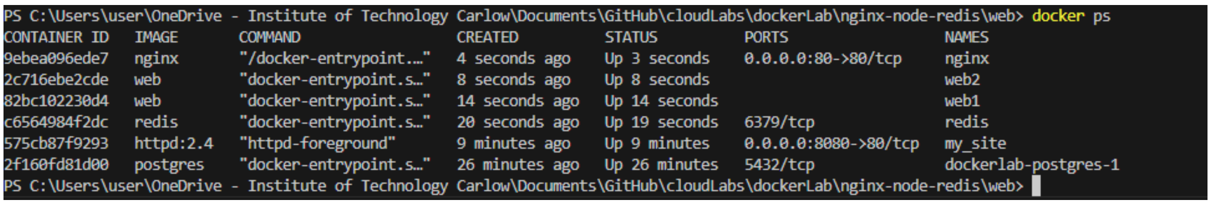
docker run -d --name web2 -h web2 --network sample-app --network-alias web2 web

Start the Nginx container and expose port 80:

docker run -d --name nginx --network sample-app -p 80:80 nginx

Verify all containers are running:

docker ps



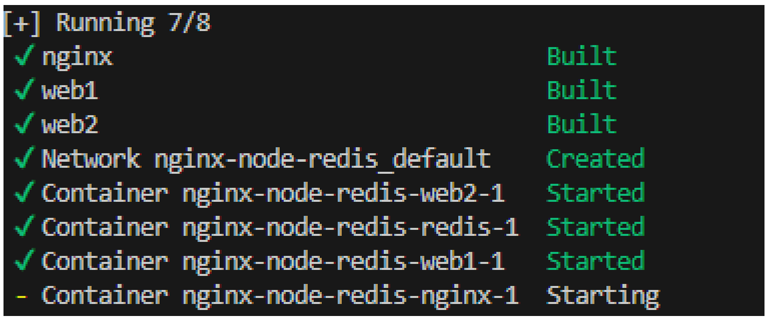
```
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab\nginx-node-redis\web> docker ps
CONTAINER ID   IMAGE       COMMAND                  CREATED          STATUS          PORTS                  NAMES
9ebea096ede7   nginx       "/docker-entrypoint.…"   4 seconds ago    Up 3 seconds    0.0.0.0:80->80/tcp     nginx
2c716ebe2cde   web         "docker-entrypoint.s…"   8 seconds ago    Up 8 seconds                           web2
82bc102230d4   web         "docker-entrypoint.s…"   14 seconds ago   Up 14 seconds                          web1
c6564984f2dc   redis       "docker-entrypoint.s…"   20 seconds ago   Up 19 seconds   6379/tcp               redis
575cb87f9293   httpd:2.4   "httpd-foreground"       9 minutes ago    Up 9 minutes    0.0.0.0:8080->80/tcp   my_site
2f160fd81d00   postgres    "docker-entrypoint.s…"   26 minutes ago   Up 26 minutes   5432/tcp               dockerlab-postgres-1
PS C:\Users\user\OneDrive - Institute of Technology Carlow\Documents\GitHub\cloudLabs\dockerLab\nginx-node-redis\web>
```

Simplify the deployment using Docker Compose

Use the docker compose up command to start the application:

$ docker compose up -d –build

Output:



```
[+] Running 7/8
 ✓ nginx                               Built
 ✓ web1                                Built
 ✓ web2                                Built
 ✓ Network nginx-node-redis_default    Created
 ✓ Container nginx-node-redis-web2-1   Started
 ✓ Container nginx-node-redis-redis-1  Started
 ✓ Container nginx-node-redis-web1-1   Started
 - Container nginx-node-redis-nginx-1  Starting
```