Lab 2 Report

# Publishing and exposing ports

## Publishing Ports

* A port is an endpoint for communication
* Publishing a port is a method of mapping an internal port to an external port

A screenshot of a computer

AI-generated content may be incorrect.

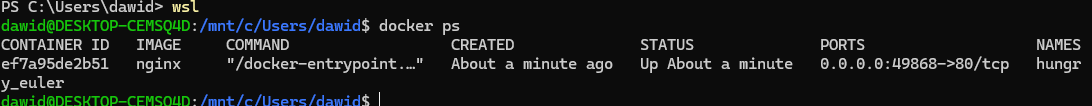
* Above, I publish an 80 nginx port to an 8080 port.

## Publishing to ephemeral ports

* Publishing to ephemeral ports just means you don’t care which external port is chosen during the mapping process

A screenshot of a computer

AI-generated content may be incorrect.



* Above you can see port 80 nginx being assigned to a random external port.

## Publishing all ports

* Container image is a package containing everything needed to run a piece of software.

A screenshot of a computer

AI-generated content may be incorrect.

* Above, I publish all exposed ports to ephemeral ports.

## Try it out

1. Installed docker desktop
2. Started new container called docker/welcome-to-docker

A screen shot of a computer

AI-generated content may be incorrect.

1. Viewed docker dashboard to see if the new container was up.



1. Went to containers website hosted on 8080

A screenshot of a computer

AI-generated content may be incorrect.

## Use Docker Compose

* Docker Compose is a type of tool that you can use to automate certain tasks within docker.
* You can create a compose.yaml files which configures certain values for your containers.

1. Created a compose.yaml file

A screen shot of a computer

AI-generated content may be incorrect.

1. Directed to the directory containing this file
2. Used docker compose up

A computer screen with text

AI-generated content may be incorrect.

1. Opened webpage

A screenshot of a computer

AI-generated content may be incorrect.

# Overriding container defaults

# Persisting container data

## Overriding the network ports

* If you want to run separate database instances, you can use -p option on docker run

## Setting environment variables

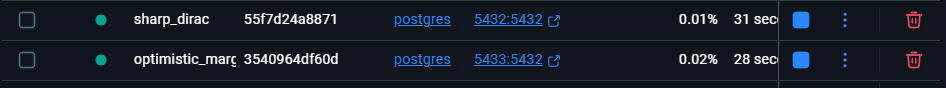
* You can set environment variables using 2 methods
  + docker run -e foo=bar postgres env
  + set env files
    - docker run --env-file .env postgres env

## Restricting the container to consume the resources

* You can control how much memory and CPU a container uses.
* Example:
  + docker run -e POSTGRES\_PASSWORD=secret --memory="512m" --cpus="0.5" postgres
  + POSTGRES\_PASSWORD environmental variables set
  + Memory set to 512m
  + CPU set to 0.5

## Run multiple instances of postgres database

1. Started a container using the PostgreSQL image
2. Started a second PostgreSQL container mapped to a different port
3. Verified both containers are up



Postgres is mapped to 2 different ports on the machine

## Run Postgres container in a controlled network

* By default, containers connect to the bridge network.
* A bridge network is a type of connection in that allows containers on the same host to communicate with each other like above.
* You can create a custom network using –network.

1. Created new custom network
2. Viewed the new network

A black background with white text

AI-generated content may be incorrect.

1. Connected postgres to the new custom network

A screenshot of a computer program

AI-generated content may be incorrect.

Above, you can see what network its connected to.

## Manage the resources

* Essentially the same section as “Restricting the container to consume the resources”.

## Override the default CMD and ENTRYPOINT in Docker Compose

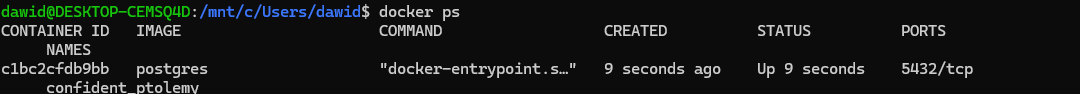
1. Created a compose file
2. Brought up the service
3. Verified the authentication

A screenshot of a computer

AI-generated content may be incorrect.

## Override the default CMD and ENTRYPOINT with docker run

* Here I ran this command:
  + docker run -e POSTGRES\_PASSWORD=secret postgres docker-entrypoint.sh -h localhost -p 5432
  + Below you can see the result of it



* Here's how it breaks down:
  + POSTGRES\_PASSWORD is a set environmental variable
  + docker-entrypoint.sh -h localhost -p 5432 defaults cmd is overwritten
    - Basically, what happens you change the default command this container runs when it starts

# Sharing local files with containers

# Multi-container applications