Week 02 - Docker Worksheet

Louis Botha, louis.botha@tuni.fi

When seeing this emoji 📷, take a screenshot and paste on the worksheet.

Return the worksheet to Moodle as a **PDF**.

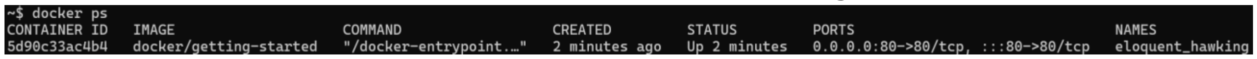
. ‼️ *Do the exercise outside of your git repository directory*

Start the exercise by running the following command:

```
docker run -d -p 80:80 docker/getting-started
```

---

1. Use a docker command to check if the container is running 📷.



```
~$ docker ps
CONTAINER ID   IMAGE                     COMMAND                  CREATED         STATUS         PORTS                               NAMES
5d90c33ac4b4   docker/getting-started    "/docker-entrypoint.…"   2 minutes ago   Up 2 minutes   0.0.0.0:80->80/tcp, :::80->80/tcp   eloquent_hawking
```

---

2. Run the following command:

```
docker run -d -p 80:80 docker/getting-started
```

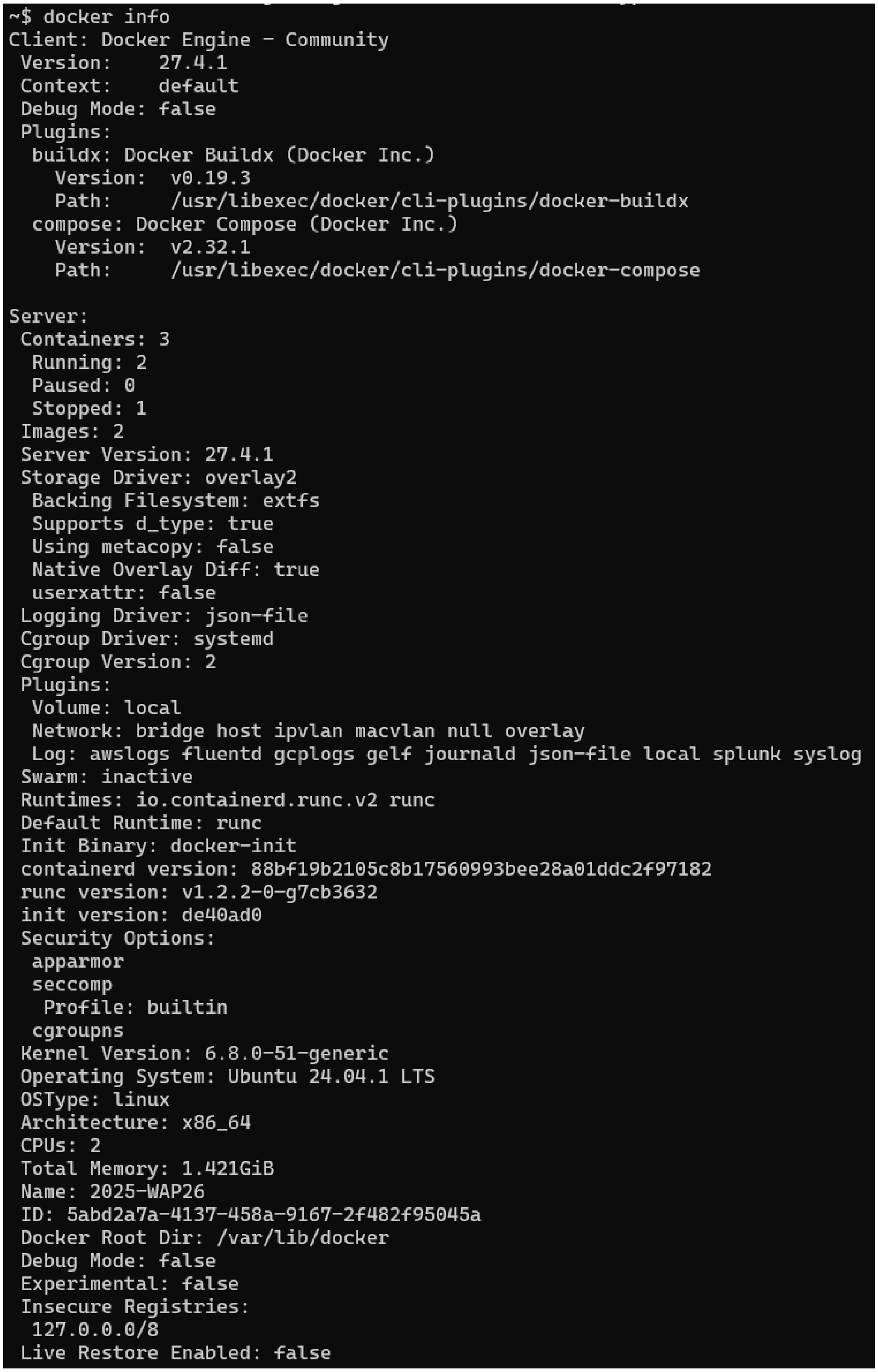
- Did it work? If not why not?
- It did not work because port 80 is already allocated

Run the following command:

```
docker run -d -p 8080:80 docker/getting-started
```

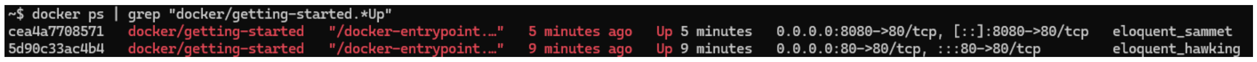
- Did it work? Yes
- What is the differrence? Port 8080 wasn't allocated yet.

Use a docker command to check how many containers are running 📷



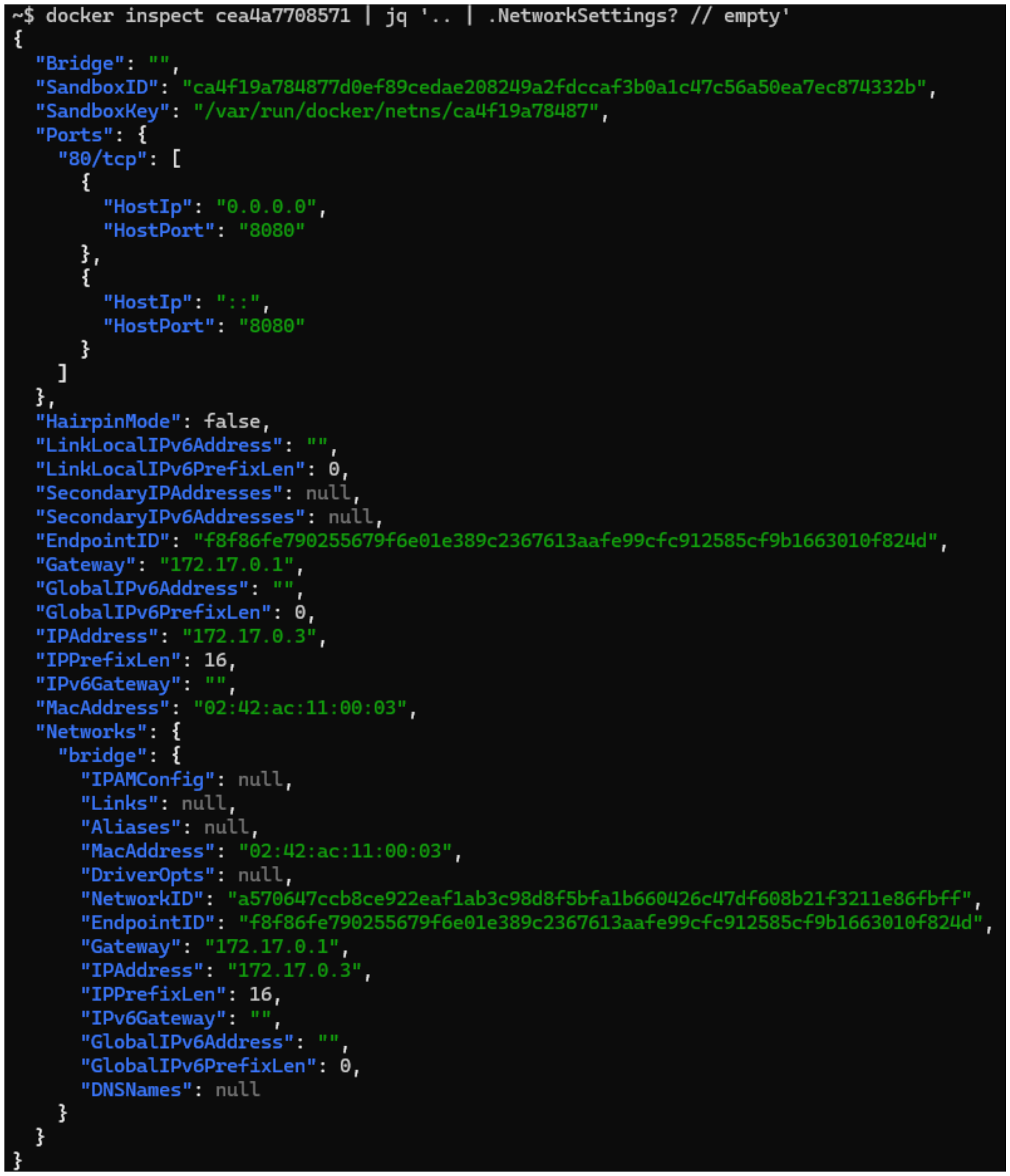
```
~$ docker info
Client: Docker Engine - Community
 Version:    27.4.1
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.19.3
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.32.1
    Path:     /usr/libexec/docker/cli-plugins/docker-compose

Server:
 Containers: 3
  Running: 2
  Paused: 0
  Stopped: 1
 Images: 2
 Server Version: 27.4.1
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Using metacopy: false
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: systemd
 Cgroup Version: 2
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local splunk syslog
 Swarm: inactive
 Runtimes: io.containerd.runc.v2 runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version: 88bf19b2105c8b17560993bee28a01ddc2f97182
 runc version: v1.2.2-0-g7cb3632
 init version: de40ad0
 Security Options:
  apparmor
  seccomp
   Profile: builtin
  cgroupns
 Kernel Version: 6.8.0-51-generic
 Operating System: Ubuntu 24.04.1 LTS
 OSType: linux
 Architecture: x86_64
 CPUs: 2
 Total Memory: 1.421GiB
 Name: 2025-WAP26
 ID: 5abd2a7a-4137-458a-9167-2f482f95045a
 Docker Root Dir: /var/lib/docker
 Debug Mode: false
 Experimental: false
 Insecure Registries:
  127.0.0.0/8
 Live Restore Enabled: false
```

3. User a docker command to check how many docker/getting-started images are on the machine .



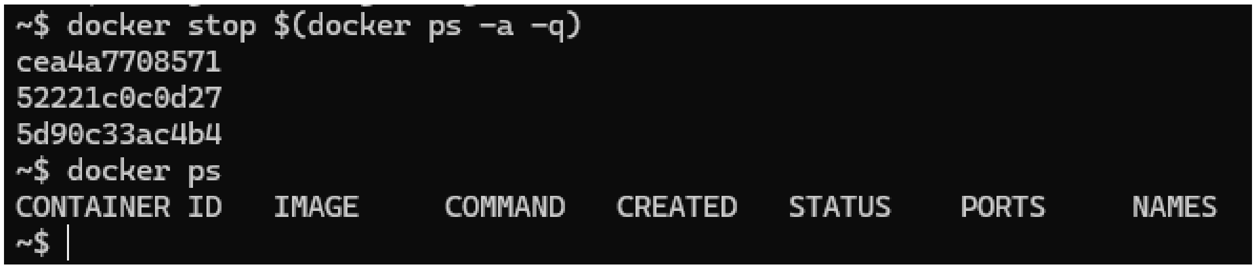
```
~$ docker ps | grep "docker/getting-started.*Up"
cea4a7708571   docker/getting-started   "/docker-entrypoint.…"   5 minutes ago   Up 5 minutes   0.0.0.0:8080->80/tcp, [::]:8080->80/tcp   eloquent_sammet
5d90c33ac4b4   docker/getting-started   "/docker-entrypoint.…"   9 minutes ago   Up 9 minutes   0.0.0.0:80->80/tcp, :::80->80/tcp         eloquent_hawking
```

4. Use the docker inspect <id> and check what is the NetworkSettings/Ports of the container .



```
~$ docker inspect cea4a7708571 | jq '.. | .NetworkSettings? // empty'
{
  "Bridge": "",
  "SandboxID": "ca4f19a784877d0ef89cedae208249a2fdccaf3b0a1c47c56a50ea7ec874332b",
  "SandboxKey": "/var/run/docker/netns/ca4f19a78487",
  "Ports": {
    "80/tcp": [
      {
        "HostIp": "0.0.0.0",
        "HostPort": "8080"
      },
      {
        "HostIp": "::",
        "HostPort": "8080"
      }
    ]
  },
  "HairpinMode": false,
  "LinkLocalIPv6Address": "",
  "LinkLocalIPv6PrefixLen": 0,
  "SecondaryIPAddresses": null,
  "SecondaryIPv6Addresses": null,
  "EndpointID": "f8f86fe790255679f6e01e389c2367613aafe99cfc912585cf9b1663010f824d",
  "Gateway": "172.17.0.1",
  "GlobalIPv6Address": "",
  "GlobalIPv6PrefixLen": 0,
  "IPAddress": "172.17.0.3",
  "IPPrefixLen": 16,
  "IPv6Gateway": "",
  "MacAddress": "02:42:ac:11:00:03",
  "Networks": {
    "bridge": {
      "IPAMConfig": null,
      "Links": null,
      "Aliases": null,
      "MacAddress": "02:42:ac:11:00:03",
      "DriverOpts": null,
      "NetworkID": "a570647ccb8ce922eaf1ab3c98d8f5bfa1b660426c47df608b21f3211e86fbff",
      "EndpointID": "f8f86fe790255679f6e01e389c2367613aafe99cfc912585cf9b1663010f824d",
      "Gateway": "172.17.0.1",
      "IPAddress": "172.17.0.3",
      "IPPrefixLen": 16,
      "IPv6Gateway": "",
      "GlobalIPv6Address": "",
      "GlobalIPv6PrefixLen": 0,
      "DNSNames": null
    }
  }
}
```

5. Use the necessary docker commands to stop all the running containers 📷.



```
~$ docker stop $(docker ps -a -q)
cea4a7708571
52221c0c0d27
5d90c33ac4b4
~$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
~$ |
```

---

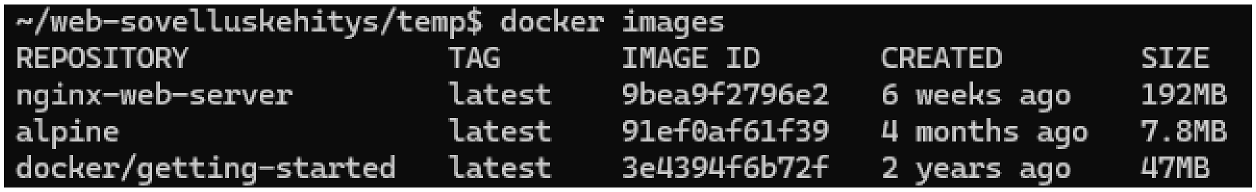
6. Copy the following content into a file named Dockerfile (the file does not have a type).

```
FROM nginx
```

Build the image

```
docker build -t nginx-web-server .
```

Use a docker command to check that the image were build and on the local machine 📷.



```
~/web-sovelluskehitys/temp$ docker images
REPOSITORY               TAG       IMAGE ID       CREATED        SIZE
nginx-web-server         latest    9bea9f2796e2   6 weeks ago    192MB
alpine                   latest    91ef0af61f39   4 months ago   7.8MB
docker/getting-started   latest    3e4394f6b72f   2 years ago    47MB
```

Run the container with docker run nginx-web-server.

- Noticed what is happening? The container is running in the foreground.
- Press CTRL + C to exit the container and stop it.

Run the container with docker run -d nginx-web-server.

- Noticed what happened, what does the -d option do? It runs the container in the background

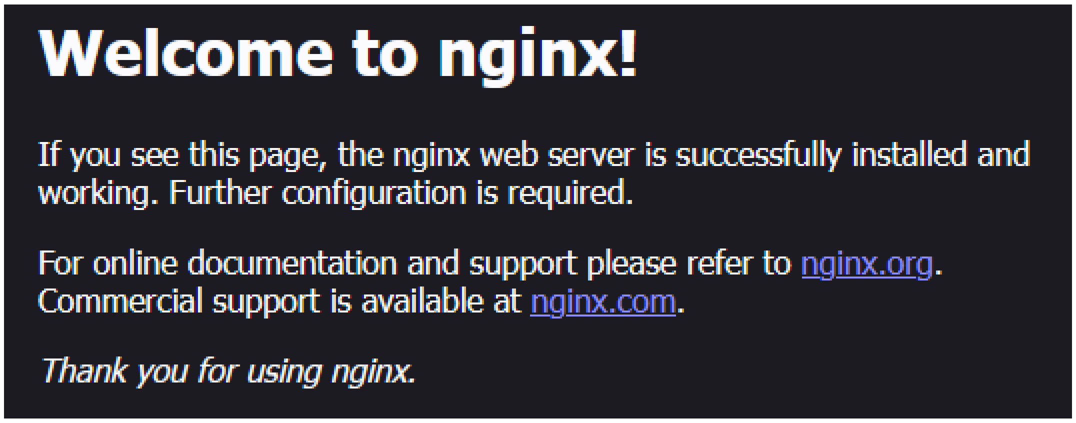
Try to access the container in a web browser by opening localhost.

- Did it work? If not, why not? Did not work and don't exactly know why.
- Stop the running *nginx-web-server* container.

Run the container with docker run -d -p 80:80 nginx-web-server.

- Did it work?
- What is the difference?
- What does the -p 80:80 do?

Try to access the container in a web browser by opening localhost 📷.


Welcome to nginx!

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

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Use a docker command to stop the running containers.

---

7. Create a directory called webserver.

Inside the webserver directory, create a directory called html.
Inside the html directory, create a file called index.html with the following content:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Hello Nginx!</title>
```

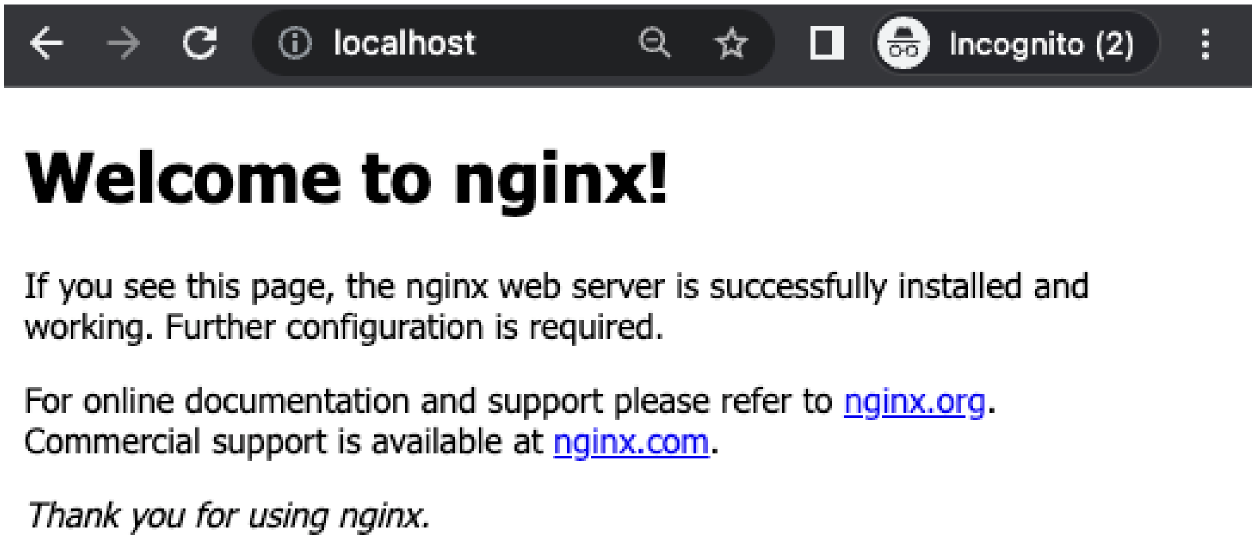
```
</head>
<body>
  <h1>Hello Nginx!</h1>
</body>
</html>
```

The directory structure should look like this:

```
❯ tree
.
└── webserver
    └── html
        └── index.html
```

Run the container with docker run -d -p 80:80 nginx inside the websever directory.

Try to access the container in a web browser by opening localhost. You should see the **nginx** welcome text.



Welcome to nginx!

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

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

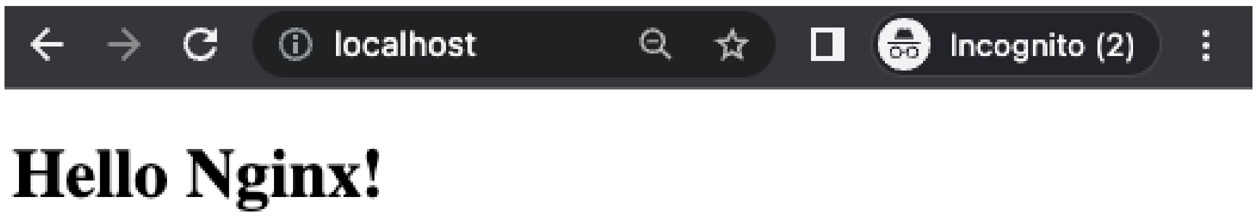
Notice that we don’t see the index.html content we created above.

Stop the running container.

Run the container with:
docker run -d -p 80:80 -v <absolute path>/webserver/html/:/usr/share/nginx/html nginx

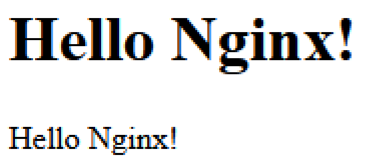
Use pwd to get the absolute path

Try to access the container in a web browser by opening localhost. You should now see the index.html content we created above.



Hello Nginx!

Make a change to the index.html file and check it in the browser 📷.
💡 A hint, start by changing the file and refreshing the page int he browser and see if it works.



Hello Nginx!

Hello Nginx!

Noticed what the -v <source>:<destination> option in the docker command allows us to do?

Use a docker command to stop the running containers.

---

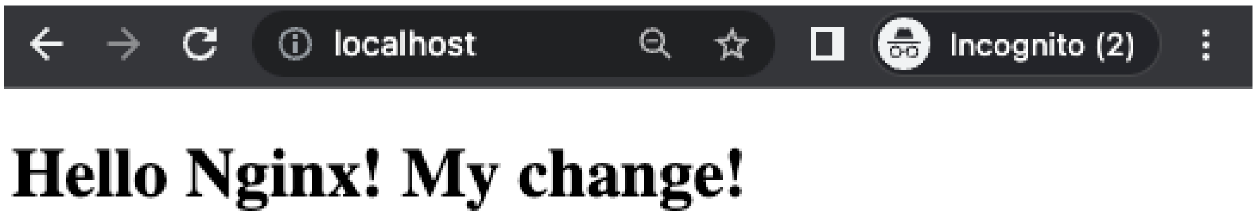
8. Inside the webserver directory create a Dockerfile with this content:

```
FROM nginx
COPY /html /usr/share/nginx/html
```

Run the command docker build -t webserver . to build the container.

Run the command docker run -p 80:80 webserver to run the container.

Try to access the container in a web browser by opening localhost. You should now see the latest index.html content that was created above.



Hello Nginx! My change!

Make some changes to the index.html file and refresh the web browser.
Notice any changes? If not, why not? Because the image was build on the index.html file before it was changed.

Run the command docker build -t webserver . to build the container.

Run the command docker run -p 80:80 webserver to run the container.

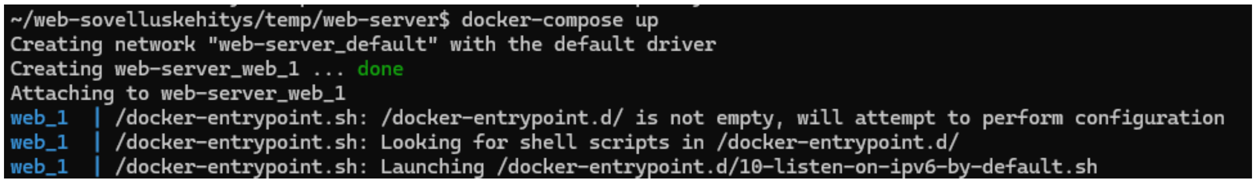
Try to access the container in a web browser by opening localhost. You should now see changes made to the index.html content above.

Use a docker command to stop the running containers.

9. Copy the following content into a file named docker-compose.yml.

```
services:
  web:
    image: nginx
    ports:
    - "8080:80"
```

Use a docker compose command to run the services on the local machine 📷



```
~/web-sovelluskehitys/temp/web-server$ docker-compose up
Creating network "web-server_default" with the default driver
Creating web-server_web_1 ... done
Attaching to web-server_web_1
web_1  | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
web_1  | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
web_1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
```

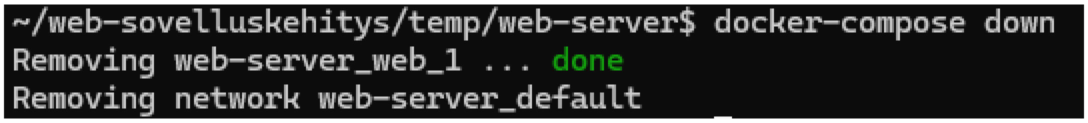
Try to access the container in a web browser by opening localhost

- Did it work? If not, why not? It worked when opened localhost:8080

Try to access the container in a web browser by opening localhost:8080

- Did it work?
- What is the difference?
- What does the line - "8080:80" do?

10. Use a docker compose command to stop the services on the local machine 📷


```
~/web-sovelluskehitys/temp/web-server$ docker-compose down
Removing web-server_web_1 ... done
Removing network web-server_default
```

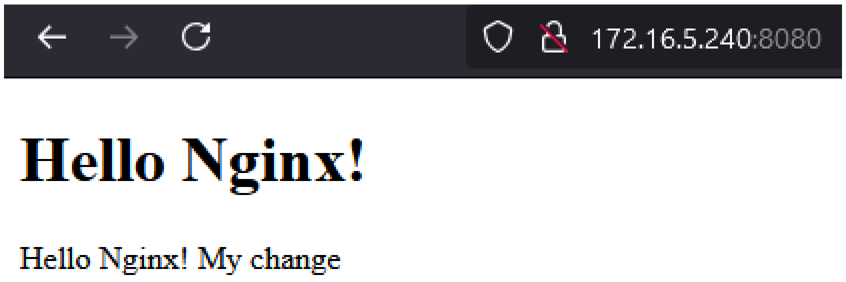
---

11. Change the following content in the docker-compose.yml file.

```
services:
  web:
    image: nginx
    ports:
    - "8080:80"
    volumes:
    - ./html:/usr/share/nginx/html
```

Use a docker compose command to run the services on the local machine.

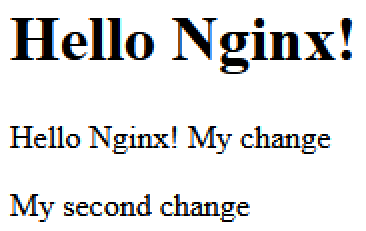
Try to access the container in a web browser by opening localhost:8080.
You should now see the latest index.html content that was created above 📷.


172.16.5.240:8080

Hello Nginx!

Hello Nginx! My change

Make a change to the index.html file and check it in the browser 📷.
💡 A hint, start by changing the file and refreshing the page int he browser and see if it works.



Hello Nginx!

Hello Nginx! My change

My second change

Use a docker compose command to stop the services on the local machine.