Assignment: Containerization with Docker Report

## Personal

* Name: Zackaria Osman
* ID: 000885686
* Email:
  + Primary: [cobalt.zr86@gmail.com](mailto:cobalt.zr86@gmail.com)
  + Secondary: [zackaria.osman@edu.sait.ca](mailto:zackaria.osman@edu.sait.ca)

## Links

GitHub repository: <https://github.com/ZackariaOsman/docker-challenge-template>

## Basic introduction of Docker

### Why is it important for me, in the context of software development?

* + Consistency
    - Ensures applications behave consistently across different environments.
  + Isolation
    - Encapsulates applications and dependencies, enhancing security and reliability.
  + Efficiency
    - Lightweight containers enable faster deployment and scaling.
  + Portability
    - Runs on any system supporting Docker, simplifying deployment and collaboration.
  + Version Control
    - Images are versioned and shareable, ensuring reproducibility and consistency.

# Challenge 1 - Simple static page server

### Docker Commands Used

\*\*\* ’.’ = current working directory \*\*\*

* docker build
  + Builds an image from a Dockerfile in the current directory.
* docker images
  + Lists all available Docker images.
* docker build -t <repository\_name>
  + Builds an image and tags it with a repository name.
* docker run -d -p <host\_port>:<container\_port> <image\_id>
  + Runs a container in detached mode, mapping ports between host and container.
* docker ps
  + Lists running containers.
* docker stop <container\_id> or docker stop <container\_name>:
  + Stops a running container.
* docker ps -a
  + Lists all containers (both running and stopped).
* docker rm <container\_id> or docker rm <container\_name>
  + Removes a container.
* docker rmi <image\_id> or docker rmi <image\_name>
  + Removes an image.

## Steps To Make It Work

**Installation**

### Install Docker

* + Downloaded the Docker Engine from the Docker website followed the instructions for the installation process.

**Configuration**

### Creating a DockerFile:

# To use the Nginx image from Docker Hub

FROM nginx:alpine

# Copy  contents of  public folder to Nginx html directory

COPY public/ /usr/share/nginx/html/

# Expose port 80 but it is not mandatory

#EXPOSE 80

# Start Nginx when container starts

#Also not mandatory

#CMD ["nginx", "-g", "daemon off;"]

#For Nginx in debug mode

#CMD ["nginx-debug", "-g", "daemon off;"]

### Building an Image:

* + Use docker build . to build an image from a Dockerfile located in the current directory (.).
  + Optionally, you can tag the image with a repository name using
    - docker build -t <repository\_name> ..

### Listing Images

* + After building, use docker images to list all locally available Docker images.

### Creation of Files

#### Creating and Running Containers:

* + Create a container from an image with docker run:
    - Use docker run -d -p <host\_port>:<container\_port> <image\_id> to run a container in detached mode (-d) and map ports (-p). Replace <host\_port> and <container\_port> with appropriate values.
      * Example: docker run -d -p 80:80 8b0a2c43a149 runs a container from image 8b0a2c43a149, mapping host port 80 to container port 80.

#### Managing Containers:

* + View running containers with
    - docker ps.
  + Stop a container using
    - docker stop <container\_id> or docker stop <container\_name>.
  + List all containers (including stopped ones) with
    - docker ps -a.
  + Remove a container with
    - docker rm <container\_id> or docker rm <container\_name>.

#### Managing Images:

* + Remove an image using
    - docker rmi <image\_id> or docker rmi <image\_name>.

### Screenshots

Png1:

A screen shot of a computer screen

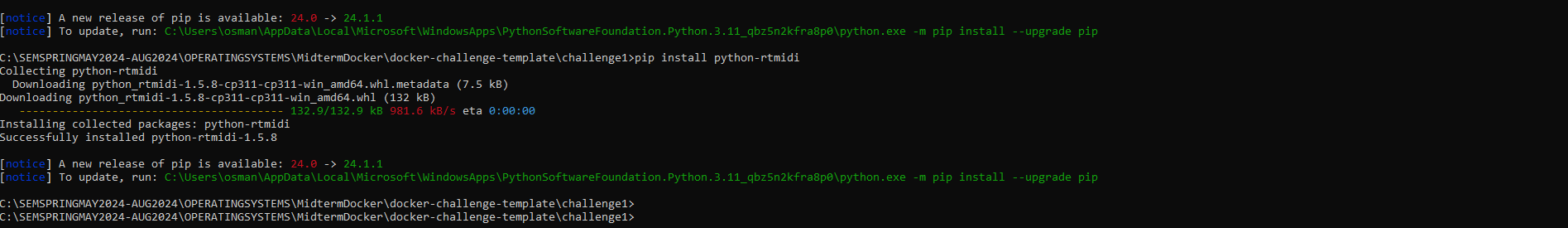
Description automatically generated

Png2:

A screen shot of a computer program

Description automatically generated

Png3:



Result:



## Lessons learned

No, nothing went wrong, but this is what I learned:

* Docker Installation and Set up
* Creating Dockerfiles
* Building Images
* Managing Containers and Images

## References

* YouTube:
  + <https://www.youtube.com/watch?v=SnSH8Ht3MIc&list=LL&index=7&t=929s>
* LinkedIn
  + <https://www.linkedin.com/pulse/why-every-software-engineer-should-learn-docker-s-gouse-basha-2amxe/>

# Challenge 2 - NodeJS application

## Docker Commands Used

\*\*\* ’.’ = current working directory \*\*\*

#to build the image

#docker build .

#to list the images

#docker images

#to add a tag to the image

#docker-compose build

#to create a container

#docker-compose up

#See the running containers

#docker-compose ps

## Steps To Make It Work

**Configuration**

### Creating a DockerFile:

# To use the Nginx image from Docker Hub

FROM nginx:alpine

# Use the Node.js image from Docker Hub

FROM node:14.17.0-alpine3.13

WORKDIR /app

# Copy application dependency manifests to the container image.

COPY package\*.json ./

# Install application dependencies.

RUN npm install

# Bundle app source

COPY . .

# Expose the port the app runs on

EXPOSE 3000

# Start the Node.js app

CMD ["npm", "start"]

### Creating a Docker-Compose:

version: '3'

services:

  web:

    build:

      context: .

      dockerfile: Dockerfile

    ports:

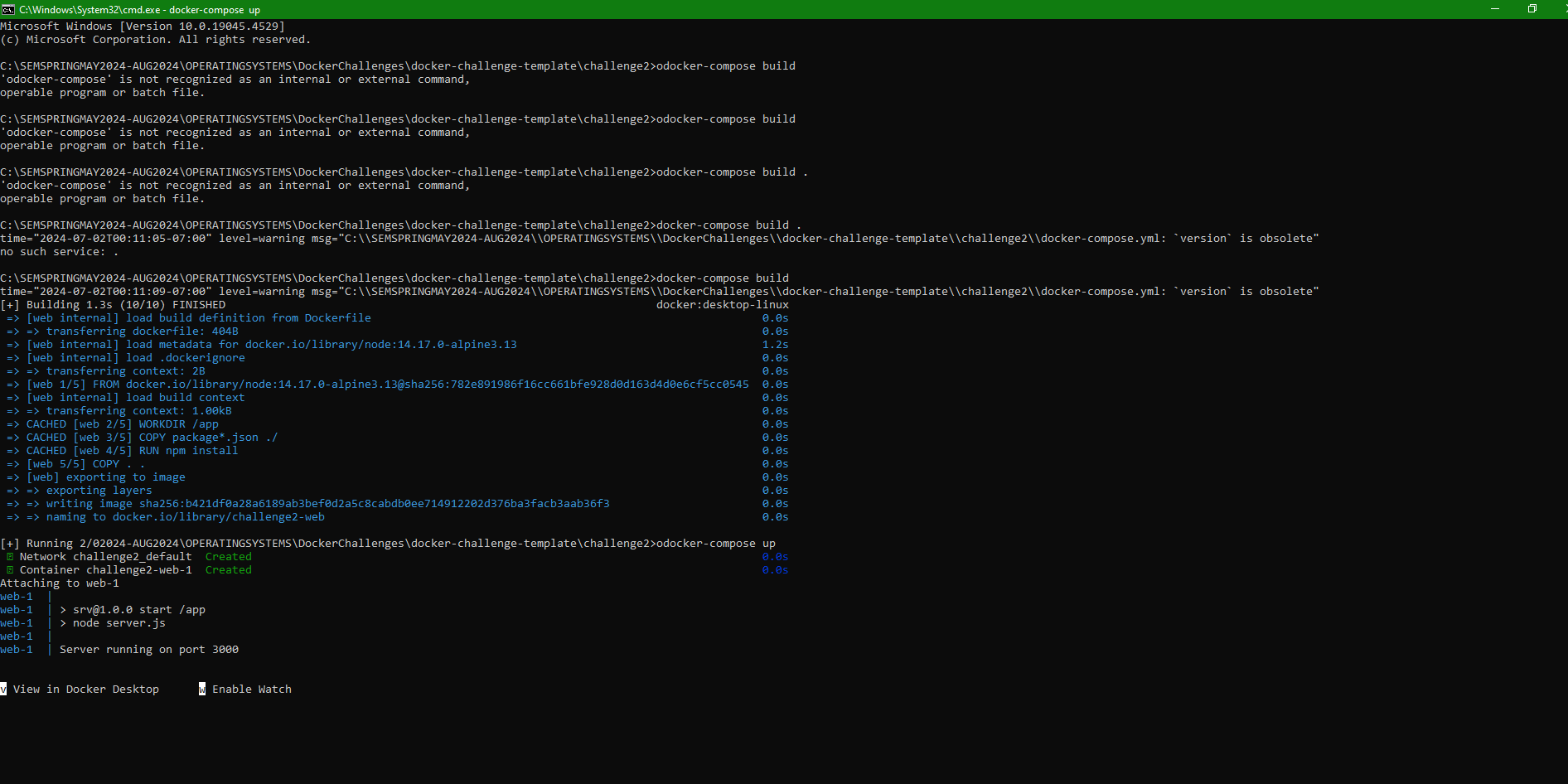
      - "8080:3000"

### Steps to Build and Run

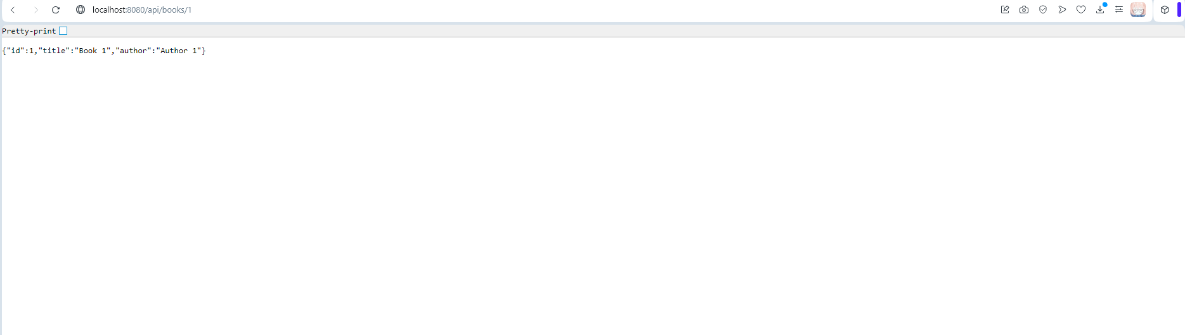
1. **Navigate to the challenge2 Directory and open CMD**
2. **Build Docker Images**:
   * docker-compose build
3. **Run Docker Containers**
   * docker-compose up
4. **Check Running Containers**:
   * docker-compose ps
5. **Open your web browser and navigate to**:
   * http://localhost:8080/api/books to get a JSON message with all books.
   * http://localhost:8080/api/books/1 to get a JSON message with the book of ID 1.

### Screenshots

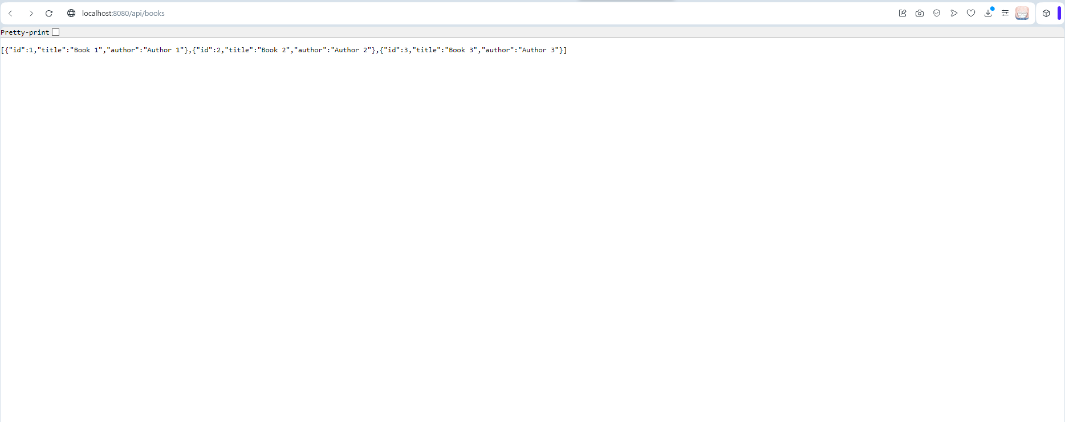
Png1:



Result1:



Result2:



## Lessons learned

Again No, nothing went wrong, but this is what I learned:

* Docker Compose
* YouTube:
  + <https://www.youtube.com/watch?v=0B2raYYH2fE>
  + <https://www.youtube.com/watch?v=hXhI2ZLDgQM>