Operating Systems
Midterm Project

CPSY-300-C

Docker
Challenges 1 & 2

Name:

Rommel Hipos

Date Submitted:

June 29, 2024

Docker Introduction

Docker is an open-source platform that allows you to automate the deployment, scaling, and management of applications using containerization. Containers are lightweight, portable units that encapsulate an application and all its dependencies, ensuring that the application runs consistently across different environments [1].

Importance of Docker in Software Development

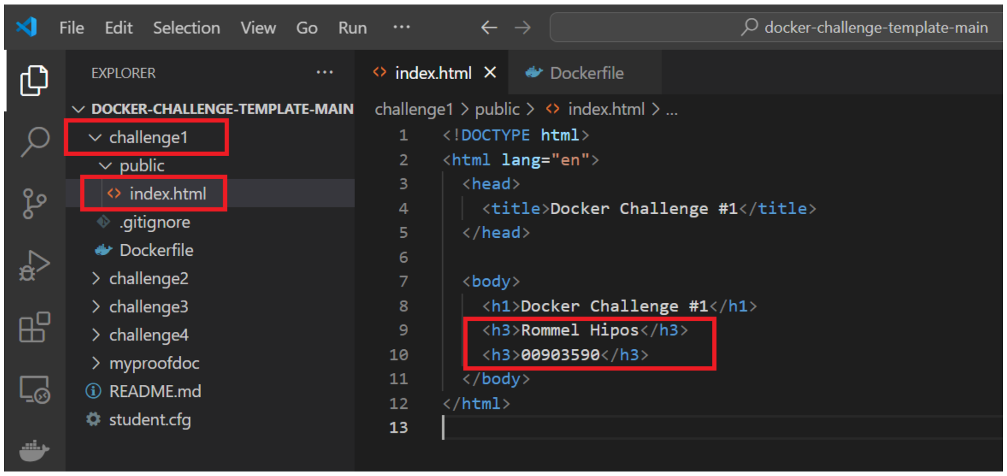
Using Docker Desktop facilitates teamwork by enabling one-click sharing of work via Git or Docker Hub. Additionally, it features an intuitive user interface (UI) for numerous common tasks, such as initiating, pausing, and ending containers. Through our Docker Community Slack channel, users of Docker may also communicate, learn from, and work together with one another. By leveraging Docker's powerful containerization technology, developers can streamline their workflows, enhance collaboration, and ensure that applications run reliably in any environment.

Additionally, Docker users can learn, connect, and collaborate with each other via our Docker Community Slack channel. We can chat with Docker community leaders, Docker Captains, and your fellow local developers in the channel. [2].

Challenge 1- Simple static page server

Steps

- Use the folder challenge1.
- Add a "public" folder with some assets.
- Add a file with the name index.html. It should contain your **name** and **SAIT ID** in the contents.

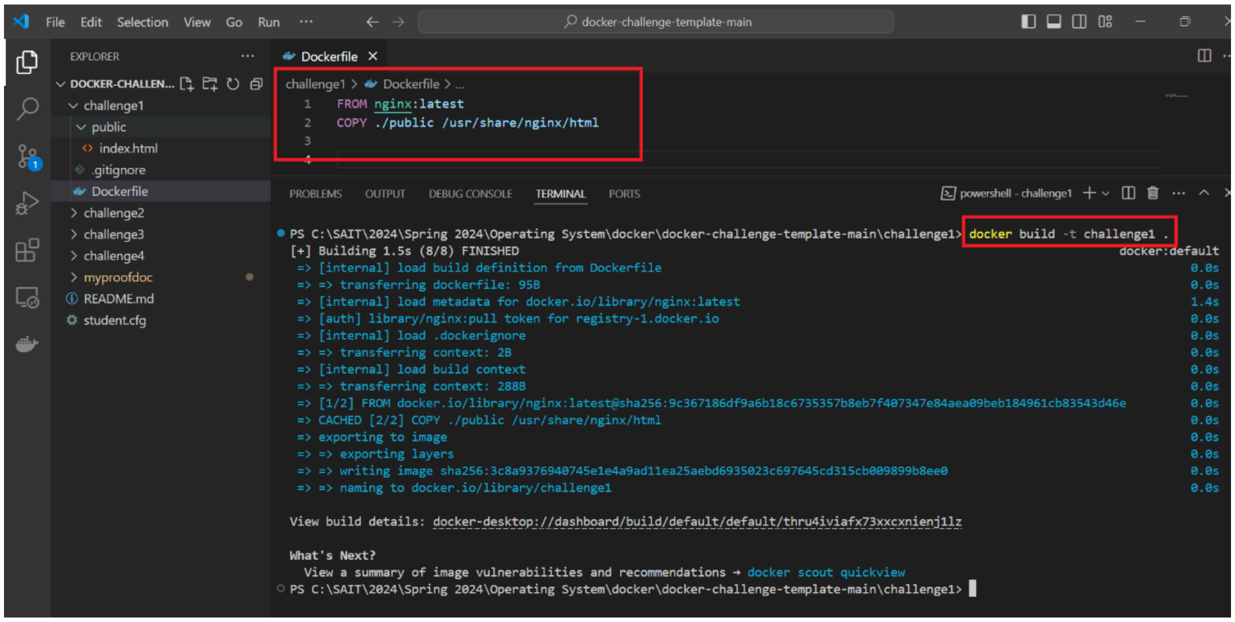


```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Docker Challenge #1</title>
  </head>

  <body>
    <h1>Docker Challenge #1</h1>
    <h3>Rommel Hipos</h3>
    <h3>00903590</h3>
  </body>
</html>
```

Lessons Learned: *The purpose of these steps is to set up a basic Docker image that includes a simple web application structure. This structure includes static assets in a public folder and a minimal index.html file that can be served by a web server running inside the Docker container.*

- Create a Dockerfile to use NGinx to serve pages existent in the public folder.
- Create the Docker image.
- Execute docker with the right parameters.

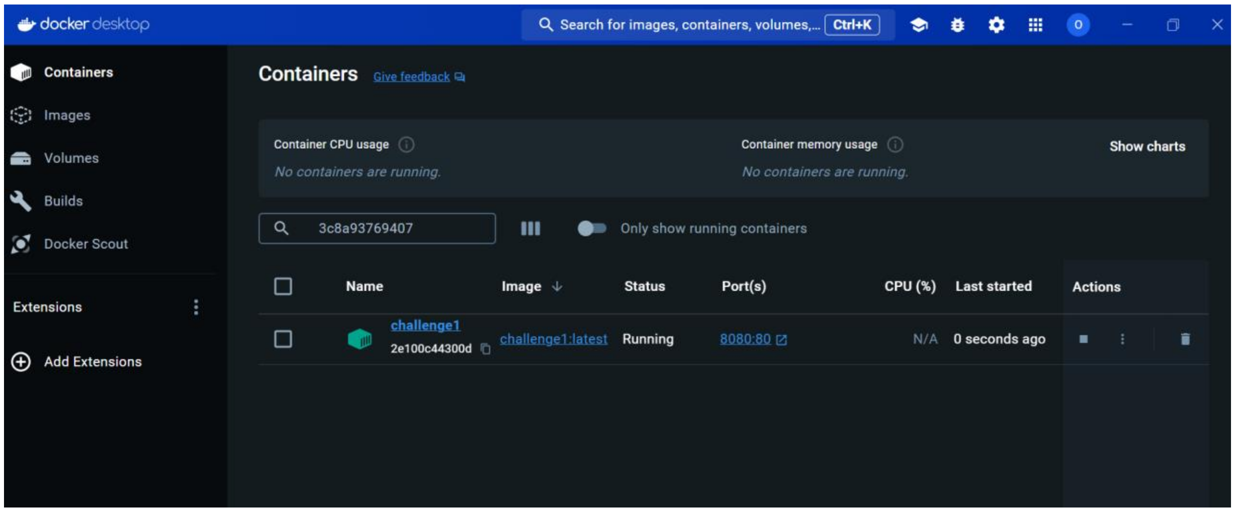
  - *Command to build the image*: ***docker build -t <name of the image> .***



```
FROM nginx:latest
COPY ./public /usr/share/nginx/html
```

```
PS C:\SAIT\2024\Spring 2024\Operating System\docker\docker-challenge-template-main\challenge1> docker build -t challenge1 .
[+] Building 1.5s (8/8) FINISHED                                                                         docker:default
 => [internal] load build definition from Dockerfile                                                              0.0s
 => => transferring dockerfile: 95B                                                                               0.0s
 => [internal] load metadata for docker.io/library/nginx:latest                                                   1.4s
 => [auth] library/nginx:pull token for registry-1.docker.io                                                      0.0s
 => [internal] load .dockerignore                                                                                 0.0s
 => => transferring context: 2B                                                                                   0.0s
 => [internal] load build context                                                                                 0.0s
 => => transferring context: 288B                                                                                 0.0s
 => [1/2] FROM docker.io/library/nginx:latest@sha256:9c367186df9a6b18c6735357b8eb7f407347e84aea09beb184961cb83543d46e  0.0s
 => CACHED [2/2] COPY ./public /usr/share/nginx/html                                                              0.0s
 => exporting to image                                                                                            0.0s
 => => exporting layers                                                                                           0.0s
 => => writing image sha256:3c8a9376940745e1e4a9ad11ea25aebd6935023c697645cd315cb009899b8ee0                      0.0s
 => => naming to docker.io/library/challenge1                                                                     0.0s

View build details: docker-desktop://dashboard/build/default/default/thru4iviafx73xxcxnienj1lz

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview
PS C:\SAIT\2024\Spring 2024\Operating System\docker\docker-challenge-template-main\challenge1>
```

  - *Once the build is successfully executed, open the Docker desktop to view the image created.*



Containers

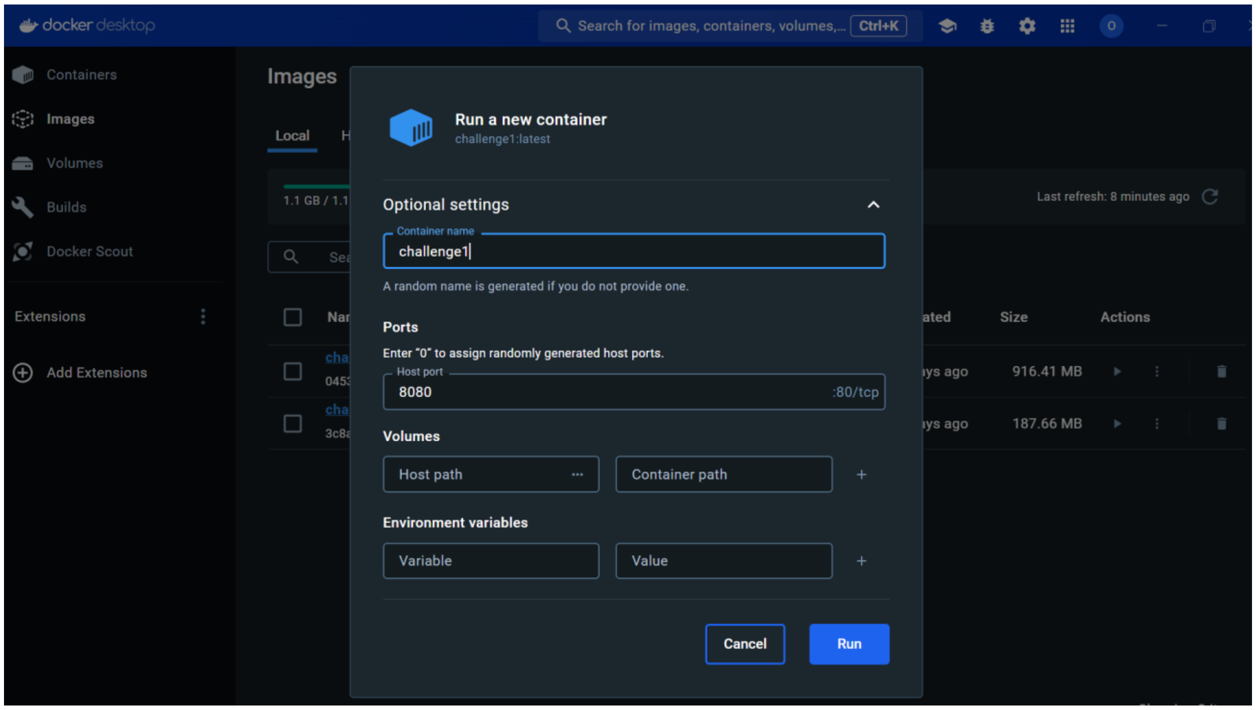
Container CPU usage: No containers are running.

Container memory usage: No containers are running.

Search: 3c8a93769407 | Only show running containers

	Name	Image	Status	Port(s)	CPU (%)	Last started	Actions
☐	challenge1 2e100c44300d	challenge1:latest	Running	8080:80	N/A	0 seconds ago	

- Run the image to generate the container and set the port to 8080


Run a new container
challenge1:latest

Optional settings

Container name: challenge1

A random name is generated if you do not provide one.

Ports

Enter "0" to assign randomly generated host ports.

Host port: 8080 :80/tcp

Volumes

Host path | Container path

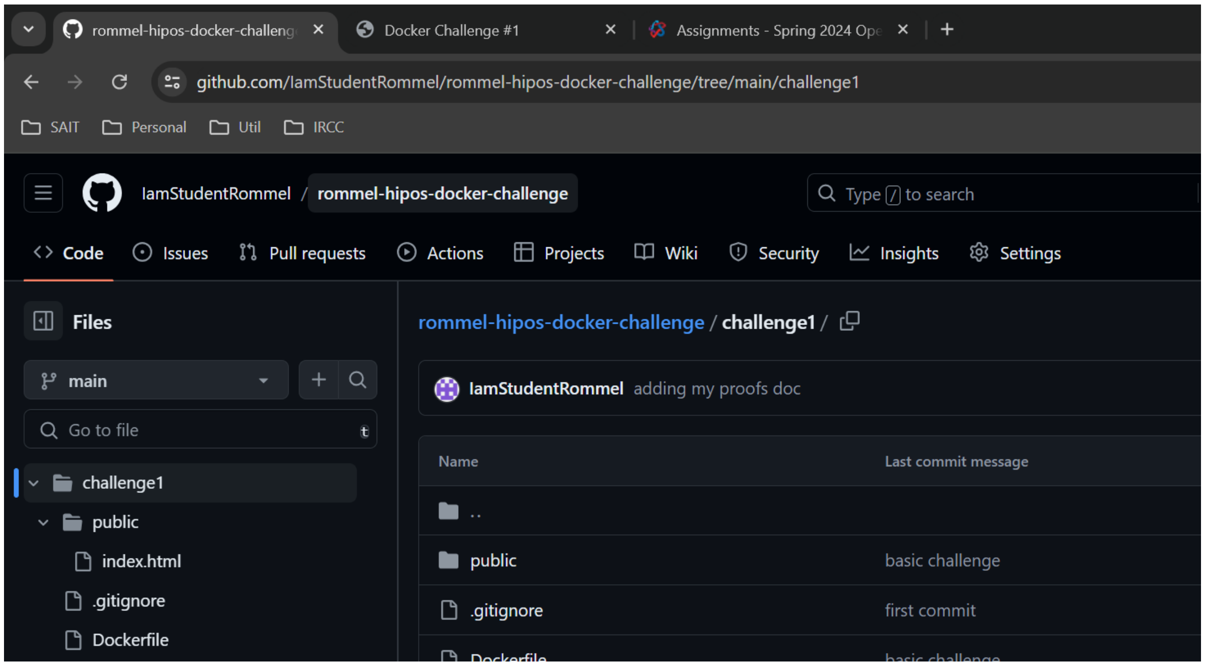
Environment variables

Variable | Value

- Now it's ready to run inside the container.

Lessons Learned*: With the help of these methods, I can quickly create a Docker image that encapsulates the whole web server setup into a portable and managed containerized environment, using NGinx to serve static web content from a public folder.*

- Commit the Dockerfile and public folder and push it to the remote repository.


github.com/IamStudentRommel/rommel-hipos-docker-challenge/tree/main/challenge1

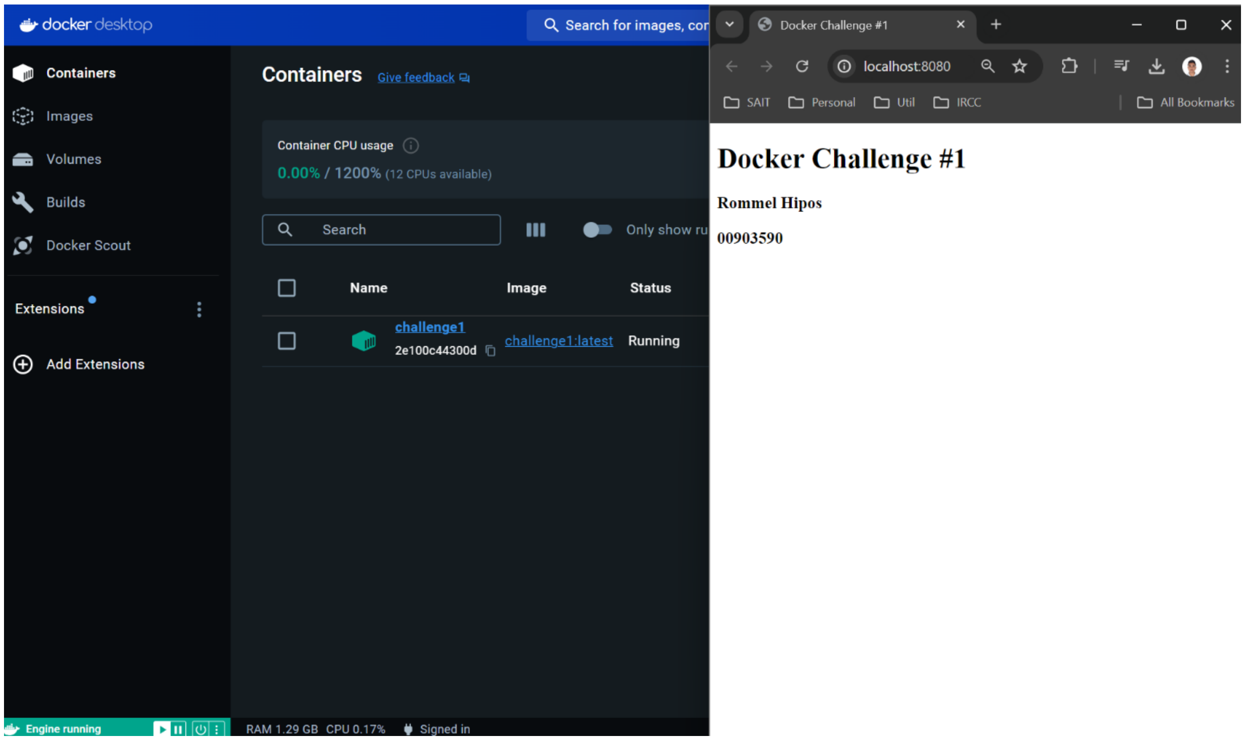
rommel-hipos-docker-challenge / challenge1 /

IamStudentRommel adding my proofs doc

Name	Last commit message
..	
public	basic challenge
.gitignore	first commit
Dockerfile	basic challenge

Expected outcomes

- When you request the URL “http://localhost:8080/” you will get a home page with your name and code.



Docker Challenge #1

Rommel Hipos

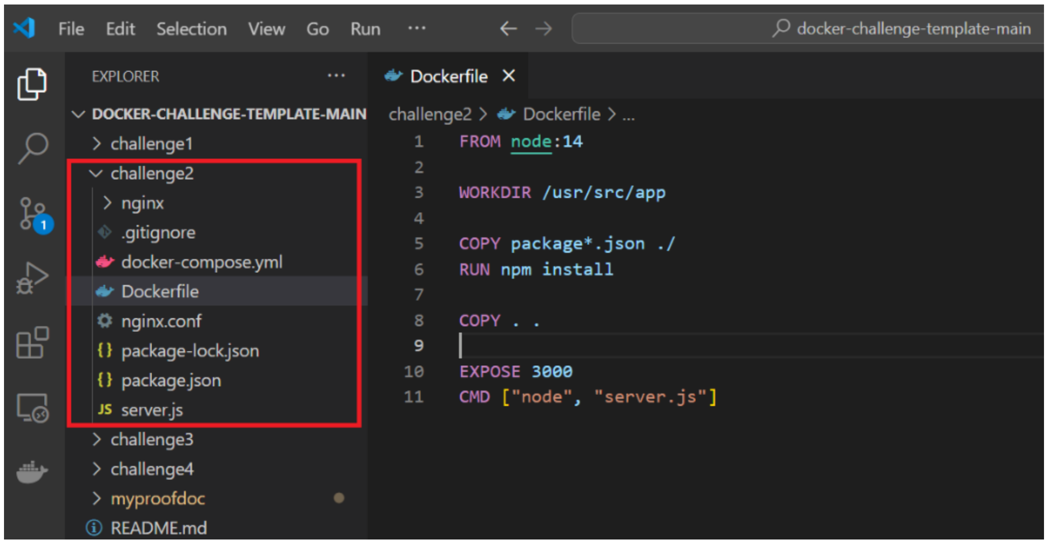
00903590

Lessons Learned*: Once the Docker image has been successfully created, you can run it using Docker Desktop to instantiate the container, making it ready to access.*

Challenge 2- NodeJS application

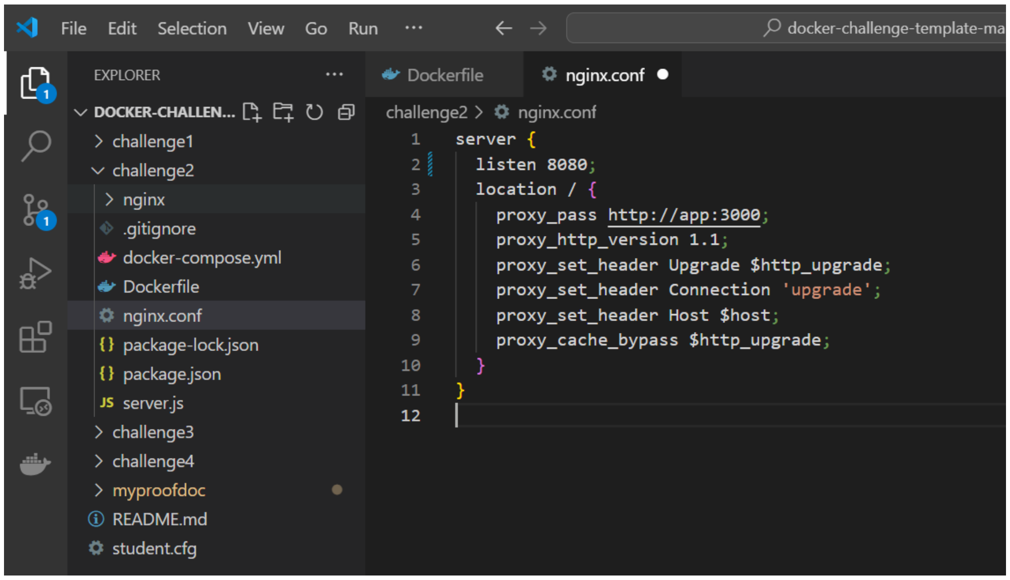
Steps

- Use the folder challenge2.
- Extract the files present on challenge2.zip to the challenge's root folder.
- Create a Dockerfile to build the server's Docker container.



```
FROM node:14

WORKDIR /usr/src/app

COPY package*.json ./
RUN npm install

COPY . .

EXPOSE 3000
CMD ["node", "server.js"]
```

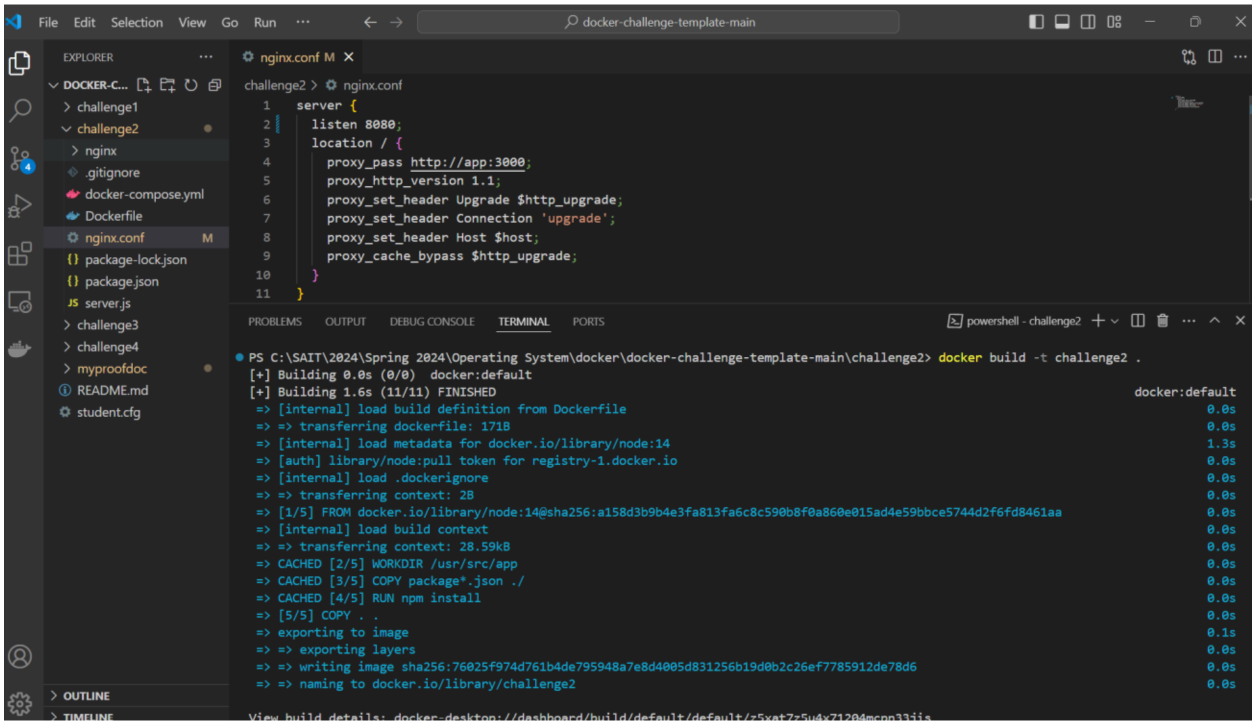
- Create the Docker compose file using NGinx and the API server from the previous step.
  - NGinx should listen on port 8080.



```
server {
  listen 8080;
  location / {
    proxy_pass http://app:3000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
  }
}
```

***Lessons Learned**: This ensures that NGinx is configured to handle requests on port 8080, acting as a gateway or reverse proxy to your API server.*

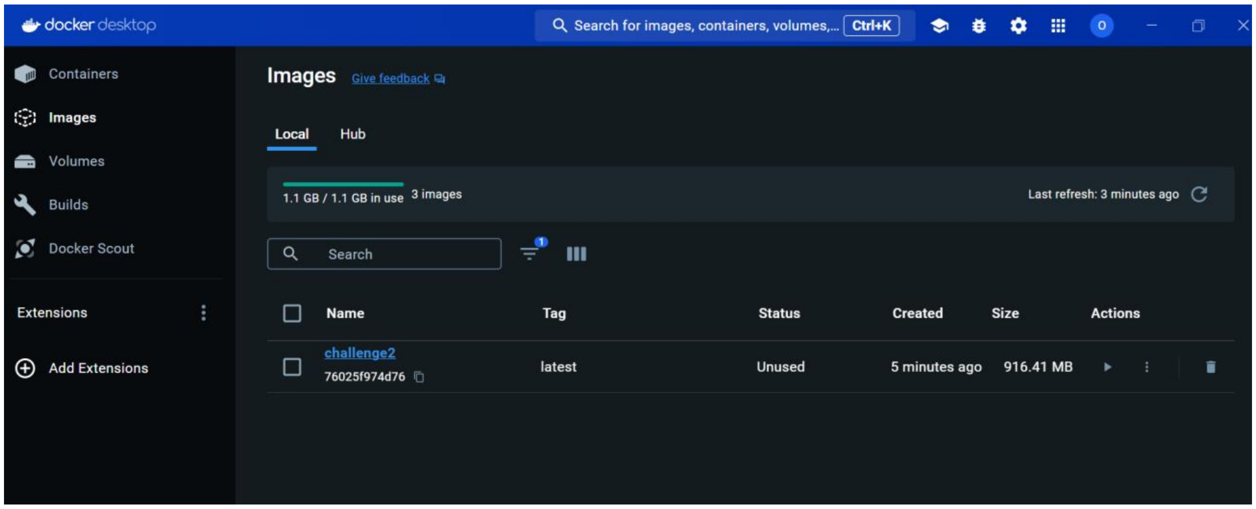
- *Command to build the image*: ***docker build -t <name of the image> .***



```
server {
  listen 8080;
  location / {
    proxy_pass http://app:3000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection 'upgrade';
    proxy_set_header Host $host;
    proxy_cache_bypass $http_upgrade;
  }
}
```

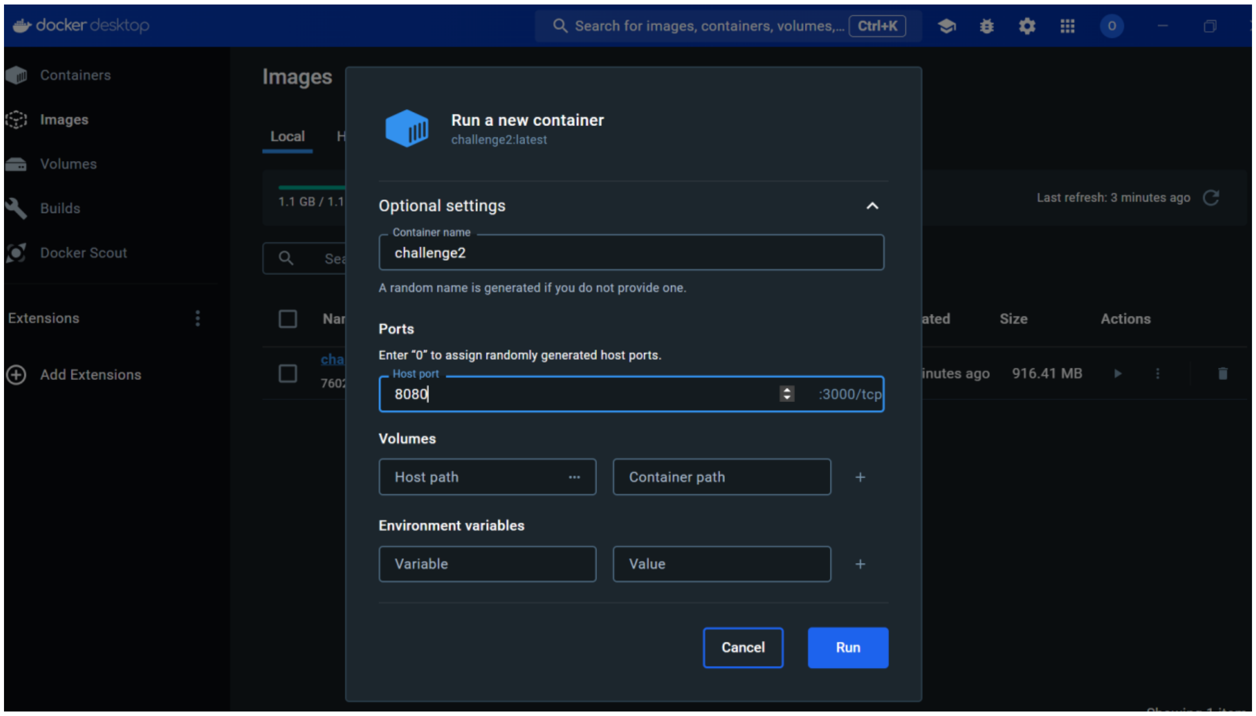
```
PS C:\SAIT\2024\Spring 2024\Operating System\docker\docker-challenge-template-main\challenge2> docker build -t challenge2 .
[+] Building 0.0s (0/0)  docker:default
[+] Building 1.6s (11/11) FINISHED                                                                          docker:default
 => [internal] load build definition from Dockerfile                                                                  0.0s
 => => transferring dockerfile: 171B                                                                                  0.0s
 => [internal] load metadata for docker.io/library/node:14                                                            1.3s
 => [auth] library/node:pull token for registry-1.docker.io                                                           0.0s
 => [internal] load .dockerignore                                                                                     0.0s
 => => transferring context: 2B                                                                                       0.0s
 => [1/5] FROM docker.io/library/node:14@sha256:a158d3b9b4e3fa813fa6c8c590b8f0a860e015ad4e59bbce5744d2f6fd8461aa      0.0s
 => [internal] load build context                                                                                     0.0s
 => => transferring context: 28.59kB                                                                                  0.0s
 => CACHED [2/5] WORKDIR /usr/src/app                                                                                 0.0s
 => CACHED [3/5] COPY package*.json ./                                                                                0.0s
 => CACHED [4/5] RUN npm install                                                                                      0.0s
 => [5/5] COPY . .                                                                                                    0.0s
 => exporting to image                                                                                                0.1s
 => => exporting layers                                                                                               0.0s
 => => writing image sha256:76025f974d761b4de795948a7e8d4005d831256b19d0b2c26ef7785912de78d6                          0.0s
 => => naming to docker.io/library/challenge2                                                                         0.0s

View build details: docker-desktop://dashboard/build/default/default/z5xat7z5u4x71204mcpn33jis
```

- *Once the build is successfully executed, open the Docker desktop to view the image created*



Name	Tag	Status	Created	Size	Actions
challenge2 76025f974d76	latest	Unused	5 minutes ago	916.41 MB	

- Run the image to generate the container and set the port to 8080



Run a new container
challenge2:latest

Optional settings

Container name: challenge2

A random name is generated if you do not provide one.

Ports

Enter "0" to assign randomly generated host ports.

Host port: 8080 :3000/tcp

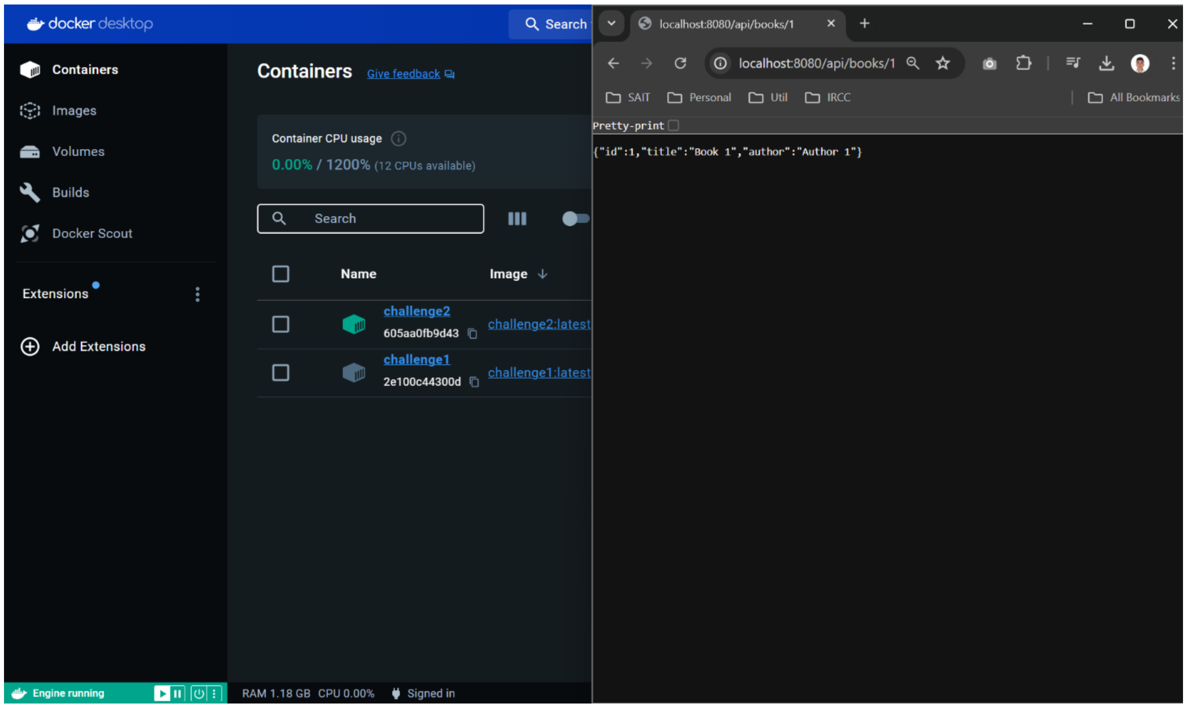
Volumes

Environment variables

- Now it's ready to run inside the container.

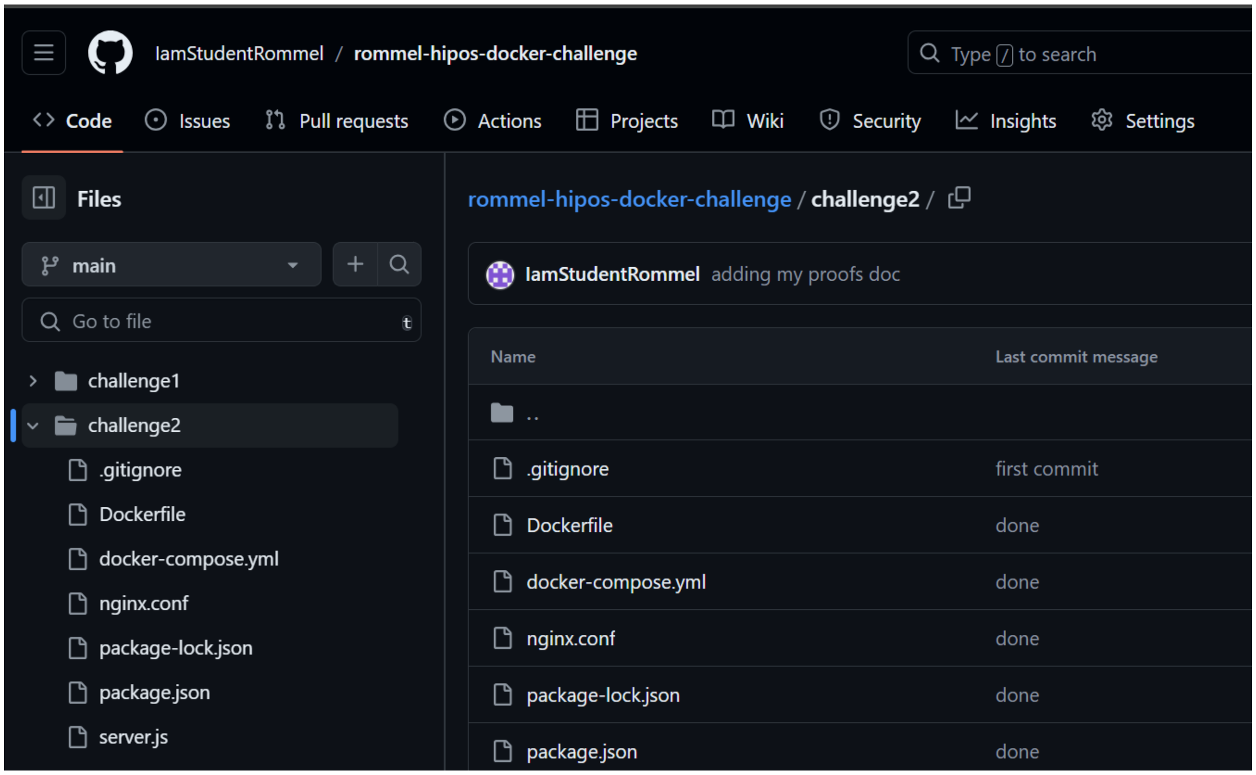
***Lessons Learned**: The goal of all these steps is to set up and get ready a development or deployment environment inside the challenge2 directory. These consist of establishing the directory structure, incorporating required project resources from a compressed file, and setting up a Dockerfile to enable the deployment of containerized servers.*

- Open a browser and point it to the address http://localhost:8080/api/books.



```
{"id":1,"title":"Book 1","author":"Author 1"}
```

- Commit all files and push them to the remote repository.



IamStudentRommel / rommel-hipos-docker-challenge

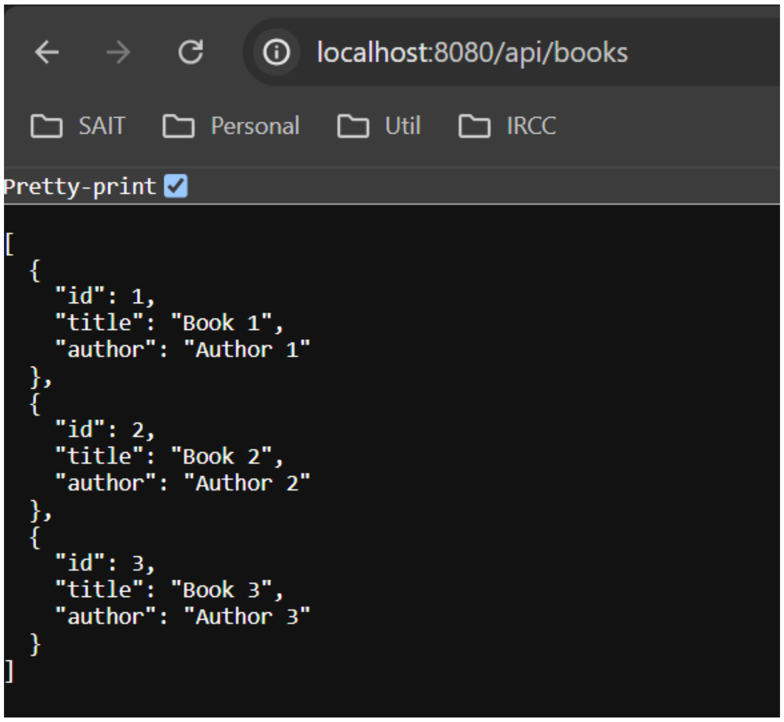
rommel-hipos-docker-challenge / challenge2 /

IamStudentRommel adding my proofs doc

Name	Last commit message
..	
.gitignore	first commit
Dockerfile	done
docker-compose.yml	done
nginx.conf	done
package-lock.json	done
package.json	done

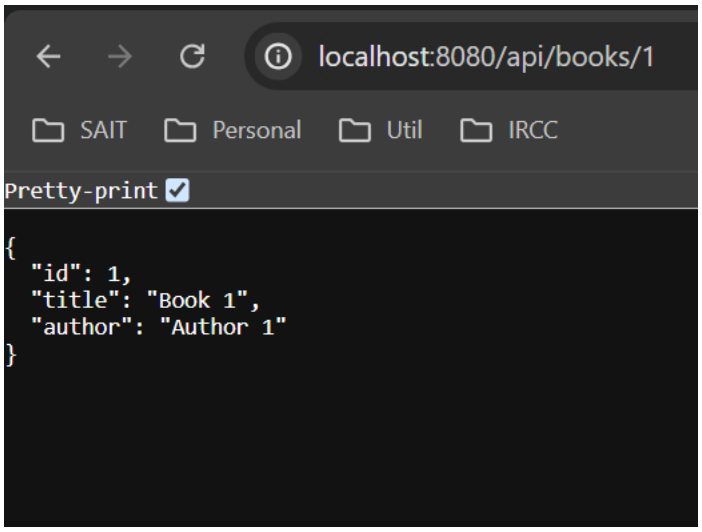
Expected outcomes

- When you access the following URLs:
    - "http://localhost:8080/api/books" you will get a JSON message with all books.



```
[
  {
    "id": 1,
    "title": "Book 1",
    "author": "Author 1"
  },
  {
    "id": 2,
    "title": "Book 2",
    "author": "Author 2"
  },
  {
    "id": 3,
    "title": "Book 3",
    "author": "Author 3"
  }
]
```

    - "http://localhost:8080/api/books/1" you will get a JSON message with just one book.

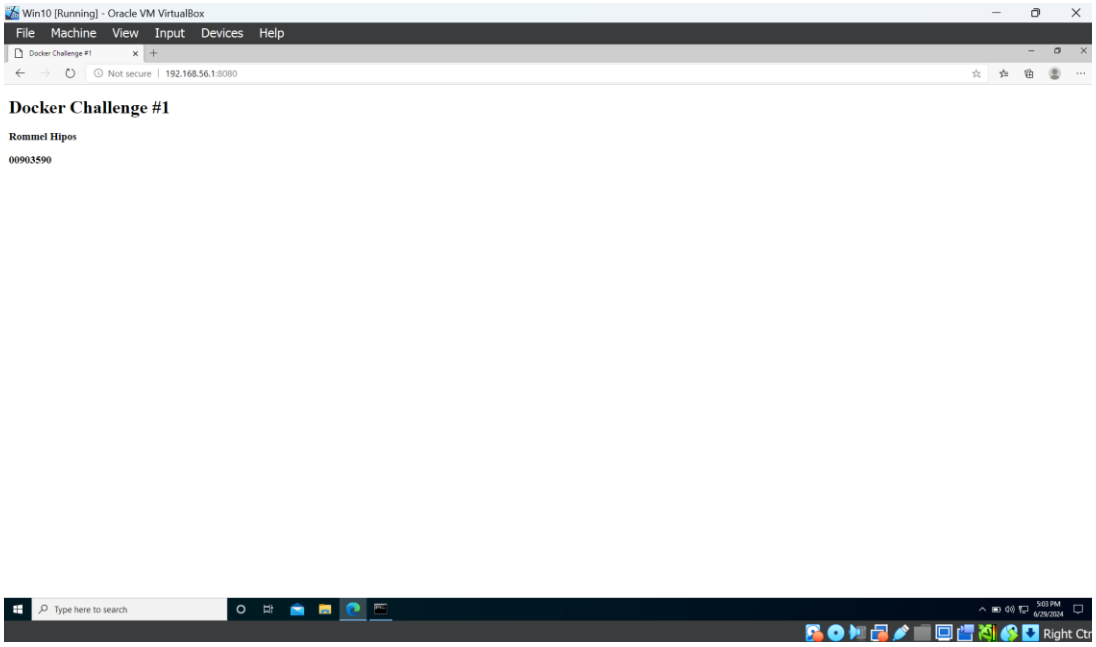


```
{
  "id": 1,
  "title": "Book 1",
  "author": "Author 1"
}
```

Test Access Using my local VM:

1. Windows VM

   a. Challenge 1



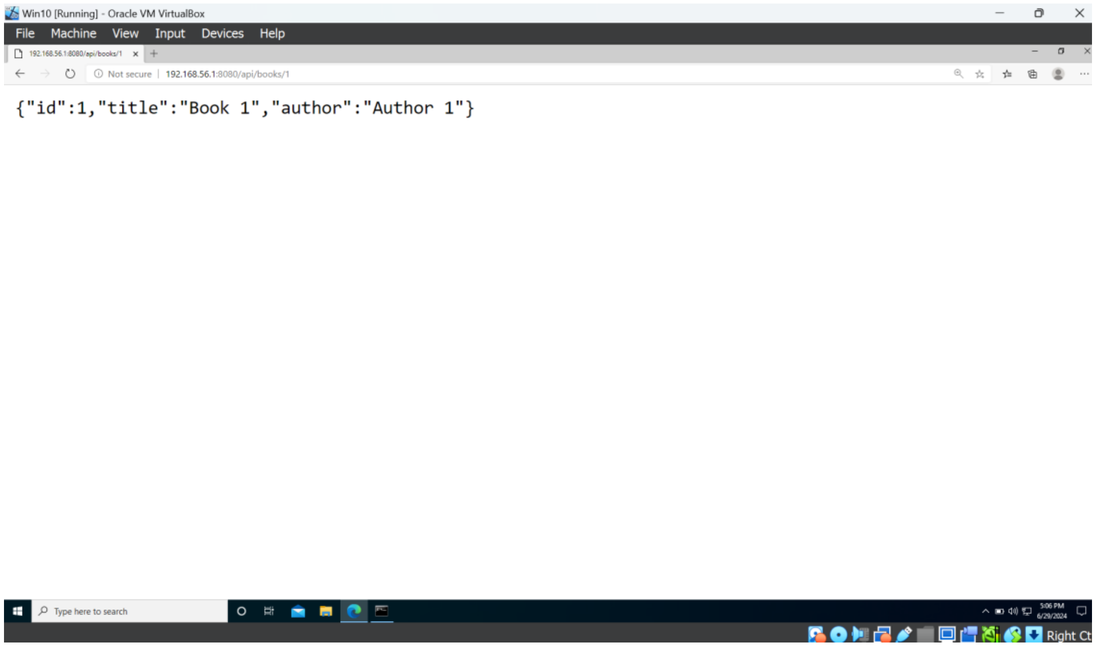
Win10 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Docker Challenge #1
Not secure | 192.168.56.1:8080
Type here to search
Right Ctrl

Docker Challenge #1

Rommel Hipos

00903590

   b. Challenge 2

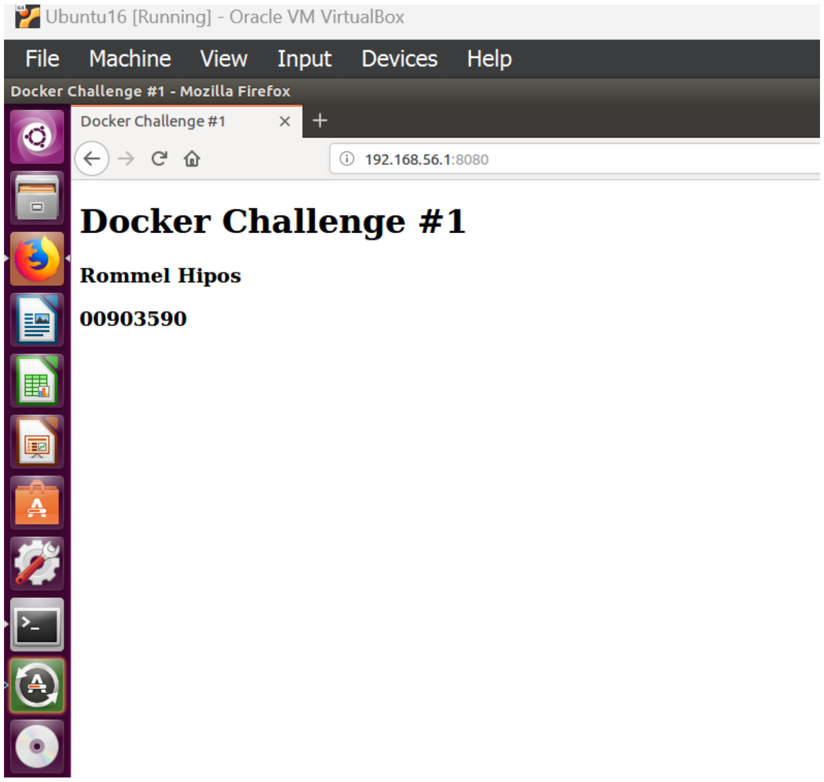


Win10 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
192.168.56.1:8080/api/books/1
Not secure | 192.168.56.1:8080/api/books/1
Type here to search
Right Ctrl

```
{"id":1,"title":"Book 1","author":"Author 1"}
```

2. Linux VM

a. Challenge 1

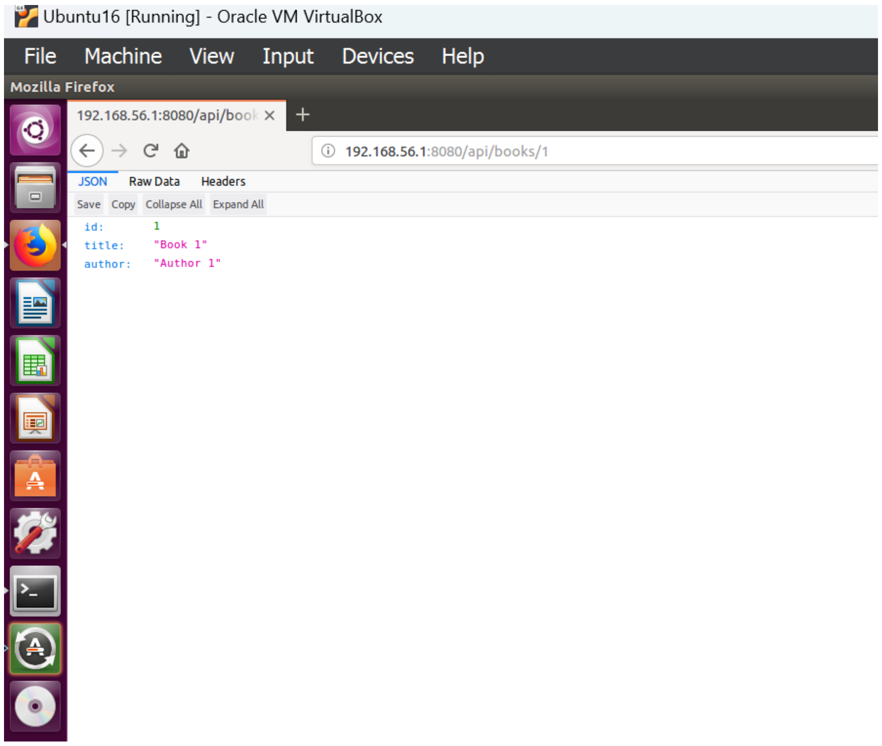

Ubuntu16 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Docker Challenge #1 - Mozilla Firefox
Docker Challenge #1
192.168.56.1:8080

Docker Challenge #1

Rommel Hipos

00903590

b. Challenge 2


Ubuntu16 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Mozilla Firefox
192.168.56.1:8080/api/book
192.168.56.1:8080/api/books/1
JSON Raw Data Headers
Save Copy Collapse All Expand All

```
id:       1
title:    "Book 1"
author:   "Author 1"
```

Prerequisites:

1. Install IDE to load the challenge template, in my case I use visual code as my editor.
   https://code.visualstudio.com/

2. Install Docker Desktop based on your current operating system.
   https://www.docker.com/products/docker-desktop/

3. This is optional, I used Postman to easily test the API (Challenge2). Postman is a powerful and user-friendly tool that allows developers to design, test, and document APIs.
   https://www.postman.com/downloads/

References

[1] [Online]. Available: https://docs.docker.com/.

[2] [Online]. Available: https://www.docker.com/blog/getting-started-with-docker-desktop/#:~:text=Docker%20Desktop%20makes%20collaboration%20easy,Pausing%20and%20resuming%20a%20container.

[3] [Online]. Available: https://github.com/IamStudentRommel/rommel-hipos-docker-challenge.git.

[4] [Online]. Available: https://medium.com/@ajitfawade/how-to-create-a-docker-project-for-a-node-js-web-application-90-days-of-devops-e3623f46bf7.

[5] [Online]. Available: https://medium.com/@ajitfawade/how-to-create-a-docker-project-for-a-node-js-web-application-90-days-of-devops-e3623f46bf7.

[6] [Online]. Available: https://www.docker.com/blog/how-to-use-the-official-nginx-docker-image/.