

Міністерство освіти і науки України
Національний технічний університет України „КПІ
імені Ігоря Сікорського ”
Факультет інформатики та обчислювальної техніки
Кафедра інформатики і програмної інженерії

ЗВІТ
лабораторної роботи № 2
з курсу «Інфраструктура програмного забезпечення WEB-
застосунків»

Тема: «Дослідження спільних ресурсів хостової та гостьової систем в
Docker»

Перевірив:

Викл. Орленко Сергій Петрович

Виконали:

Борисик Владислав

Мешков Андрій

Мочалов Дмитро

Київ 2024

1. Завдання

Мета роботи: полягає у дослідженні специфіки запуску Docker контейнерів, ознайомленні з репозиторієм Docker Hub та, за потреби, Docker Desktop. Навчитися прокидати порти з гостьової на хостову машини, що дасть змогу працювати з власним веб-сервером nginx.

Вхідні дані ЛР2

У якості вхідних даних для ЛР2 є:

– виконана ЛР1 та її Docker-образи.

Додатково слід пам'ятати що в цій лабораторній роботі:

*– без символів “<>”;

** – прізвища AAA=1+1+1=3=8003, а ZYXZ=26+25+24+26=101=8101.

Вихідні дані ЛР2

У якості вихідних даних для ЛР2 є: робочий контейнер з веб-сервером nginx, звіт.

Завдання

1. Навчитися використовувати спільні ресурси хостової та гостьової систем на прикладі Docker nginx.
2. Переглянути контет сторінок Docker nginx з хостової машини.

Хід роботи

1. Для того, щоб на сайт Nginx, який працює всередині контейнера, можна було зайти через певний порт на хостовій машині, потрібно прокинути порт.

$$B + M + M = 2 + 13 + 13 = 28 \rightarrow 8028$$

`docker run -p 8028:80 lab01_1brbmm` – запускає Docker-контейнер з прокиданням порту.

- `-p 8028:80` — прокидання порту 8028 на хостовій машині до порту 80 всередині контейнера.

Перевіримо сторінку за посиланням <http://127.0.0.1:8028/>.

На рисунках 1.1-1.3 показані ці команди та результати виконання

```
(base) ➔ lab1 docker run -p 8027:80 lab01_1brbmm
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/09/29 13:57:12 [notice] 1#1: using the "epoll" event method
2024/09/29 13:57:12 [notice] 1#1: nginx/1.27.1
2024/09/29 13:57:12 [notice] 1#1: built by gcc 13.2.1 20240309 (Alpine 13.2.1_git20240309)
2024/09/29 13:57:12 [notice] 1#1: OS: Linux 6.4.16-linuxkit
2024/09/29 13:57:12 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/09/29 13:57:12 [notice] 1#1: start worker processes
2024/09/29 13:57:12 [notice] 1#1: start worker process 30
2024/09/29 13:57:12 [notice] 1#1: start worker process 31
2024/09/29 13:57:12 [notice] 1#1: start worker process 32
2024/09/29 13:57:12 [notice] 1#1: start worker process 33
2024/09/29 13:57:12 [notice] 1#1: start worker process 34
2024/09/29 13:57:12 [notice] 1#1: start worker process 35
2024/09/29 13:57:12 [notice] 1#1: start worker process 36
2024/09/29 13:57:12 [notice] 1#1: start worker process 37
```

Рисунок 1.1 – команди та результати виконання

```
bash: curl: command not found
(base) ➔ ~ curl http://127.0.0.1:8027

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
(base) ➔ ~
```

Рисунок 1.2 – команди та результати виконання

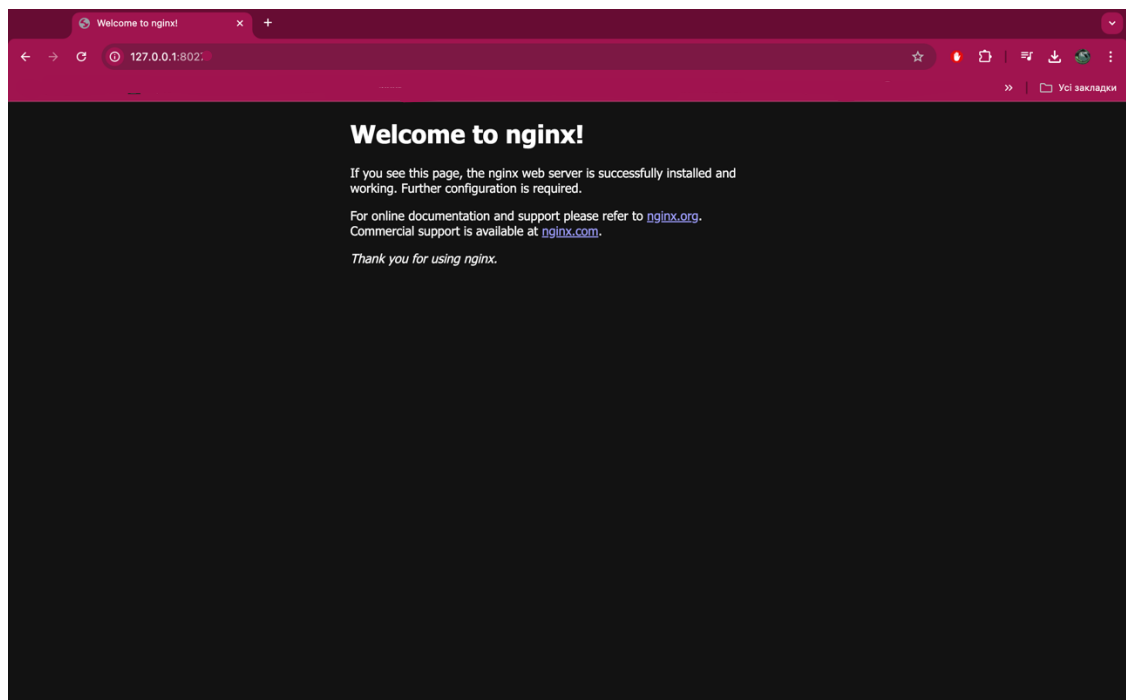


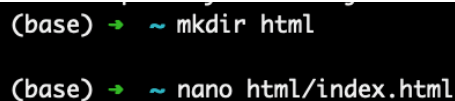
Рисунок 1.3 – веб-сторінка

2. Щоб замінити контент дефолтної сторінки Nginx на власний, включаючи перелік ПБ членів бригади і поточну дату створення образу, потрібно внести зміни в Dockerfile і

створити новий образ.

`mkdir html` – створимо папку для html файлу.

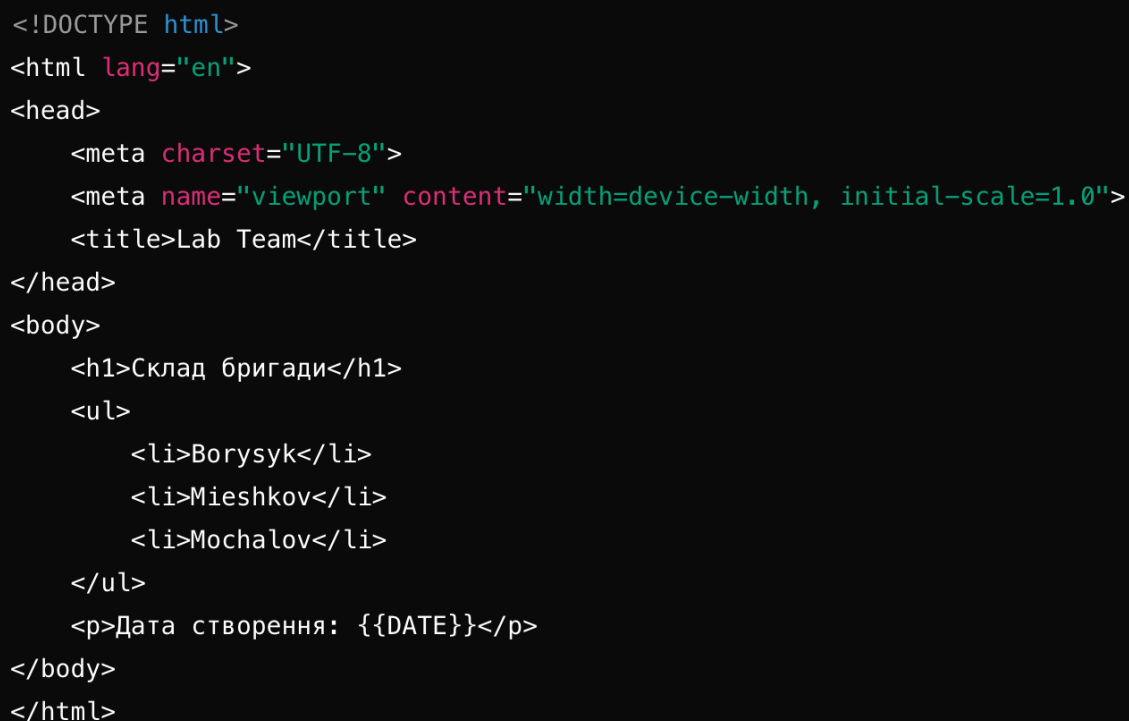
`nano html/index.html` – створимо html файл.



```
(base) → ~ mkdir html
(base) → ~ nano html/index.html
```

Рисунок 1.4 – команди та результати виконання

Наповнимо сторінку кодом.

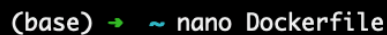


```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Lab Team</title>
</head>
<body>
  <h1>Склад бригади</h1>
  <ul>
    <li>Borysyk</li>
    <li>Mieshkov</li>
    <li>Mochalov</li>
  </ul>
  <p>Дата створення: {{DATE}}</p>
</body>
</html>
```

Рисунок 1.5 – код нової веб-сторінки

Тепер необхідно оновити Dockerfile, щоб замінити дефолтну сторінку Nginx на вашу HTML сторінку.

`nano Dockerfile` - відкрийте Dockerfile для редагування. Внесемо новий зміст.



```
(base) → ~ nano Dockerfile
```

Рисунок 1.6 – команди та результати виконання

```
FROM nginx:alpine

COPY html/index.html /usr/share/nginx/html/index.html
RUN sed -i "s/{{DATE}}/${date +"%Y-%m-%d"}/" /usr/share/nginx/html/index.html
```

A

Рисунок 1.7 – новий зміст Docker файлу

Створимо новий Docker-образ з ім'ям lab01_2bbmm (де bmm — це перші літери прізвищ членів бригади).

```
docker build -t lab01_2bbmm .
```

```
docker run -p 8027:80 lab01_2bbmm
```

 – запустимо контейнер.

Перейдемо у браузер і відкриємо сторінку <http://127.0.0.1:8028>. Ми повинні побачити сторінку з переліком членів бригади та датою створення.

```
(base) ➔ ~ docker build -t lab01_2bbmm .

[+] Building 1.8s (7/7) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 111B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/nginx:alpine
=> [internal] load build context
=> => transferring context: 458B
=> CACHED [1/2] FROM docker.io/library/nginx:alpine@sha256:a5127daff3d6f4606be3100a252419bfa84fd6eeScd74d0feaca1a5068f97dcf
=> [2/2] COPY html/index.html /usr/share/nginx/html/index.html
=> exporting to image
=> => exporting layers
=> => writing image sha256:cfe4c910a0e7f813367fdbf0d0a6d734121e281b04d409c4526deb9191034482
=> => naming to docker.io/library/lab01_2bbmm

What's Next?
 1. Sign in to your Docker account → docker login
 2. View a summary of image vulnerabilities and recommendations → docker scout quickview
(base) ➔ ~
```

Рисунок 1.8 – команди та результати виконання

```

(base) → lab1 docker run -p 8027:80 lab01_2bbmm
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/09/29 13:57:12 [notice] 1#1: using the "epoll" event method
2024/09/29 13:57:12 [notice] 1#1: nginx/1.27.1
2024/09/29 13:57:12 [notice] 1#1: built by gcc 13.2.1 20240309 (Alpine 13.2.1_git20240309)
2024/09/29 13:57:12 [notice] 1#1: OS: Linux 6.4.16-linuxkit
2024/09/29 13:57:12 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/09/29 13:57:12 [notice] 1#1: start worker processes
2024/09/29 13:57:12 [notice] 1#1: start worker process 30
2024/09/29 13:57:12 [notice] 1#1: start worker process 31
2024/09/29 13:57:12 [notice] 1#1: start worker process 32
2024/09/29 13:57:12 [notice] 1#1: start worker process 33
2024/09/29 13:57:12 [notice] 1#1: start worker process 34
2024/09/29 13:57:12 [notice] 1#1: start worker process 35
2024/09/29 13:57:12 [notice] 1#1: start worker process 36
2024/09/29 13:57:12 [notice] 1#1: start worker process 37

```

Рисунок 1.9 – команди та результати виконання



Рисунок 1.10 – новий вміст сторінки

3.У каталозі `./lab1` на локальній машині створемо HTML-файл з потрібною інформацією.

`nano index.html` – створемо файл.

Занесемо у файл код сторінки. Збережемо зміни у файлі.

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Команда</title>
</head>
<body>
  <h1>Склад бригади</h1>
  <ul>
    <li>Borysyk: Хоби – футбол</li>
    <li>Mieshkov: Хоби – програмування</li>
    <li>Mochalov: Хоби – подорожі</li>
  </ul>

</body>
</html>

```

Рисунок 1.11 – код сторінки

Відкриємо Dockerfile для редагування. Оновимо Dockerfile так, щоб він використовував VOLUME для розшарювання веб-застосунку з папки ./lab1 в контейнер.

```

FROM nginx:alpine
VOLUME /usr/share/nginx/html
WORKDIR /usr/share/nginx/html

```

Рисунок 1.12 – зміст Dockerfile

`docker build -t lab01_3brbmm .` – Побудуємо новий образ з оновленим Dockerfile.

`docker run -p 8027:80 -v $(pwd)/lab01:/usr/share/nginx/html lab01_3brbmm` – запустимо контейнер.

- `-v $(pwd)/lab1:/usr/share/nginx/html` — розшарює папку ./lab1 з нашим HTML-

контентом на хостовій машині в папку Nginx всередині контейнера.

```
(base) ➔ lab1 docker build -t lab01_3brbmm .
[*] Building 1.5s (6/6) FINISHED
[+] Building 1.5s (6/6) FINISHED
--> [internal] load .dockerignore
--> [internal] load build definition from Dockerfile
--> [internal] load metadata for docker.io/library/nginx:alpine
--> CACHED [1/2] FROM docker.io/library/nginx:alpine@sha256:a5127daff3d6f4600be310ba252419bf84f6be5cd74d0feca105068f97dcf
--> [2/2] WORKDIR /usr/share/nginx/html
--> exporting layers
--> writing image sha256:27f6ad2b7d64f3f887c6a7d6f576f9155ec722cf42e68cf520bc3381cf4135
--> naming to docker.io/library/lab01_3brbmm
What's Next?
1. Sign in to your Docker account • docker login
2. View a summary of image vulnerabilities and recommendations • docker scout quickview
(base) ➔ lab1
```

Рисунок 1.13 – команди та результати виконання

```
(base) ➔ docker run -p 8027:80 -v $(pwd)/lab1:/usr/share/nginx/html lab01_3brbmm
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2024/09/29 14:33:16 [notice] 1#1: using the "epoll" event method
2024/09/29 14:33:16 [notice] 1#1: nginx/1.27.1
2024/09/29 14:33:16 [notice] 1#1: built by gcc 13.2.1 20240309 (Alpine 13.2.1_git20240309)
2024/09/29 14:33:16 [notice] 1#1: OS: Linux 6.4.16-linuxkit
2024/09/29 14:33:16 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2024/09/29 14:33:16 [notice] 1#1: start worker processes
2024/09/29 14:33:16 [notice] 1#1: start worker process 30
2024/09/29 14:33:16 [notice] 1#1: start worker process 31
2024/09/29 14:33:16 [notice] 1#1: start worker process 32
2024/09/29 14:33:16 [notice] 1#1: start worker process 33
2024/09/29 14:33:16 [notice] 1#1: start worker process 34
2024/09/29 14:33:16 [notice] 1#1: start worker process 35
2024/09/29 14:33:16 [notice] 1#1: start worker process 36
2024/09/29 14:33:16 [notice] 1#1: start worker process 37
```

Рисунок 1.14 – команди та результати виконання

Перевіримо нову веб-сторінку.



Рисунок 1.15 – новий вміст сторінки

ВИСНОВКИ

У процесі виконання лабораторної роботи були отримані практичні навички з використання Docker для створення, запуску та управління контейнерами. Зокрема, було продемонстровано, як створити Docker-образ на основі офіційного образу Nginx, а також як використовувати команду `'docker run'` для запуску контейнера з перенаправленням портів.

В процесі лабораторної роботи було досліджено, як Nginx працює як веб-сервер. Було проведено тестування веб-сайту, розгорнутого в контейнері, з можливістю доступу через браузер. Це допомогло краще зрозуміти принципи роботи веб-серверів та особливості налаштування.

Лабораторна робота включала створення спільного ресурсу між хостовою і гостьовою системами через Docker Volume. Це дозволило зберегти контент веб-сайту на локальному комп'ютері і з легкістю оновлювати його без необхідності перебудови Docker-образу.

Було продемонстровано, як за допомогою команди `'RUN'` у Dockerfile можна динамічно оновлювати контент веб-сторінки, вставляючи поточну дату. Це підтверджує гнучкість Docker у створенні і налаштуванні середовища.

Лабораторна робота надала можливість не лише теоретично ознайомитися з Docker та Nginx, але й на практиці впровадити отримані знання. Дослідження спільних ресурсів між хостовою та гостьовою системами продемонструвало потужність та гнучкість контейнеризації у розробці та розгортанні веб-додатків.