

**Міністерство освіти і науки України Національний
технічний університет України «Київський політехнічний
інститут імені Ігоря Сікорського» Факультет
інформатики та обчислювальної техніки Кафедра
обчислювальної техніки**

Лабораторна робота №1

з дисципліни

«Інфраструктура програмного забезпечення WEB-застосунків»

Виконав:

студент групи ІП-05

Гапій Денис Едуардович

Перевірив:

Орленко С. П.

Київ 2023

Тема: «Дослідження системи контейнерів Docker»

Мета: полягає у дослідженні специфіки запуску Docker контейнерів, ознайомленні з репозиторієм Docker Hub та, за потреби, Docker Desktop

Виконання:

1. На базі alpine/nginx створити власний image NAME=lab01_1brgap.

Створення докер-файлу:

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure>mkdir Lab1
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure>cd Lab1
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab1>echo "FROM nginx:alpine" >> Dockerfile
```

```
Dockerfile
S: > Dev > Studying > KPI-Studying > 7th semester > Software Infrastructure > Lab1 > Dockerfile
1 FROM nginx:alpine
2
```

Неуспішний білд через те що не запустив спершу Docker Dektop (працюю на Windows):

```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab1>docker build -t lab01_1brgap .
ERROR: error during connect: this error may indicate that the docker daemon is not running: Get "http://%2F%2F.%2Fpipe%2Fdocker_engine/_ping": open //./pipe/docker_engine: The system cannot find the file specified.

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab1>docker build -t lab01_1brgap .
[+] Building 0.2s (2/2) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile             0.1s
=> => transferring dockerfile: 59B                               0.0s
=> [internal] load .dockerignore                               0.1s
=> => transferring context: 2B                                     0.0s
Dockerfile:1
-----
1 | >>> "FROM nginx:alpine"
2 |
-----
ERROR: failed to solve: dockerfile parse error on line 1: unknown instruction: "FROM (did you mean FROM?)"

View build details: docker-desktop://dashboard/build/default/default/klindsfsvxxhwt6gqxw41q4v
```

Успішний білд:

```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab1>docker build -t lab01_lbrgap .
[+] Building 10.9s (6/6) FINISHED
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 56B
=> [internal] load metadata for docker.io/library/nginx:alpine
=> [auth] library/nginx:pull token for registry-1.docker.io
=> [1/1] FROM docker.io/library/nginx:alpine@sha256:a59278fd22a9d411121e190b8cec8aa57b306aa333245919777583beb72
=> => resolve docker.io/library/nginx:alpine@sha256:a59278fd22a9d411121e190b8cec8aa57b306aa333245919777583beb72
=> => sha256:d4735778d47c0be8db66c446904aa2ba47f3e7509c0c9c3985ecb3b96bb7179f 629B / 629B
=> => sha256:a59278fd22a9d411121e190b8cec8aa57b306aa333245919777583beb728f59 6.70kB / 6.70kB
=> => sha256:2d2a2257c6e9d2e5b50d4fbeb436d8d2b55631c2a89935a425b417eb95212686 2.19kB / 2.19kB
=> => sha256:529b5644c430c06553d2e8082c6713fe19a4169c9dc2369cbb960081f52924ff 11.72kB / 11.72kB
=> => sha256:fed54a1dc458a7f591fa1c986669998655ad54d260d53691c8ef4841185883d4 1.90MB / 1.90MB
=> => sha256:c926b61bad3b94ae7351bafdc0c184c159ebf0643b085f7ef1d47ecdc7316833c 3.40MB / 3.40MB
=> => sha256:8695c106552e600555f5fc1bc2b299b420c52583bbf537e6c0468bc7821a3f7b 955B / 955B
=> => sha256:9e50a0e580b1e5240c8bf21f791b11fb7a8f3c04249f5db56f1bc72f2fa73929 1.21kB / 1.21kB
=> => extracting sha256:c926b61bad3b94ae7351bafdc0c184c159ebf0643b085f7ef1d47ecdc7316833c
=> => sha256:dffa16519b51a7abc6df8837b2ceffb699eedd09394ecfeff363ae5321cb7ad2 366B / 366B
=> => sha256:5dd532e9cec09472cd07e594cb6dce78c43ba5248310263f8f766c74b9fb6ae 1.40kB / 1.40kB
=> => sha256:fe117667dcd024947ead1f25ad99a5e522efcf3b7dbd0752b6fb5e73feffb407 12.65MB / 12.65MB
=> => extracting sha256:fed54a1dc458a7f591fa1c986669998655ad54d260d53691c8ef4841185883d4
=> => extracting sha256:d4735778d47c0be8db66c446904aa2ba47f3e7509c0c9c3985ecb3b96bb7179f
=> => extracting sha256:8695c106552e600555f5fc1bc2b299b420c52583bbf537e6c0468bc7821a3f7b
=> => extracting sha256:dffa16519b51a7abc6df8837b2ceffb699eedd09394ecfeff363ae5321cb7ad2
=> => extracting sha256:9e50a0e580b1e5240c8bf21f791b11fb7a8f3c04249f5db56f1bc72f2fa73929
=> => extracting sha256:5dd532e9cec09472cd07e594cb6dce78c43ba5248310263f8f766c74b9fb6ae
=> => extracting sha256:fe117667dcd024947ead1f25ad99a5e522efcf3b7dbd0752b6fb5e73feffb407
=> => exporting to image
=> => exporting layers
=> => writing image sha256:529b5644c430c06553d2e8082c6713fe19a4169c9dc2369cbb960081f52924ff
=> => naming to docker.io/library/lab01_lbrgap
```

View build details: [docker-desktop://dashboard/build/default/default/82zp9m2lb4fg094nx01bcp541](#)

What's Next?

View a summary of image vulnerabilities and recommendations → [docker scout quickview](#)

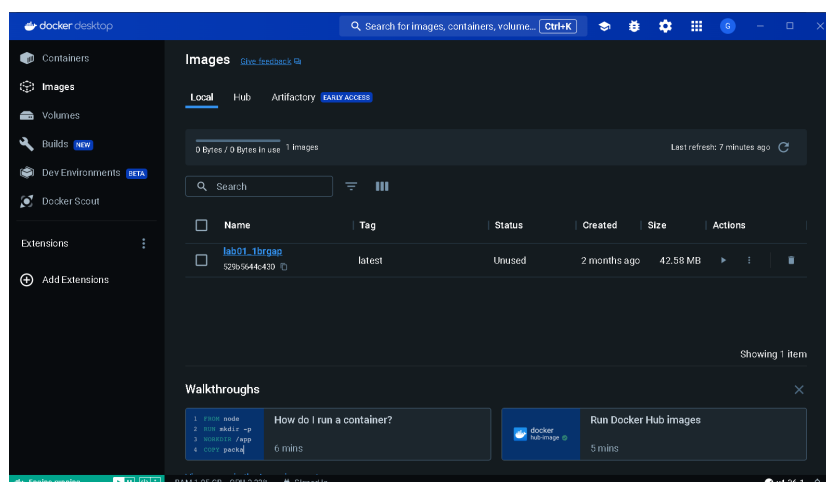
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab1>

2. Користуючись командою docker images, вивів перелік образів (images).

```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab1>docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
lab01_lbrgap        latest             529b5644c430      8 weeks ago        42.6MB

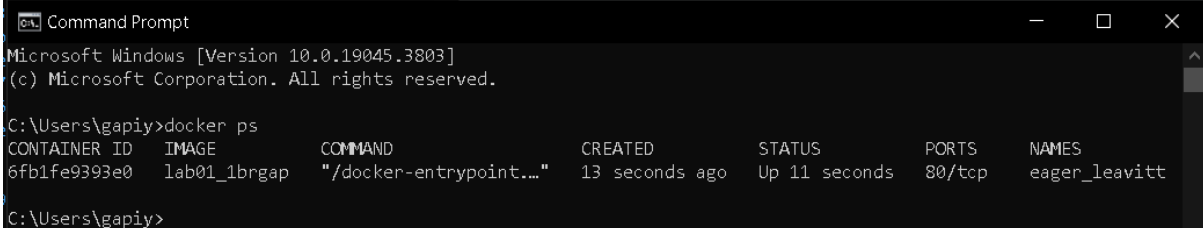
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab1>
```

Представлення у Docker Desktop



3. Запустив контейнер та вивів інформацію по активним контейнерам.

```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab1>docker run lab01_1brgap
/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
2023/12/23 12:25:36 [notice] 1#1: using the "epoll" event method
2023/12/23 12:25:36 [notice] 1#1: nginx/1.25.3
2023/12/23 12:25:36 [notice] 1#1: built by gcc 12.2.1 20220924 (Alpine 12.2.1_git20220924-r10)
2023/12/23 12:25:36 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2023/12/23 12:25:36 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/12/23 12:25:36 [notice] 1#1: start worker processes
2023/12/23 12:25:36 [notice] 1#1: start worker process 30
2023/12/23 12:25:36 [notice] 1#1: start worker process 31
2023/12/23 12:25:36 [notice] 1#1: start worker process 32
2023/12/23 12:25:36 [notice] 1#1: start worker process 33
```



```
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

C:\Users\gapiy>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS   NAMES
6fb1fe9393e0   lab01_1brgap  "/docker-entrypoint..." 13 seconds ago Up 11 seconds 80/tcp   eager_leavitt

C:\Users\gapiy>
```

4. Зупинити контейнер та вивів інформацію по активним контейнерам.

```
C:\Users\gapiy>docker stop 6fb1
6fb1
C:\Users\gapiy>docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS   NAMES
C:\Users\gapiy>docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS   NAMES
6fb1fe9393e0   lab01_1brgap  "/docker-entrypoint..." 9 minutes ago  Exited (0) 6 minutes ago  eager_leavitt

C:\Users\gapiy>
```

Відповіді на контрольні питання:

*лінивий спосіб, але зате переглядав ваші лекції)

1. Docker – основні поняття та команди.

ОСНОВНІ ПОНЯТТЯ

- **Image** - зібрана підсистема, необхідна для роботи процесу, збережена в образі.
- **Container** - процес, ініціалізований на базі образу. Тобто контейнер існує тільки коли запущений. Це як екземпляр класу, а образ це типу клас.
- **Host** - середовище, в якому запускається докер. Простіше кажучи - ваша локальна машина.
- **Volume** - це дисковий простір між хостом і контейнером. Простіше - це папка на вашій локальній машині примонтирована всередину контейнера.
- **Dockerfile** - файл з набором інструкцій для створення образу майбутнього контейнера
- **Service** - по суті це запущений образ (один або декілька контейнерів), додатково конфігурований такими опціями як відкриття портів, маппинг папок (volume) та інше. Зазвичай це робиться за допомогою docker-compose.yml файлу.
- **Docker-compose** - тулза, що полегшує збірку і запуск системи складається з декількох контейнерів, пов'язаних між собою.
- **Build** - процес створення образу з набору інструкцій в докерфайле, або декількох докерфайлов, якщо білд робиться за допомогою композер.

- **FROM** - задає базовий (батьківський) образ.
- **LABEL** - описує метадані. Наприклад - відомості про те, хто створив і підтримує образ.
- **ENV** - встановлює постійні змінні середовища.
- **RUN** - виконує команду і створює шар образу. Використовується для установки в контейнер пакетів.
- **COPY** - копіює в контейнер файли і папки.
- **ADD** - копіює файли і папки в контейнер, може розпаковувати локальні .tar-файли.
- **CMD** - описує команду з аргументами, яку потрібно виконати коли контейнер буде запущений. Аргументи можуть бути перевизначені при запуску контейнера. У файлі може бути присутнім лише одна інструкція CMD.
- **WORKDIR** - задає робочу директорию для наступної інструкції.
- **ARG** - задає змінні для передачі Docker в процесі побудови образу.
- **ENTRYPOINT** - надає команду з аргументами для виклику під час виконання контейнера. Аргументи не перевизначаються.
- **EXPOSE** - вказує на необхідність відкрити порт.
- **VOLUME** - створює точку монтування для роботи з постійним сховищем.

При запуску `docker build` - прапорець `-t` чи `--tag` відповідає за надання тегу / імені образу, а ``.` означає що збиратись буде з даної директорії.

2. Поясніть у чому відмінність між Docker IMAGE та Docker CONTAINER?

Image - шаблон / образ, а Container - одиниця / екземпляр / представлення для виконання того шаблону.

3. Що означає тег ALPINE в докер образах?

<<... мінімалістичний лінукс, що дозволяє економити пам'ять ...>>

4. Яким чином можна переглянути контейнери, що були зупинені?

Або через `docker ps`, і якщо ми там не побачимо діючого контейнера, або ж додати прапорець `-a` та побачити в колонці STATUS ``завершено / вийшли``

5. Яка роль Dockerfile при створенні образів?

Інструкція (список команд) для того де, як та яким повинен створюватися образ.

**Міністерство освіти і науки України Національний
технічний університет України «Київський політехнічний
інститут імені Ігоря Сікорського» Факультет
інформатики та обчислювальної техніки Кафедра
обчислювальної техніки**

Лабораторна робота №2

з дисципліни

«Інфраструктура програмного забезпечення WEB-застосунків»

Виконав:

студент групи ІП-05

Гапій Денис Едуардович

Перевірив:

Орленко С. П.

Київ 2023

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

Мета: 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.

Виконання:

1.Прокинули порти. В команду docker run додав такі ключі, щоб на сайт контейнера можна було зайти через 127.0.0.1:8024 (GAP = 7+1+16 = 24):

```
C:\Windows\System32\cmd.exe - docker run -p 8024:80 lab02_1brgap
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>docker build -t lab02_1brgap .
[+] Building 2.3s (6/6) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 56B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/nginx:alpine
=> [auth] library/nginx:pull token for registry-1.docker.io
=> CACHED [1/1] FROM docker.io/library/nginx:alpine@sha256:a59278fd22a9d41121e190b8cec8aa57b306aa33324591977775
=> exporting to image
=> => exporting layers
=> => writing image sha256:529b5644c430c06553d2e8082c6713fe19a4169c9dc2369cbb960081f52924ff
=> => naming to docker.io/library/lab02_1brgap

View build details: docker-desktop:///dashboard/build/default/default/w6h051qdh06nkp4cfqm5n7m0

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview

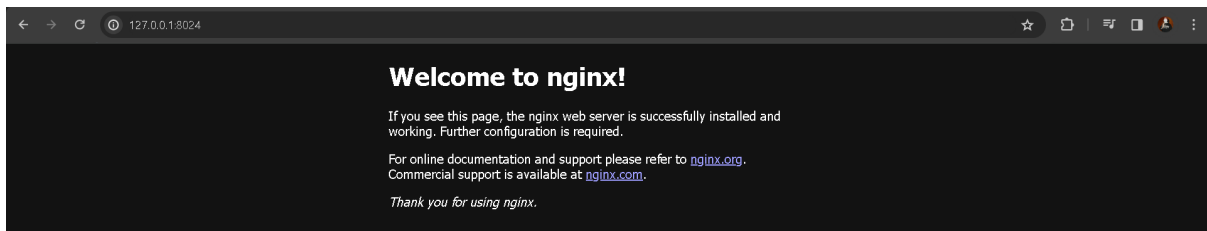
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>docker run -p 8024:80 lab02_1brgap
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /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
2023/12/23 13:09:43 [notice] 1#1: using the "epoll" event method
2023/12/23 13:09:43 [notice] 1#1: nginx/1.25.3
2023/12/23 13:09:43 [notice] 1#1: built by gcc 12.2.1 20220924 (Alpine 12.2.1_git20220924-r10)
2023/12/23 13:09:43 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2023/12/23 13:09:43 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/12/23 13:09:43 [notice] 1#1: start worker processes
2023/12/23 13:09:43 [notice] 1#1: start worker process 30
2023/12/23 13:09:43 [notice] 1#1: start worker process 31
2023/12/23 13:09:43 [notice] 1#1: start worker process 32
2023/12/23 13:09:43 [notice] 1#1: start worker process 33
```

Вивів інформацію по активним контейнерам:

```
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

C:\Users\gapiy>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
197aeca6d61b   lab02_1brgap   "/docker-entrypoint..." 2 minutes ago  Up 2 minutes  0.0.0.0:8024->80/tcp     sleepy_cars
```


Для перевірки результату через браузер гостьової машини звернувся до відповідної адреси і порту, а саме перейшов на localhost:8024 у браузері:



2. Замінив контент дефолтної сторінки nginx на власний – перелік ПІБ всіх членів бригади + поточна дата створення image. Замінив через ADD/COPY та, за потреби RUN, у Dockerfile. Створив НОВИЙ image NAME=lab02_2bgar на базі імейджу lab02_1brgar.

Дефолтний лістинг сторінки:

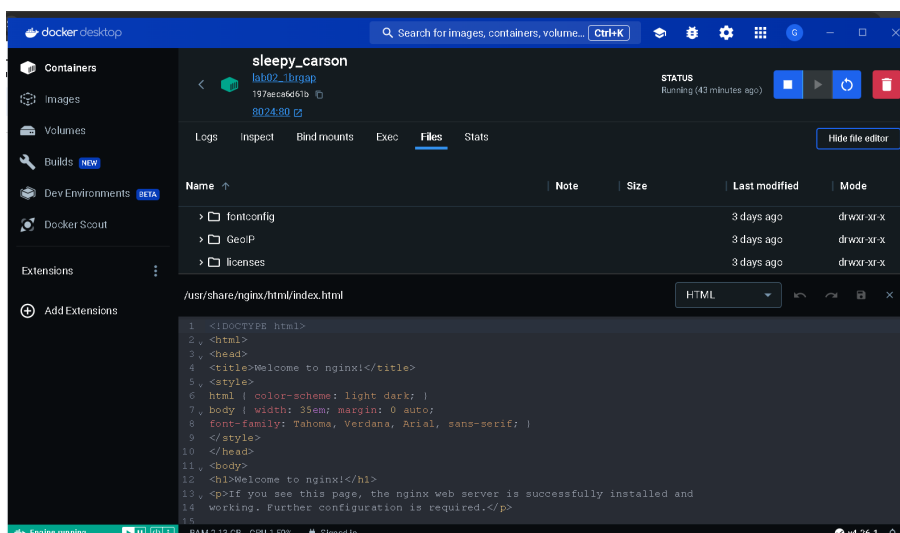
```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                    NAMES
197aeca6d61b   lab02_1brgar   "/docker-entrypoint..." 30 minutes ago Up 30 minutes 0.0.0.0:8024->80/tcp      sleepy_carson

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>docker exec -it 197a sh
/ # tail -n 1 /usr/share/nginx/html/index.html
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>
/ #
```

Перегляд через Docker Desktop:



Змінений Dockerfile:

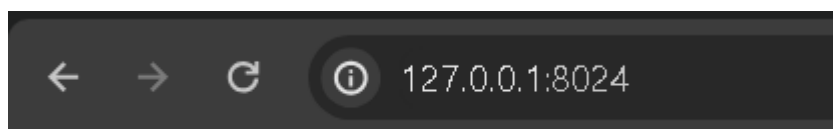
```
Dockerfile S:\...\Lab1  Dockerfile S:\...\Lab2 X
S: > Dev > Studying > KPI-Studying > 7th semester > Software Infrastructure > Lab2 > Dockerfile
1 FROM nginx:alpine
2 RUN echo "Gapiy Denis Eduardovich 23.12.2023" > /usr/share/nginx/html/index.html
```

Новий image:

```
[+] Building 1.6s (6/6) FINISHED docker:default
=> [internal] load build definition from Dockerfile 0.1s
=> => transferring dockerfile: 136B 0.0s
=> [internal] load .dockerignore 0.1s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/nginx:alpine 0.7s
=> CACHED [1/2] FROM docker.io/library/nginx:alpine@sha256:a59278fd22a9d411121e190b8cec8aa57b306aa33324591977775 0.0s
=> [2/2] RUN echo "Gapiy Denis Eduardovich 23.12.2023" > /usr/share/nginx/html/index.html 0.5s
=> exporting to image 0.1s
=> => exporting layers 0.1s
=> => writing image sha256:9726483e441c5fe6696d03eb681fbdad8d18a5983ba8ba16de1af902e39b4d31 0.0s
=> => naming to docker.io/library/lab02_2bgap 0.0s
```

```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>docker run -p 8024:80 lab02_2bgap
/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
2023/12/23 14:11:37 [notice] 1#1: using the "epoll" event method
2023/12/23 14:11:37 [notice] 1#1: nginx/1.25.3
2023/12/23 14:11:37 [notice] 1#1: built by gcc 12.2.1 20220924 (Alpine 12.2.1_git20220924-r10)
2023/12/23 14:11:37 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2023/12/23 14:11:37 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/12/23 14:11:37 [notice] 1#1: start worker processes
2023/12/23 14:11:37 [notice] 1#1: start worker process 30
2023/12/23 14:11:37 [notice] 1#1: start worker process 31
2023/12/23 14:11:37 [notice] 1#1: start worker process 32
2023/12/23 14:11:37 [notice] 1#1: start worker process 33
172.17.0.1 - - [23/Dec/2023:14:11:44 +0000] "GET / HTTP/1.1" 200 35 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36" "-"
```

Представлення сторінки у браузері:



Gapiy Denis Eduardovich 23.12.2023

3. Замінити контент дефолтної сторінки nginx на власний, але розшарений з каталогу `./lab02` – односторінковий WEB-застосунок, який презентуватиме склад бригади (з вказанням хобі, тощо), який але створить файл у папці `./lab02` і зробити так, щоб через VOLUME

цей файл ставав сторінкою. Створити НОВИЙ image NAME=lab02_3brgap на базі імейджу lab01_1brgap.

Dockerfile:

```
Dockerfile S:\...\Lab1  Dockerfile S:\...\Lab2  X  index.html
S: > Dev > Studying > KPI-Studying > 7th semester > Software Infrastructure > Lab2 > Dockerfile
1  FROM lab01_1brgap
2  COPY ./lab02 /usr/share/nginx/html/
3  VOLUME /usr/share/nginx/html
```

Лістинг сторінки:

```
Dockerfile S:\...\Lab1  Dockerfile S:\...\Lab2  X  index.html  X
S: > Dev > Studying > KPI-Studying > 7th semester > Software Infrastructure > Lab2 > lab02 > index.html > p
1  <p align="center">Gapiy Denis Eduardovich - games</p>
2  <p align="right">23.12.2023</p>
```

Успішний білд:

```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>docker build -t lab02_3brgap .
[+] Building 0.6s (7/7) FINISHED
=> [internal] load .dockerignore                                docker:default 0.1s
=> => transferring context: 2B                                   0.0s
=> [internal] load build definition from Dockerfile             0.1s
=> => transferring dockerfile: 121B                              0.0s
=> [internal] load metadata for docker.io/library/lab01_1brgap:latest 0.0s
=> [internal] load build context                                0.1s
=> => transferring context: 158B                                  0.0s
=> CACHED [1/2] FROM docker.io/library/lab01_1brgap            0.0s
=> [2/2] COPY ./lab02 /usr/share/nginx/html/                   0.1s
=> exporting to image                                           0.1s
=> => exporting layers                                           0.1s
=> => writing image sha256:b317ba586e1c6bc9b740ae9f7ad97ac99cb8d8912ee3f726a3c3e2ed1df954fd 0.0s
=> => naming to docker.io/library/lab02_3brgap                 0.0s

View build details: docker-desktop://dashboard/build/default/default/z6bsum1rzrz3y12a55ed972eb

What's Next?
  View a summary of image vulnerabilities and recommendations → docker scout quickview

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>
```

Список збірок:

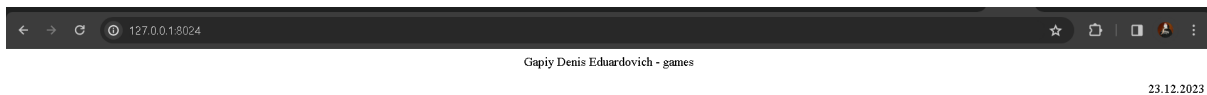
```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
lab02_3brgap        latest          b317ba586e1c   41 seconds ago  42.6MB
lab02_2bgap         latest          9726483e441c   14 minutes ago  42.6MB
lab01_1brgap        latest          529b5644c430   8 weeks ago    42.6MB
lab02_1brgap        latest          529b5644c430   8 weeks ago    42.6MB

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>
```

Запуск контейнера:

```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab2>docker run -p 8024:80 lab02_3brgap
/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
2023/12/23 14:25:09 [notice] 1#1: using the "epoll" event method
2023/12/23 14:25:09 [notice] 1#1: nginx/1.25.3
2023/12/23 14:25:09 [notice] 1#1: built by gcc 12.2.1 20220924 (Alpine 12.2.1_git20220924-r10)
2023/12/23 14:25:09 [notice] 1#1: OS: Linux 5.15.133.1-microsoft-standard-WSL2
2023/12/23 14:25:09 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/12/23 14:25:09 [notice] 1#1: start worker processes
2023/12/23 14:25:09 [notice] 1#1: start worker process 30
2023/12/23 14:25:09 [notice] 1#1: start worker process 31
2023/12/23 14:25:09 [notice] 1#1: start worker process 32
2023/12/23 14:25:09 [notice] 1#1: start worker process 33
172.17.0.1 - - [23/Dec/2023:14:25:15 +0000] "GET / HTTP/1.1" 200 86 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.0.0 Safari/537.36" "-"
```

Представлення сторінки у браузері:



Відповіді на контрольні питання:

1. Docker – інструкції Dockerfile.

- FROM - задає базовий (батьківський) образ.
- LABEL - описує метадані. Наприклад - відомості про те, хто створив і підтримує образ.
- ENV - встановлює постійні змінні середовища.
- RUN - виконує команду і створює шар образу. Використовується для установки в контейнер пакетів.
- COPY - копіює в контейнер файли і папки.
- ADD - копіює файли і папки в контейнер, може розпаковувати локальні .tar-файли.
- CMD - описує команду з аргументами, яку потрібно виконати коли контейнер буде запущений. Аргументи можуть бути перевизначені при запуску контейнера. У файлі може бути присутнім лише одна інструкція CMD.
- WORKDIR - задає робочу директорию для наступної інструкції.
- ARG - задає змінні для передачі Docker в процесі побудови образу.
- ENTRYPOINT - надає команду з аргументами для виклику під час виконання контейнера. Аргументи не перевизначаються.
- EXPOSE - вказує на необхідність відкрити порт.
- VOLUME - створює точку монтування для роботи з постійним сховищем.

2. Що таке прокидання портів та навіщо його робити при запуску докер контейнерів?

Це процес зв'язування портів контейнера з портами хоста. Це дозволяє контейнеру виходити у мереж і обмінюватися даними з іншими комп'ютерами. У нашому випадку ми робили це щоб мати змогу зайти на веб-сторінку лише за конкретним портом який ми вказали. Для цього під час створення образу варто застосувати прапорець -p чи --publish-all. Аргументи передаються у

послідовності: хостовий порт, а далі порт контейнера. Як додатковий функціонал - можемо застосувати перед цим вказати інший ip адрес, якщо не хочемо використовувати дефолтний 127.0.0.1. А також після зворотнього слешу `` задати протокол який хочемо. (Основна відмінність між TCP (transmission control protocol) і UDP (user datagram protocol) полягає в тому, що TCP - це протокол, що базується на з'єднанні, а UDP - без потреби з'єднання для початку 'комунікації'. Хоча TCP надійніший, він повільніше передає дані. UDP менш надійний, але працює швидше).

3. Що таке VOLUME у докері?

VOLUME - це спеціальний тип каталогу в контейнері, який зберігається окремо від файлової системи контейнера. VOLUME дозволяє зберігати дані, які повинні бути доступні для всіх контейнерів, що використовують цей VOLUME.

4. Що виконує інструкція RUN у Dockerfile?

Інструкція RUN є однією з найпоширеніших інструкцій у Dockerfile. Вона використовується для виконання всіх необхідних дій / команд для підготовки контейнера до запуску.

5. Яка команда дозволяє зайти в командний рядок контейнера Docker?

`docker exec (-it fa10 для конкретного ідентифікатора контейнера)`

**Міністерство освіти і науки України Національний
технічний університет України «Київський політехнічний
інститут імені Ігоря Сікорського» Факультет
інформатики та обчислювальної техніки Кафедра
обчислювальної техніки**

Лабораторна робота №3

з дисципліни

«Інфраструктура програмного забезпечення WEB-застосунків»

Виконав:

студент групи ІП-05

Гапій Денис Едуардович

Перевірив:

Орленко С. П.

Київ 2023

Тема: «Створення та керування портативними віртуальними середовищами»

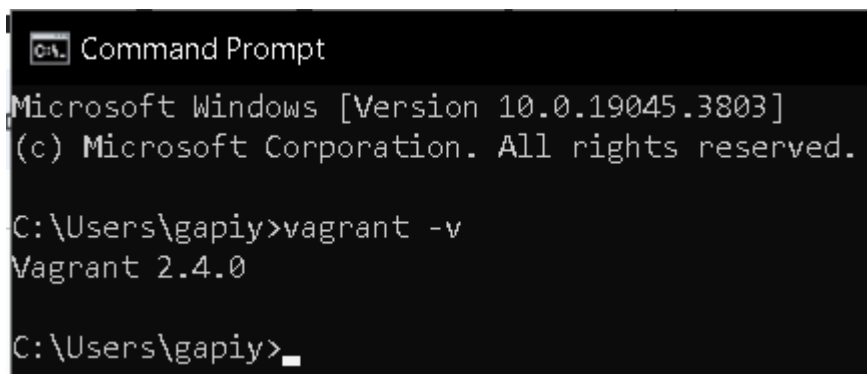
Мета: полягає у дослідженні специфіки створення та

налаштування віртуальних машин за допомогою Vagrant, ознайомленні з репозиторієм Vagrant Cloud. Навчитися прокидати порти з гостьової на хостову машини, що дасть змогу працювати з власним веб-сервером nginx. Закріпити на практиці різницю між контейнеризацією та віртуалізацією.

Виконання:

1. Інсталяція Vagrant почалася зі скачування wizzard-y з ресурсу https://developer.hashicorp.com/vagrant/install?product_intent=vagrant#Windows.

Перевірка версії:

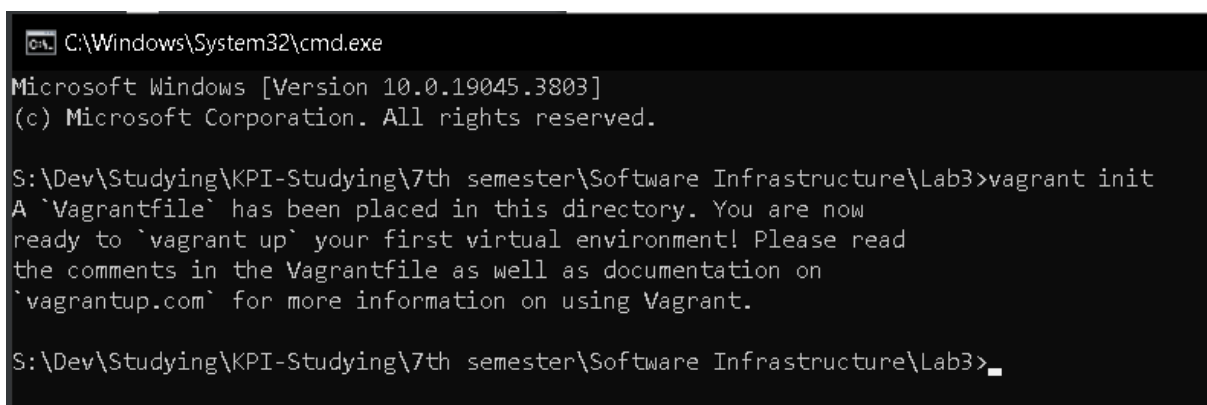


```
Command Prompt
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

C:\Users\gapiy>vagrant -v
Vagrant 2.4.0

C:\Users\gapiy>
```

Створення робочого середовища:



```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.3803]
(c) Microsoft Corporation. All rights reserved.

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab3>vagrant init
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab3>
```

2. Налаштування конфігураційного файлу Vagrantfile, з прокиданням портів:

```
Vagrantfile
S: > Dev > Studying > KPI-Studying > 7th semester > Software Infrastructure > Lab3 > Vagrantfile
1  vagrant.configure("2") do |config|
2      config.vm.box = "ubuntu/trusty64"
3      config.vm.hostname = "gap"
4      config.vm.network "forwarded_port", guest: 80, host: 8024, host_ip:"127.0.0.1"
5  end
6
```

Додаткове завантаження застосунку для віртуалізації, щоб не виникало наступної помилки. (У моєму випадку це був VirtualBox).

```
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab3>vagrant init
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab3>vagrant up
No usable default provider could be found for your system.

Vagrant relies on interactions with 3rd party systems, known as
"providers", to provide Vagrant with resources to run development
environments. Examples are VirtualBox, VMware, Hyper-V.

The easiest solution to this message is to install VirtualBox, which
is available for free on all major platforms.

If you believe you already have a provider available, make sure it
is properly installed and configured. You can see more details about
why a particular provider isn't working by forcing usage with
`vagrant up --provider=PROVIDER`, which should give you a more specific
error message for that particular provider.
```

Створення гостьової машини відповідно до Vagrantfile:


```
C:\Windows\System32\cmd.exe

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab3>vagrant up
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Box 'ubuntu/trusty64' could not be found. Attempting to find and install...
    default: Box Provider: virtualbox
    default: Box Version: >= 0
==> default: Loading metadata for box 'ubuntu/trusty64'
    default: URL: https://vagrantcloud.com/api/v2/vagrant/ubuntu/trusty64
==> default: Adding box 'ubuntu/trusty64' (v20190514.0.0) for provider: virtualbox
    default: Downloading: https://vagrantcloud.com/ubuntu/boxes/trusty64/versions/20190514.0.0/providers/virtualbox/unknown/vagrant.box
Download redirected to host: cloud-images.ubuntu.com
    default:
==> default: Successfully added box 'ubuntu/trusty64' (v20190514.0.0) for 'virtualbox'!
==> default: Importing base box 'ubuntu/trusty64'...
==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'ubuntu/trusty64' version '20190514.0.0' is up to date...
==> default: Setting the name of the VM: Lab3_default_1703360685216_5937
==> default: Clearing any previously set forwarded ports...
Vagrant is currently configured to create VirtualBox synced folders with
the 'SharedFoldersEnableSymlinksCreate' option enabled. If the Vagrant
guest is not trusted, you may want to disable this option. For more
information on this option, please refer to the VirtualBox manual:

    https://www.virtualbox.org/manual/ch04.html#sharedfolders

This option can be disabled globally with an environment variable:

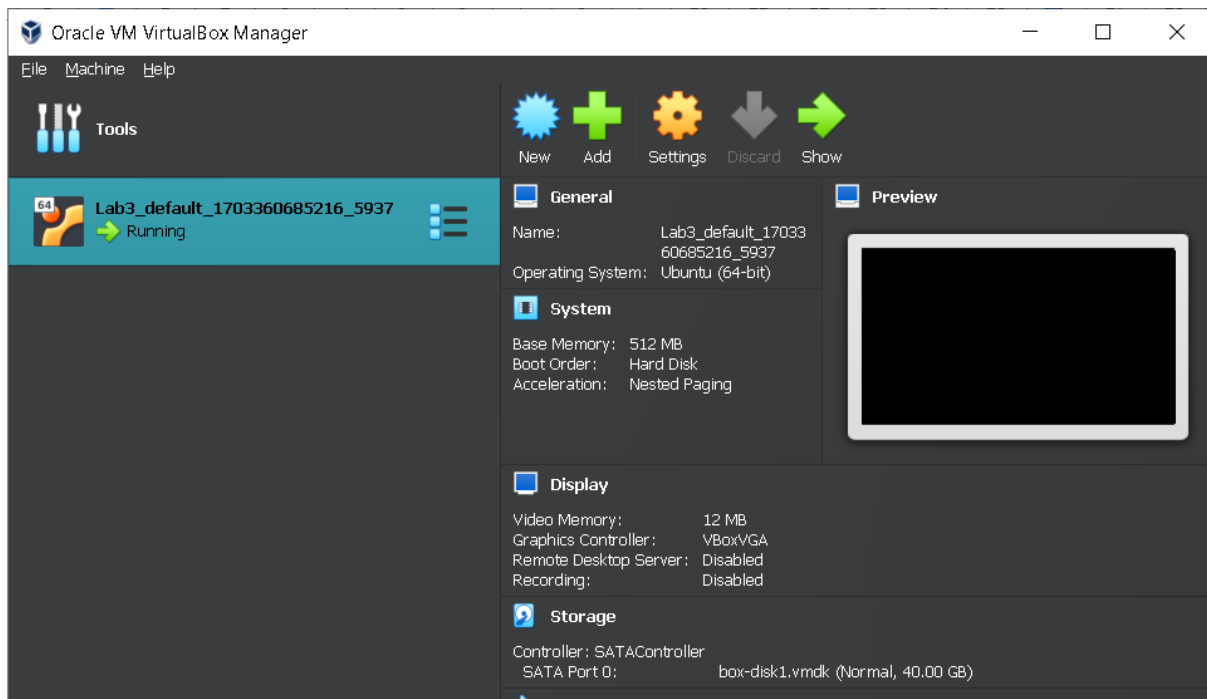
    VAGRANT_DISABLE_VBOXSYMLINKCREATE=1

or on a per folder basis within the Vagrantfile:

    config.vm.synced_folder '/host/path', '/guest/path', SharedFoldersEnableSymlinksCreate: false
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
==> default: Forwarding ports...
    default: 80 (guest) => 8024 (host) (adapter 1)
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: vagrant
    default: SSH auth method: private key
    default: Warning: Connection reset. Retrying...
    default: Warning: Connection aborted. Retrying...
    default: Warning: Connection reset. Retrying...
    default: Warning: Connection aborted. Retrying...
    default:
    default: Vagrant insecure key detected. Vagrant will automatically replace
    default: this with a newly generated keypair for better security.
    default:
    default: Inserting generated public key within guest...
    default: Removing insecure key from the guest if it's present...
    default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
    default: The guest additions on this VM do not match the installed version of
    default: VirtualBox! In most cases this is fine, but in rare cases it can
    default: prevent things such as shared folders from working properly. If you see
    default: shared folder errors, please make sure the guest additions within the
    default: virtual machine match the version of VirtualBox you have installed on
    default: your host and reload your VM.
    default:
    default: Guest Additions Version: 4.3.40
    default: VirtualBox Version: 7.0
==> default: Setting hostname...
==> default: Mounting shared folders...
    default: /vagrant => S:/Dev/Studying/KPI-Studying/7th semester/Software Infrastructure/Lab3

S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab3>
```

Створений екземпляр у віртуалці:



Підключення по ssh:

```
vagrant@gap: ~  
S:\Dev\Studying\KPI-Studying\7th semester\Software Infrastructure\Lab3>vagrant ssh default  
Welcome to Ubuntu 14.04.6 LTS (GNU/Linux 3.13.0-170-generic x86_64)  
  
* Documentation:  https://help.ubuntu.com/  
  
System information disabled due to load higher than 1.0  
  
UA Infrastructure Extended Security Maintenance (ESM) is not enabled.  
  
0 updates can be installed immediately.  
0 of these updates are security updates.  
  
Enable UA Infrastructure ESM to receive 64 additional security updates.  
See https://ubuntu.com/advantage or run: sudo ua status  
  
New release '16.04.7 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
vagrant@gap:~$
```

Встановлення nginx на створену віртуальну машину:

```
vagrant@gap: ~  
vagrant@gap:~$ sudo apt-get -y install nginx  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following extra packages will be installed:  
  libxslt1.1 nginx-common nginx-core  
Suggested packages:  
  fcgiwrap nginx-doc  
The following NEW packages will be installed:  
  libxslt1.1 nginx nginx-common nginx-core  
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.  
Need to get 495 kB of archives.  
After this operation, 1,802 kB of additional disk space will be used.  
Get:1 http://archive.ubuntu.com/ubuntu/ trusty-updates/main libxslt1.1 amd64 1.1.28-2ubuntu0.2 [146 kB]  
Get:2 http://archive.ubuntu.com/ubuntu/ trusty-updates/main nginx-common all 1.4.6-1ubuntu3.9 [18.9 kB]  
Get:3 http://archive.ubuntu.com/ubuntu/ trusty-updates/main nginx-core amd64 1.4.6-1ubuntu3.9 [325 kB]  
Get:4 http://archive.ubuntu.com/ubuntu/ trusty-updates/main nginx all 1.4.6-1ubuntu3.9 [5,418 B]  
Fetched 495 kB in 0s (779 kB/s)  
Preconfiguring packages ...  
Selecting previously unselected package libxslt1.1:amd64.  
(Reading database ... 63238 files and directories currently installed.)  
Preparing to unpack .../libxslt1.1_1.1.28-2ubuntu0.2_amd64.deb ...  
Unpacking libxslt1.1:amd64 (1.1.28-2ubuntu0.2) ...  
Selecting previously unselected package nginx-common.  
Preparing to unpack .../nginx-common_1.4.6-1ubuntu3.9_all.deb ...  
Unpacking nginx-common (1.4.6-1ubuntu3.9) ...  
Selecting previously unselected package nginx-core.  
Preparing to unpack .../nginx-core_1.4.6-1ubuntu3.9_amd64.deb ...  
Unpacking nginx-core (1.4.6-1ubuntu3.9) ...  
Selecting previously unselected package nginx.  
Preparing to unpack .../nginx_1.4.6-1ubuntu3.9_all.deb ...  
Unpacking nginx (1.4.6-1ubuntu3.9) ...  
Processing triggers for ureadahead (0.100.0-16) ...  
Processing triggers for ufw (0.34~rc-0ubuntu2) ...  
Processing triggers for man-db (2.6.7.1-1ubuntu1) ...  
Setting up libxslt1.1:amd64 (1.1.28-2ubuntu0.2) ...  
Setting up nginx-common (1.4.6-1ubuntu3.9) ...  
Processing triggers for ureadahead (0.100.0-16) ...  
Processing triggers for ufw (0.34~rc-0ubuntu2) ...  
Setting up nginx-core (1.4.6-1ubuntu3.9) ...  
Setting up nginx (1.4.6-1ubuntu3.9) ...  
Processing triggers for libc-bin (2.19-0ubuntu6.15) ...  
vagrant@gap:~$
```

Запуск nginx:

```
Processing triggers for libc-bin (2.19-0ubuntu6.15) ...  
vagrant@gap:~$ sudo service nginx start  
vagrant@gap:~$
```

Представлення роботи в браузері:



3. Замінено контент дефолтної сторінки nginx на власний.

Лістинг у vim:

```
Ctrl_ vagrant@gap: /usr/share/nginx/html
vagrant@gap:/usr/share/nginx/html$ sudo vim index.html
vagrant@gap:/usr/share/nginx/html$ █
```

```
171  :set paste          Set paste          Paste without auto-indent  Enter :set paste to paste without auto-indent
vagrant@gap: /usr/share/nginx/html
<h1>Welcome to nginx!</h1>
<p align="center">Gapiy Denis Eduardovich - not devops</p>
<p align="right">23.12.2023</p>
<p><em>Thank you for using nginx.</em></p>
</body>
</html>
~
16,31 Bot
```

Фінальний результат:



3.5. Відповідно до інструкції, представленою у файлі Vagrantfile, у нас є можливість автоматизувати завантажування потрібних нам пакетів (nginx зокрема) вписавши це ще на етапі конфігурації:

```
# Enable provisioning with a shell script. Additional provisioners such as
# Ansible, Chef, Docker, Puppet and Salt are also available. Please see the
# documentation for more information about their specific syntax and use.
# config.vm.provision "shell", inline: <<-SHELL
#   apt-get update
#   apt-get install -y apache2
# SHELL
```

Відповіді на контрольні питання:

1. Поняття та приклади провайдерів віртуальних машин.

Провайдери VM дозволяють віртуалізації: машин, серверів, мереж та сховищ, необхідних для Розгортання / Керування / Підтримку віртуальних машин.

Найпоширеніший є VirtualBox від Oracle. Вагрантом також рекомендуються застосунки віртуалізації по типу VMWare / Hyper-V.

Серед 'не-локальних' можуть бути наступні: AWS, Google Cloud, IBM Cloud, etc.

2. Vagrant Boxes та Vagrant Cloud.

Vagrant Box – це готовий до використання образ (ОС) віртуальної машини, який можна використовувати для розгортання віртуальних машин. Vagrant Boxes можна створювати за допомогою HashiCorp Packer.

Vagrant Cloud – це репозиторій Vagrant Boxes, доступний для всіх користувачів. Vagrant Cloud містить широкий вибір Vagrant Boxes для різних операційних систем, програмного забезпечення та завдань

3. Vagrantfile – ініціалізація та інструкції для налаштувань.

Це текстовий файл, який використовується для ініціалізації Vagrant-проекту та надання інструкцій для налаштування віртуальних машин. Серед інструкцій можуть вказуватись провайдер, вагрант боксу, прокидання портів, запуску скриптів і так далі.

4. Команди Vagrant. Найпоширеніші:

`vagrant up` – розгортання віртуальних машин.

`vagrant halt` – зупинення

`vagrant destroy` – видалення

`vagrant ssh` – підключення до машини по SSH.

`vagrant provision` – виконання скрипта для налаштування ВМ.

5. Розібратись що відбувається на картинці нижче та познайомитись з поняттям SCM (System Configuration Management): Ansible, Chef, Puppet і про існування Terraform теж дізнатись (без вдавання в деталі)?

1. Розробник конфігурує та створює vagrant проект.

2. Vagrant взаємодія з virtualBox передаючи потрібні налаштування вмі.
3. Середовище віртуалізації створює екземпляр віртуальної машини.
4. За допомоги провайдерів налаштовує програмне забезпечення на створеній вмі.
5. Запуск скриптів чи аналітичних застосунків на запусненій вмі.
6. Самостійне 'в'їзання' у вмі розробником за допомогою SSH.

Сертифікати про проходження курсів

1. Containerized Applications on AWS (coursera)

Посилання на курс:

<https://www.coursera.org/learn/containerized-applications-on-aws>

Сертифікат / підтвердження:

<https://coursera.org/share/10b27afa4845e46d38fa1164169e158d>



Нотатки:

- Benefits of containers:
 - Optimization of resource utilization: You can fit multiple, lightweight containers on a single virtual machine, and increase the efficiency and density of your resources.
 - Automation: The standard packaging and interaction with containers can make it easier to automate software development lifecycle tasks, such as building, deploying, and scaling applications.
 - Speed: Containers are quick to run. You can start, scale, or delete containers in seconds.
 - Scalability: Because containers facilitate microservices architectures, you can meet demand by using containers to scale out horizontally both quickly and efficiently.
- Increased developer productivity:
 - Developers can spend less time on operations and environment debugging, and spend more time on application development.

- Code portability: Having confidence that your workload will behave consistently across environments is one of the major benefits of containers.
- The Docker daemon runs on a host and listens for API calls to manage Docker objects. For example, if you want to build an image, run a container, or create a volume, you can use the Docker command line interface (CLI) to create the request. The request will then be submitted to the Docker daemon to be managed.
- The Docker command line interface (CLI) is the client interface for interacting with Docker objects. The Docker CLI is how you issue commands to Docker to build your containers, run your containers, or stop your containers.
- A Dockerfile is a text document that contains instructions on how to build your container image. A Dockerfile includes instructions for what base layer to use (for example, whether you want to use a minimal version of Linux or use an image that has preinstalled software, such as a database engine). Other instructions include running commands in the container, or copying your application data, dependencies, and configurations into the container.
- A container image is created when you run the build command from the Docker CLI and it follows the instructions in a Dockerfile. A container image is made up of a series of read-only layers, with one writable layer on top where files can be added, modified, or deleted. Each layer is created from an instruction in the Dockerfile.
- An image registry is a repository where you can store container images. You can store the images either publicly or privately. From the repository, images can be pulled and deployed, or used by other developers.

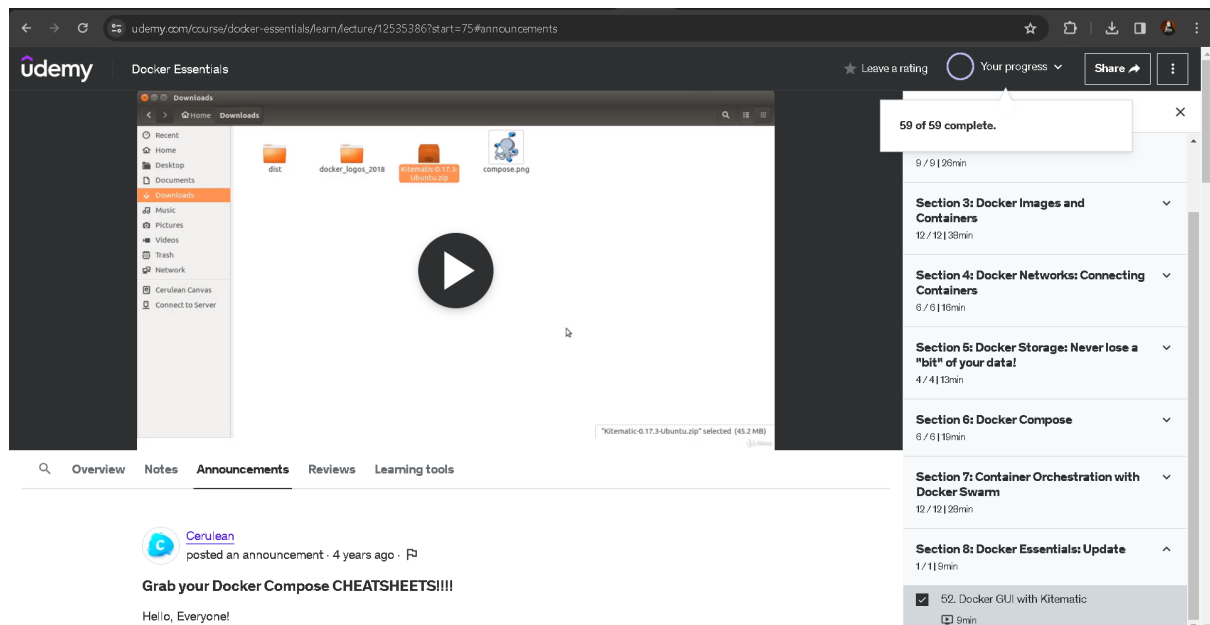
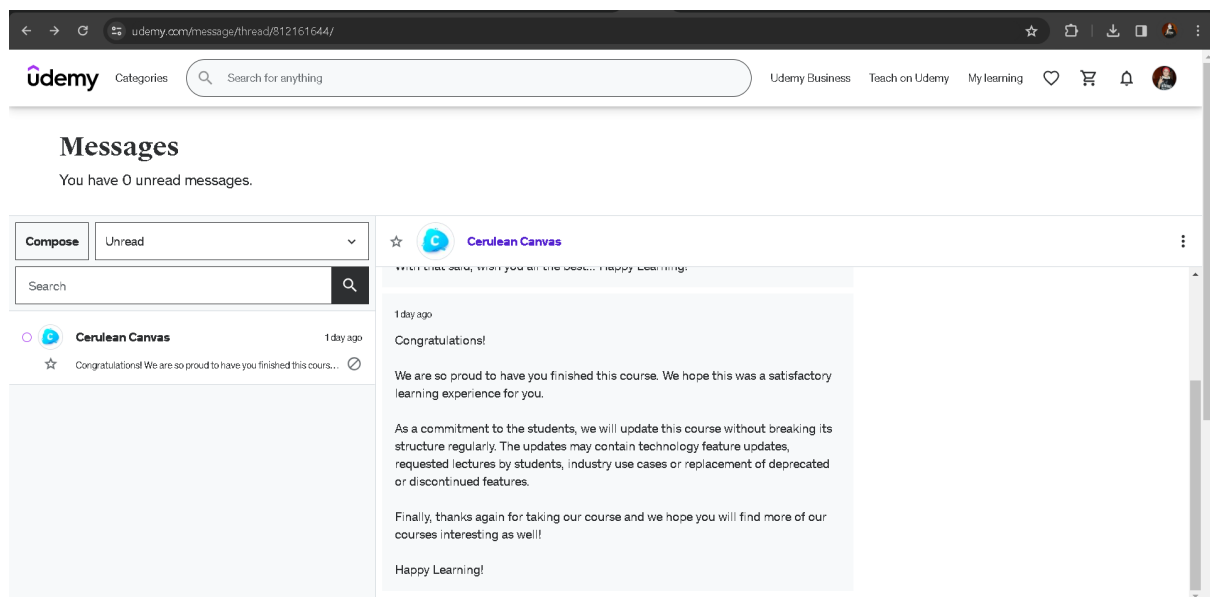
The screenshot shows a web browser displaying a Coursera course page. The browser's address bar shows the URL 'coursera.org/learn/containerized-applications-on-aws/lecture/mzt3/docker-basics'. The Coursera logo is in the top left, and a search bar is in the top right. On the left side, there is a 'Hide menu' button and a list of course items: 'Welcome to the Course', 'Container Basics' (with sub-items: 'Video: Week 1 Introduction', 'Video: Containers Explained Part 1', 'Video: Containers Explained Part 2', 'Reading: Reading 1.1', 'Video: Docker Basics', 'Reading: Reading 1.2'), 'Working with Containers', and 'Week 1 FAQ and Assessment'. The main content area is titled 'Docker Basics' and features a diagram titled 'Docker container image'. The diagram shows a 'Docker image' (represented by a stack of colored rectangles) being pushed to a 'Registry' (represented by a server rack icon), which then serves it to a 'Host' (represented by a laptop icon). A video player is visible on the right side of the diagram, showing a person speaking. In the top right corner of the browser window, a calendar for December 2023 is open, showing the date '20' as 'Wednesday, December 20, 2023' at '7:43:15 PM'. A 'Save note' button is in the bottom right corner of the video player area.

-
- За допомогою приватних реєстрів ви можете отримати хостинговий сервіс, який може керувати зображеннями і забезпечувати безпечний доступ. Хоча у вас є доступ до приватних реєстрів через такі сервіси, як Docker Hub, є сервіс AWS, на якому я хочу зупинитися, який називається Amazon Elastic Container Registry, або Amazon ECR. ECR дуже добре інтегрується в середовище AWS
- AWS App Runner is the fully managed container application service, where it can host my web applications and API services. I simply hand App Runner a container image or source code for a web application, and App Runner prepares the hosting infrastructure automatically.
- AWS Lambda is a serverless compute service. You hand over your code to Lambda, and your code is run on AWS infrastructure. You don't worry about the servers or the infrastructure running the code. You can also package up your code into a container image to hand on to Lambda
- AWS CodeBuild is a fully managed continuous integration service. CodeBuild compiles source code, runs tests, and produces software packages for development teams. CodeBuild uses a container image to run the build commands. CodeBuild supplies container images with popular languages and build tools installed, or you can supply your own image.

- приклад завдань які додатково були там але вони були не обов'язковими:
<https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/DEV-AWS-MO-ContainersRedux/exercise-2-apprunner.html>

2. Docker Essentials by Cerulean (udemy)

Посилання на курс: <https://www.udemy.com/course/docker-essentials/>
Сертифікат / підтвердження:



Нотатки:

- Матеріал дуже подібний до лекційного, просто лише викладається інструктором з 'азійським' акцентом. Тому краще

б за витрачений час глянув, ще декілька лекцій від 'рідного' лектора.

- What are the contents of a Dockerfile?
 - FROM: Defines the base image. All the instructions that follow are run in a container launched from the base image.
 - WORKDIR: Sets the working directory for the subsequent instructions.
 - ENV: Sets environment variables.
 - COPY: Copies files and directories into the container image.
 - RUN: Runs commands in the new container. This instruction commits a new layer on top of the present layer.
 - CMD: Sets the default command to run when the container is launched.
 - EXPOSE: Is used to document the containers that the port exposes.
- Docker commands
 - docker build: Builds an image from a Dockerfile. In the demonstration, we pass -t to tag the image that's created.
 - docker run: Creates and starts a container. In the demonstration, we use -p to expose ports, -e to set environment variables, and -v to bind mount volumes.
 - docker exec: Runs a command in a running container.
 - docker stop: Stops a container.
 - docker rm: Removes a container. Use -f to force the remove
- The main difference between Docker and Vagrant is how these tools approach deploying a virtual environment. While Vagrant allows you to create an entire virtual machine, Docker packages your applications into miniature containers deployed in a predictable scenario

- приклад розбиття архітектури застосунку зі звичайного бд+бізнес+представлення для користувача:

Microservices application

