

Nhóm 06

Tìm hiểu và triển khai ứng dụng trên Docker

NT132.O11.ANTT

GVHD: Đỗ Hoàng Hiến



THÀNH VIÊN

20520823

Mai Ngọc Phương Trinh

21521195

Trần Lê Minh Ngọc

21522240

Huỳnh Minh Khuê

20520653

Lê Đào Khánh Ngọc

20521518

Nguyễn Thị Hồng Lam

Nội dung

01

Docker

02

Docker Compose

03

Docker và

Docker-compose

04

**Ứng dụng trên
Docker**

01

Docker





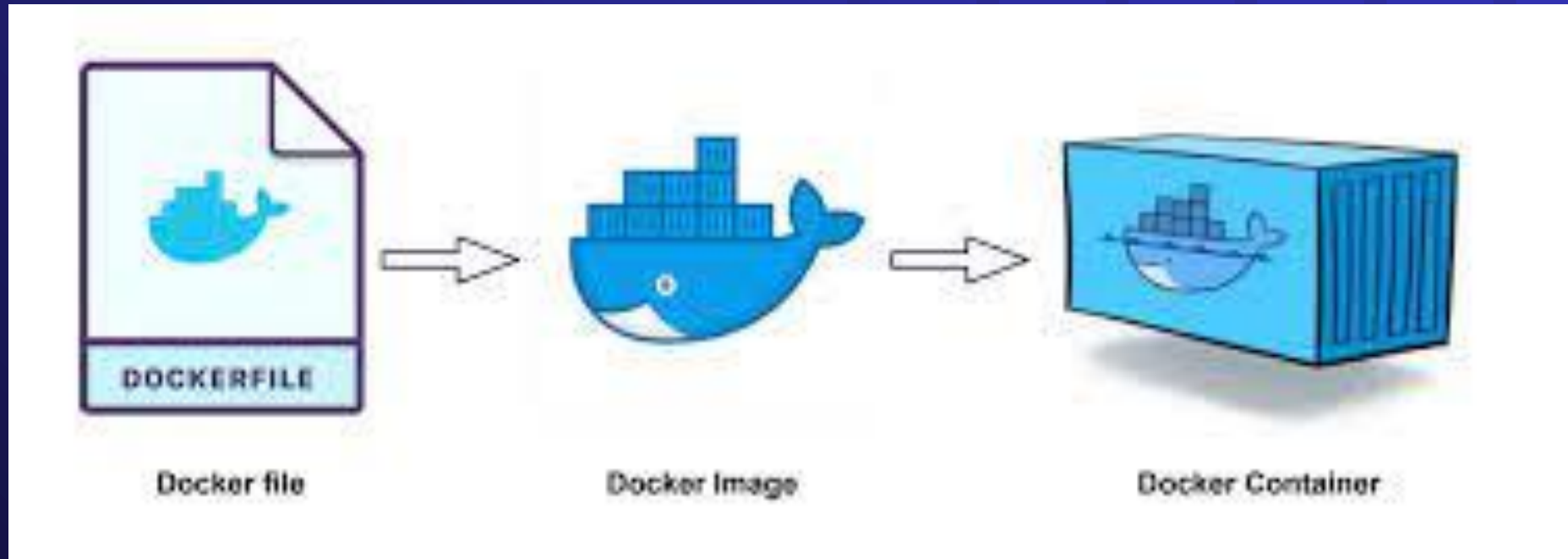
01

Docker



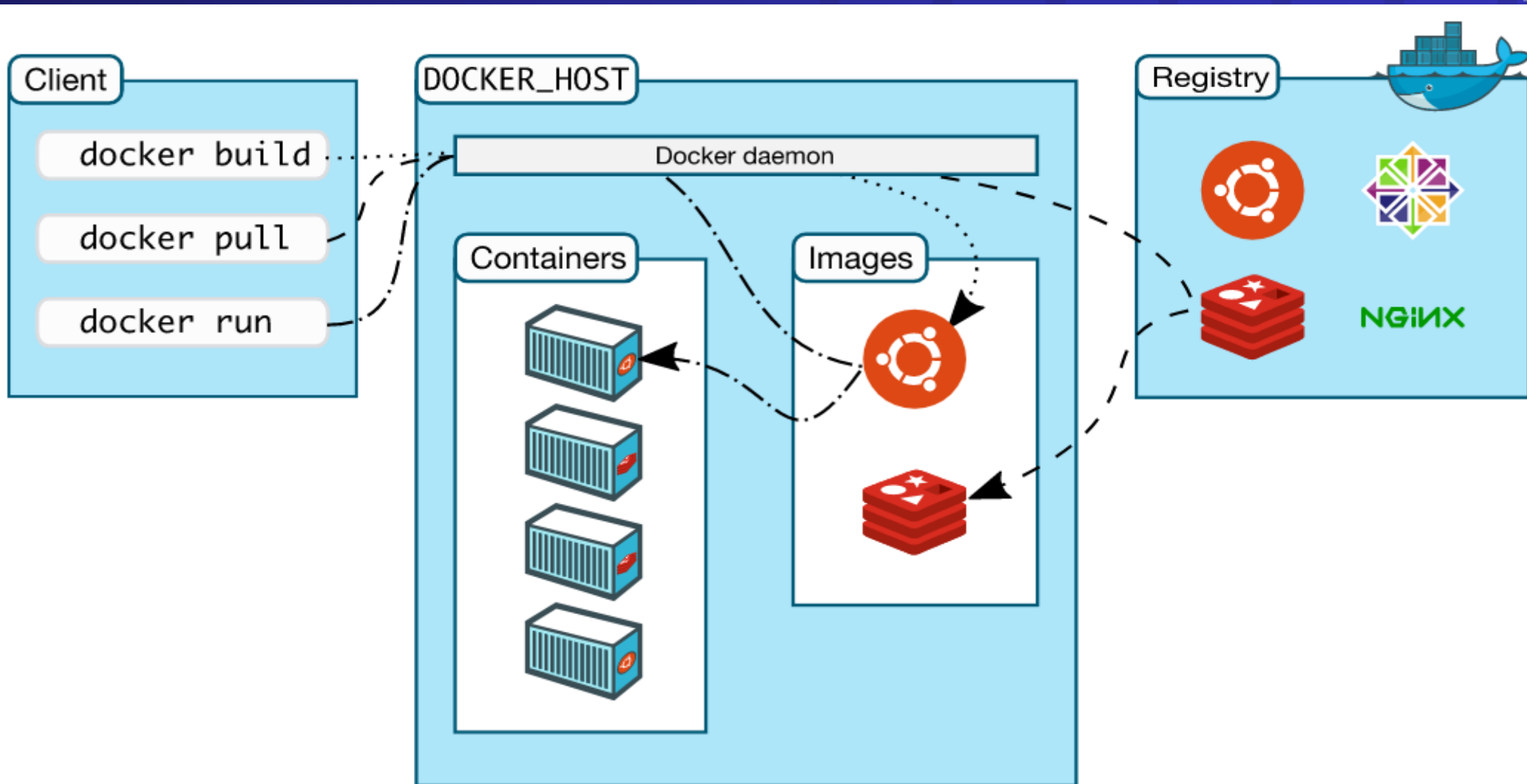
Docker là nền tảng phần mềm mã nguồn mở cho phép triển khai và quản lý ứng dụng dựa trên công nghệ container. Với Docker, ta có thể thiết lập sử dụng các nền tảng để có thể xây dựng và mở rộng quy mô ứng dụng thông qua các container nhanh chóng và thuận tiện hơn.

01 Docker



01 Docker

Kiến trúc của Docker



01 Docker

Lợi ích của Docker

1

Tiện lợi: nhanh chóng tạo môi trường

2

Dễ dàng sử dụng: build, test nhanh chóng

3

Tốc độ: nhẹ và nhanh

4

Khả năng di động: có thể di chuyển giữa các máy

5

Chia sẻ: DockerHub có nhiều public images được tạo bởi cộng đồng

6

Môi trường chạy và khả năng mở rộng: dễ dàng liên kết các container

01 Docker

Các thành phần trong Docker

1

Docker Engine

Một công cụ để đóng gói ứng dụng

2

Docker Hub

Một “github for docker images”

3

Docker Images

Một khuôn mẫu để tạo một container

4

Dockerfile

Một tập tin bao gồm các chỉ dẫn để build một image

5

Docker Container

Một instance của một image

6

Docker Client

Một công cụ giúp người dùng giao tiếp với Docker host

7

Docker Engine

Lắng nghe các yêu cầu từ Docker Client

8

Docker Volumes

Phần dữ liệu được tạo ra khi container được khởi tạo

01

Docker

Một số lệnh cơ bản

- **List image/container:**

```
$ docker image/container ls
```

- **Delete image/container:**

```
$ docker image/container rm  
<tên image/container >
```

- **Delete all image hiện có:**

```
$ docker image rm $(docker  
images -a -q)
```

- **List all container hiện có:**

```
$ docker ps -a
```

- **Stop a container cụ thể:**

```
$ docker stop <tên  
container>
```

- **Run container từ image và thay đổi tên container:**

```
$ docker run -name <tên  
container> <tên image
```

01

Docker

Một số lệnh cơ bản

- **Tạo một container chạy ngầm:**

```
$ docker run -d <tên image>
```

- **Tải một image trên docker hub:**

```
$ docker pull <tên image>
```

- **Start một container:**

```
$ docker start <tên container>
```

- **Stop all container:**

```
$ docker stop $(docker ps -a -q)
```

- **Delete all container hiện có:**

```
$ docker rm $(docker ps -a -q)
```

- **Show log a container:**

```
$ docker logs <tên container>
```

- **Build một image từ container:**

```
$ docker build -t <tên container>
```

01 Docker

Container windows có thể chạy trên linux không? Ngược lại?

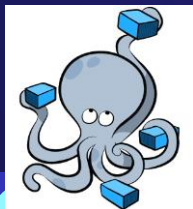
- Không thể chạy container Windows trực tiếp trên Linux và ngược lại. Điều này là do các container sử dụng các tài nguyên và trình điều khiển cơ bản của hệ điều hành, Windows sử dụng kernel Windows và container Linux sử dụng kernel Linux.
- Container Linux có thể chạy trên Windows thông qua một tính năng được gọi là "Windows Subsystem for Linux" (WSL). WSL cung cấp một môi trường Linux tương tác chạy trên Windows mà không cần máy ảo hoặc khởi động lại hệ thống.

02

Docker Compose



02 Docker Compose



Docker compose là công cụ dùng để định nghĩa và run multi-container cho Docker application. Với compose chúng ta sử dụng file YAML để config các services cho application của bạn. Sau đó dùng command để create và run từ những config đó.

+ 02 Docker Compose

Lợi ích của docker compose

1

Quản lý đa vùng chứa: cấu hình và quản lý nhiều container

2

Quản lý phụ thuộc giữa các vùng chứa: xác định và quản lý mối quan hệ phụ thuộc giữa các vùng chứa

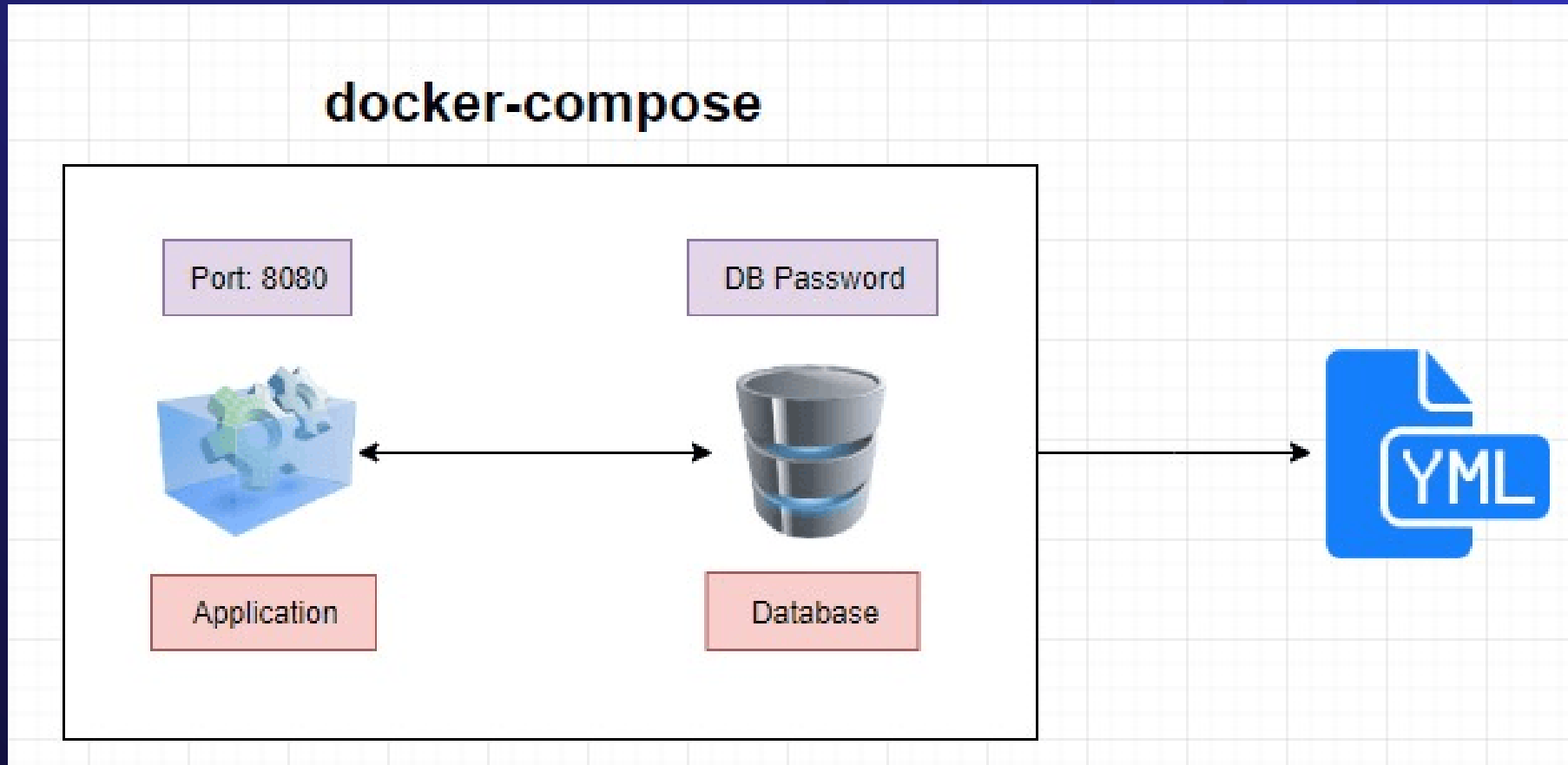
3

Khả năng mở rộng: mở rộng ứng dụng dễ dàng

4

Phù hợp hơn với các trình soạn thảo mã nguồn

+ 02 Docker Compose



+ 02 Docker Compose

Một số lệnh cơ bản

- `docker-compose up`: Khởi động các container
- `docker-compose down`: Dừng và xóa các container
- `docker-compose ps`: Hiển thị trạng thái của các container
- `docker-compose build`: Tạo image từ Dockerfile trong mỗi dịch vụ
- `docker-compose restart`: Khởi động lại các container
- `docker-compose stop`: Dừng các container

+ 02 Docker Compose

Một số lệnh cơ bản

- **docker-compose rm:** Xóa các container không sử dụng
- **docker-compose logs:** Hiển thị các logs của các container
- **docker-compose config:** Hiển thị các cấu hình của Docker Compose
- **docker-compose exec:** Thực thi một lệnh trên một container
- **docker-compose port:** Hiển thị các port của các container
- **docker-compose top:** Hiển thị các process đang chạy trong các container

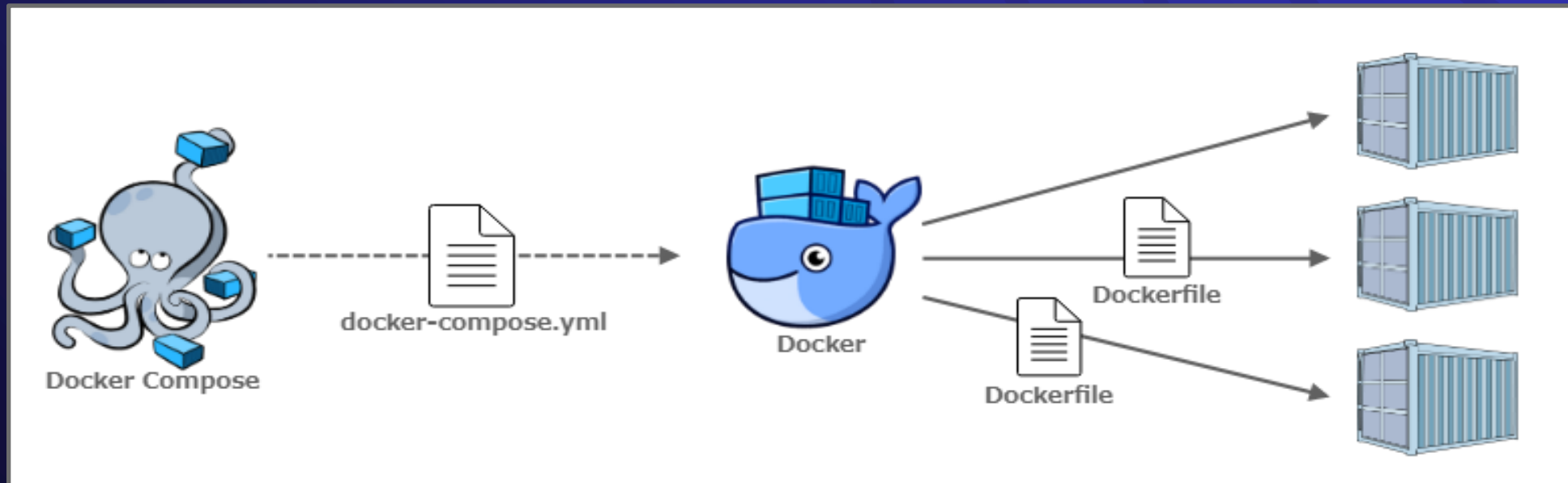
03

Docker và Docker-compose



+ 03 Docker và Docker-compose

- Docker run hoàn toàn dựa trên dòng lệnh, trong khi docker-compose đọc dữ liệu cấu hình từ tệp YAML.
- Docker run chỉ có thể khởi động một container mỗi lần, docker-compose sẽ định cấu hình và chạy nhiều container.





04

Ứng dụng trên Docker



+ 04 Ứng dụng

➤ Lợi ích của việc triển khai ứng dụng trên docker

1

Dễ sử dụng

Nền tảng này dựa trên mã nguồn mở, có một cộng đồng lớn hỗ trợ

2

Không tiêu tốn nhiều tài nguyên

Cho phép các container chia sẻ những chức năng kernel

3

Mở rộng quy mô tốt hơn

Docker rất nhẹ, các container có thể được thay đổi kích thước

4

Triển khai nhất quán

Mỗi container độc lập và chạy trên Linux kernel.

5

Cung cấp tính năng kiểm soát phiên bản

Các thay đổi code được tự động lưu dưới dạng những layer trong file image.

6

Tương thích với Microservice

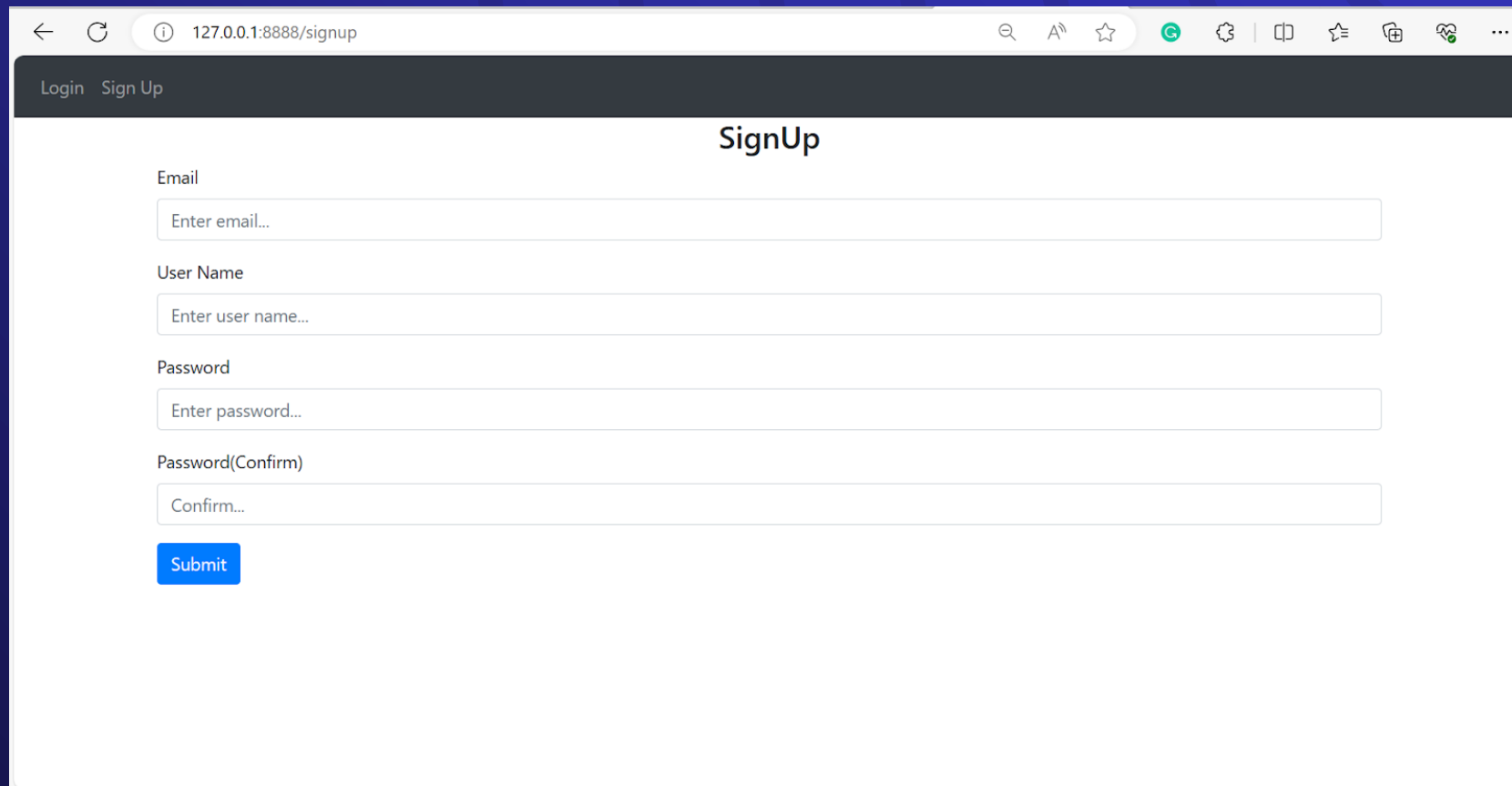
Sự độc lập của các service có thể giúp làm cho hệ thống chống lỗi tốt hơn



04 Ứng dụng

a. Ứng dụng ghi chú công việc Todolist

Mục đích: Cho phép người dùng thêm, xóa, lưu trữ, xem các mẫu văn bản trên bảng ghi nhớ.



The screenshot displays a web browser window with the address bar showing '127.0.0.1:8888/signup'. The page has a dark header with 'Login' and 'Sign Up' links. The main content area is titled 'SignUp' and contains a form with the following fields:

- Email: Enter email...
- User Name: Enter user name...
- Password: Enter password...
- Password(Confirm): Confirm...

A blue 'Submit' button is located below the form fields.

04 Ứng dụng

```
dockerfile > ...
1 FROM python:3.12.0
2 WORKDIR /app
3 COPY . /app/
4 RUN pip install -r requirements.txt
5 ENTRYPOINT [ "python" ]
6 CMD [ "app.py" ]
```

- dockerfile

```
requirements.txt
1 flask
2 Flask-SQLAlchemy
3 flask-login
4 python-dotenv
```

- requirements

```
docker-compose.yml
1 version: '3.8'
2 services:
3   todoapp:
4     volumes:
5       - D:/Todoapp_Tutorial/instance/todoapp.db:/instance/todoapp.db
6     build:
7       context: ./
8     ports:
9       - 8888:5000
```

- docker-compose

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

PS D:\Todolist_Tutorial> docker ps

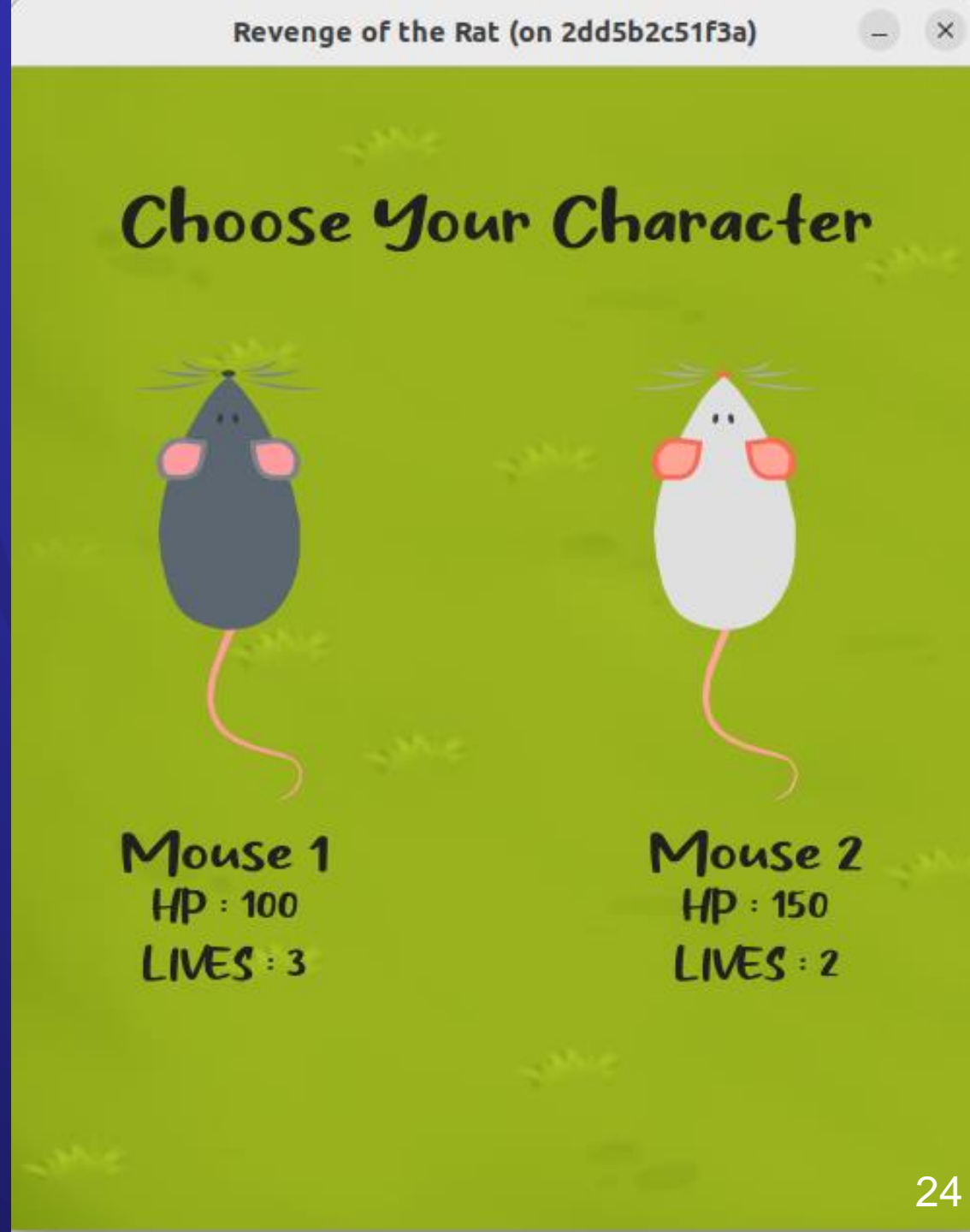
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
6006005f04fd	todolist_tutorial-todolist	"python app.py"	9 hours ago	Up 16 seconds	0.0.0.0:8888->5000/tcp	todolist_tutorial-todolist-1

PS D:\Todolist_Tutorial> docker-compose build

04 Ứng dụng

b. Trò chơi “Revenge of the Rat”

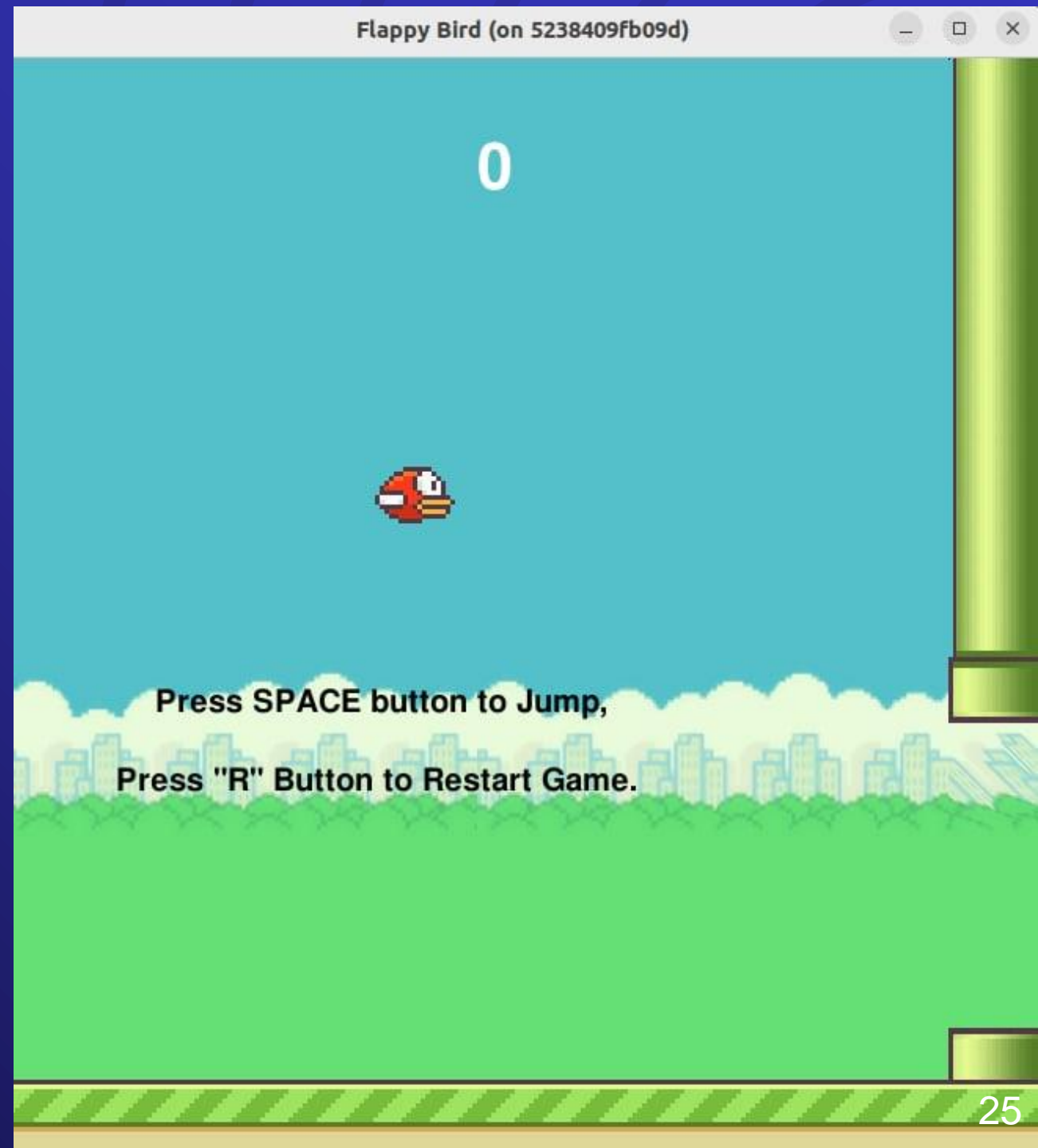
- Nhân vật “chuột” sẽ có nhiệm vụ tiêu diệt các nhân vật “mèo” bằng cách bắn vào chúng.
- Người chơi sẽ có ba lượt chơi mỗi khi va vào “mèo” số lần nhất định, “chuột” sẽ mất một mạng và sau ba lần như vậy, trò chơi sẽ kết thúc.



04 Ứng dụng

c. Trò chơi “Flappy Bird”

- Nhân vật bird sẽ có nhiệm vụ bay qua các chỗ trống của các cây cột.
- Người chơi sẽ thua nếu điều khiển bird va chạm vào cột hoặc rơi xuống đất.



04 Ứng dụng

- dockerfile của 2 ứng dụng game

```
Dockerfile
~/Downloads

1 FROM ubuntu:20.04
2 ENV DEBIAN_FRONTEND=noninteractive
3
4 # install semua dependensi paket (library) yang dibutuhkan
5 RUN apt-get update && apt-get upgrade -y && apt-get install -y \
6     tzdata \
7     libssl-dev \
8     openssl \
9     zlib1g-dev \
10    build-essential \
11    checkinstall \
12    libffi-dev \
13    libsqlite3-dev \
14    vim \
15    curl \
16    make \
17    sudo \
18    python3-pip \
19    python3-pygame \
20    libsdl1.2-dev \
21    libsdl-image1.2-dev \
22    libsdl-mixer1.2-dev \
23    libsdl-sound1.2-dev \
24    libsdl-ttf2.0-dev \
25    libsdl2-dev \
26    libsdl2-image-dev \
27    libsdl2-mixer-dev \
28    libsdl2-ttf-dev \
29    libsdl2-gfx-dev \
30    libsdl2-net-dev
```

```
31
32 # install x11
33 RUN apt install -qqy x11-apps
34
35 # install pygame
36 RUN pip3 install pygame
37
38 # menambahkan parameter build argument
39 ARG USER=docker
40 ARG UID=1000
41 ARG GID=1000
42
43 # menambahkan default password untuk user
44 ARG PW=docker
45 RUN useradd -m ${USER} --uid=${UID} --shell /bin/bash && echo "${USER}:${PW}" | chpasswd \
46     && adduser docker sudo
47
48 # Setup default user, ketika memasuki kontainer
49 USER ${UID}:${GID}
50 WORKDIR /home/${USER}
```


04 Ứng dụng

- Docker-compose của 2 ứng dụng game

```
Open  docker-compose.yml  Save  ~/Downloads
1 version: '3'
2 services:
3   game:
4     image: game:latest
5     container_name: Application_using_pygame
6     build:
7       context: .
8       dockerfile: Dockerfile-game
9     privileged: true
10    stdin_open: true
11    tty: true
12    cap_add:
13      - SYS_PTRACE
14    user: "1000:1000"
15    volumes:
16      - /tmp/.X11-unix:/tmp/.X11-unix
17      - /run/dbus:/run/dbus
18      - /dev/shm:/dev/shm
19      - /dev/snd:/dev/snd
20      - /dev/dri:/dev/dri
21      - ${XDG_RUNTIME_DIR}/pulse/native:${XDG_RUNTIME_DIR}/pulse/native
22      - /run/user/1000/pulse:/run/user/1000/pulse
23      - /var/run/dbus:/var/run/dbus
24      - /home/minhngoc/Downloads:/home/docker
25    environment:
26      - DISPLAY
27      - PULSE_SERVER=unix:${XDG_RUNTIME_DIR}/pulse/native
28      - XDG_RUNTIME_DIR=/run/user/1000
```



Library

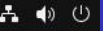
Type he...

- My Computer
 - Debian 1
 - Windows
 - Debian 1
 - Ubuntu 6
 - Metasplo
 - Ubuntu 6
 - FreeBSD

Ubuntu 64-bit

Activities Files

Thg 12 25 22:23



File manager window showing the contents of the 'ratgame' directory.

Name	Size	Location
ratgame	9 items	
Dockerfile-game	1,1 kB	
game.py	825 bytes	flappybird
img	34 items	ratgame
settings.py	491 bytes	flappybird
head.png	89,6 kB	ratgame
gun.png	4,0 kB	ratgame/img
game.cpython-37.pyc	1,3 kB	flappybird/__pycache__
game.cpython-38.pyc	1,3 kB	flappybird/__pycache__
game.cpython-310.pyc	1,3 kB	flappybird/__pycache__

"ratgame" selected (containing 9 items)

04 Ứng dụng

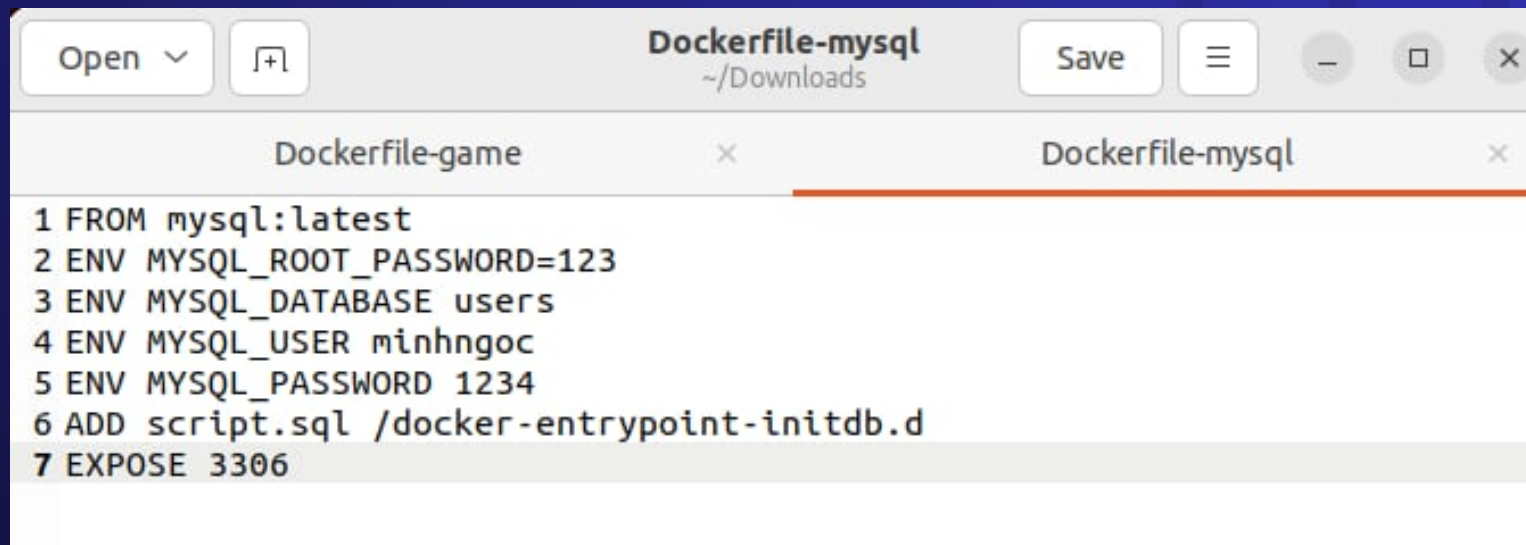
d. Triển khai MySQL

- Docker hóa CSDL MySQL
- Triển khai MySQL trên Docker mang lại nhiều lợi ích về quản lý, di động và tích hợp trong các môi trường phức tạp và đa dạng.

```
mysql> select * from users;
```

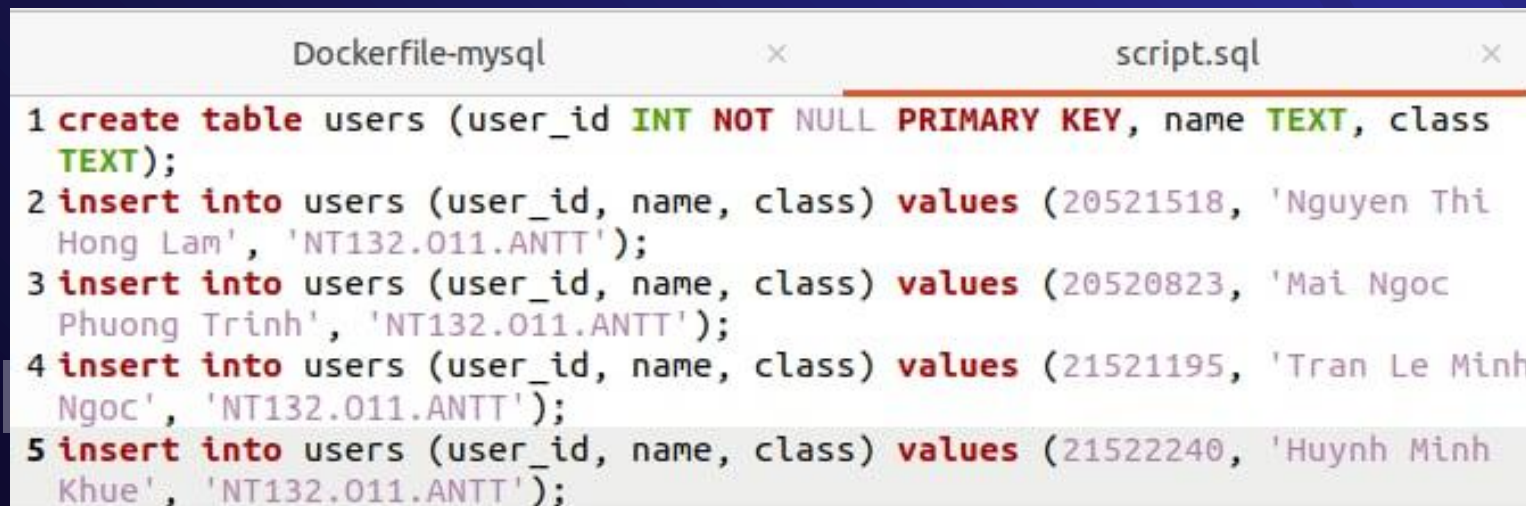
user_id	name	class
20520823	Mai Ngoc Phuong Trinh	NT132.011.ANTT
20521518	Nguyen Thi Hong Lam	NT132.011.ANTT
21521195	Tran Le Minh Ngoc	NT132.011.ANTT
21522240	Huynh Minh Khue	NT132.011.ANTT

04 Ứng dụng



The screenshot shows a code editor window titled "Dockerfile-mysql" with a file path of "~/Downloads". It contains two tabs: "Dockerfile-game" and "Dockerfile-mysql". The "Dockerfile-mysql" tab is active and displays the following Dockerfile content:

```
1 FROM mysql:latest
2 ENV MYSQL_ROOT_PASSWORD=123
3 ENV MYSQL_DATABASE users
4 ENV MYSQL_USER minhngoc
5 ENV MYSQL_PASSWORD 1234
6 ADD script.sql /docker-entrypoint-initdb.d
7 EXPOSE 3306
```



The screenshot shows a code editor window with two tabs: "Dockerfile-mysql" and "script.sql". The "script.sql" tab is active and displays the following SQL script content:

```
1 create table users (user_id INT NOT NULL PRIMARY KEY, name TEXT, class TEXT);
2 insert into users (user_id, name, class) values (20521518, 'Nguyen Thi Hong Lam', 'NT132.011.ANTT');
3 insert into users (user_id, name, class) values (20520823, 'Mai Ngoc Phuong Trinh', 'NT132.011.ANTT');
4 insert into users (user_id, name, class) values (21521195, 'Tran Le Minh Ngoc', 'NT132.011.ANTT');
5 insert into users (user_id, name, class) values (21522240, 'Huynh Minh Khue', 'NT132.011.ANTT');
```

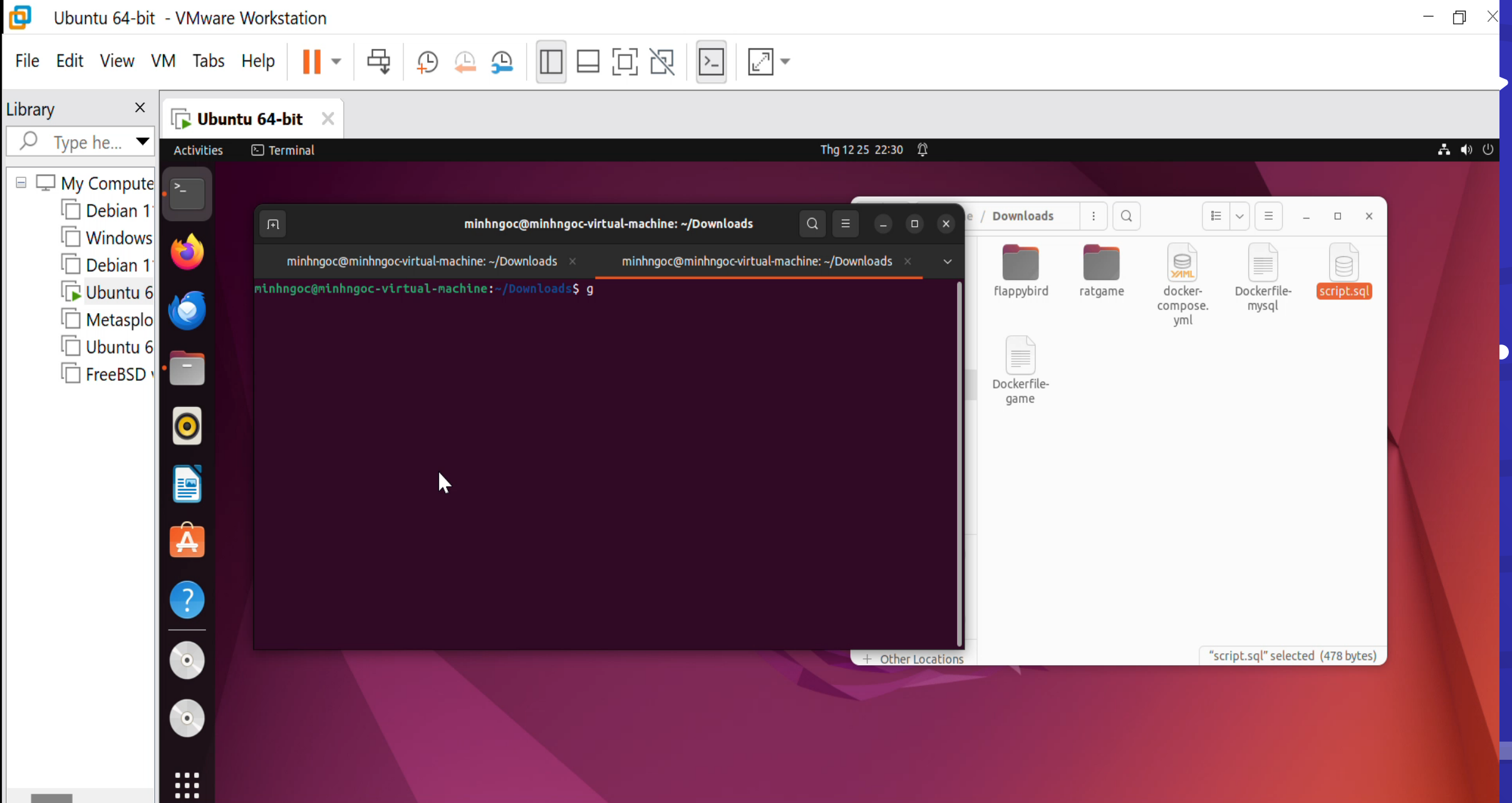
- dockerfile của mysql

- script.sql

04 Ứng dụng

```
build-sql:
  image: build-sql:latest
  container_name: Application_using_mysql
  build:
    context: .
    dockerfile: Dockerfile-mysql
  ports:
    - "3306:3306"
```

- docker-compose của mysql

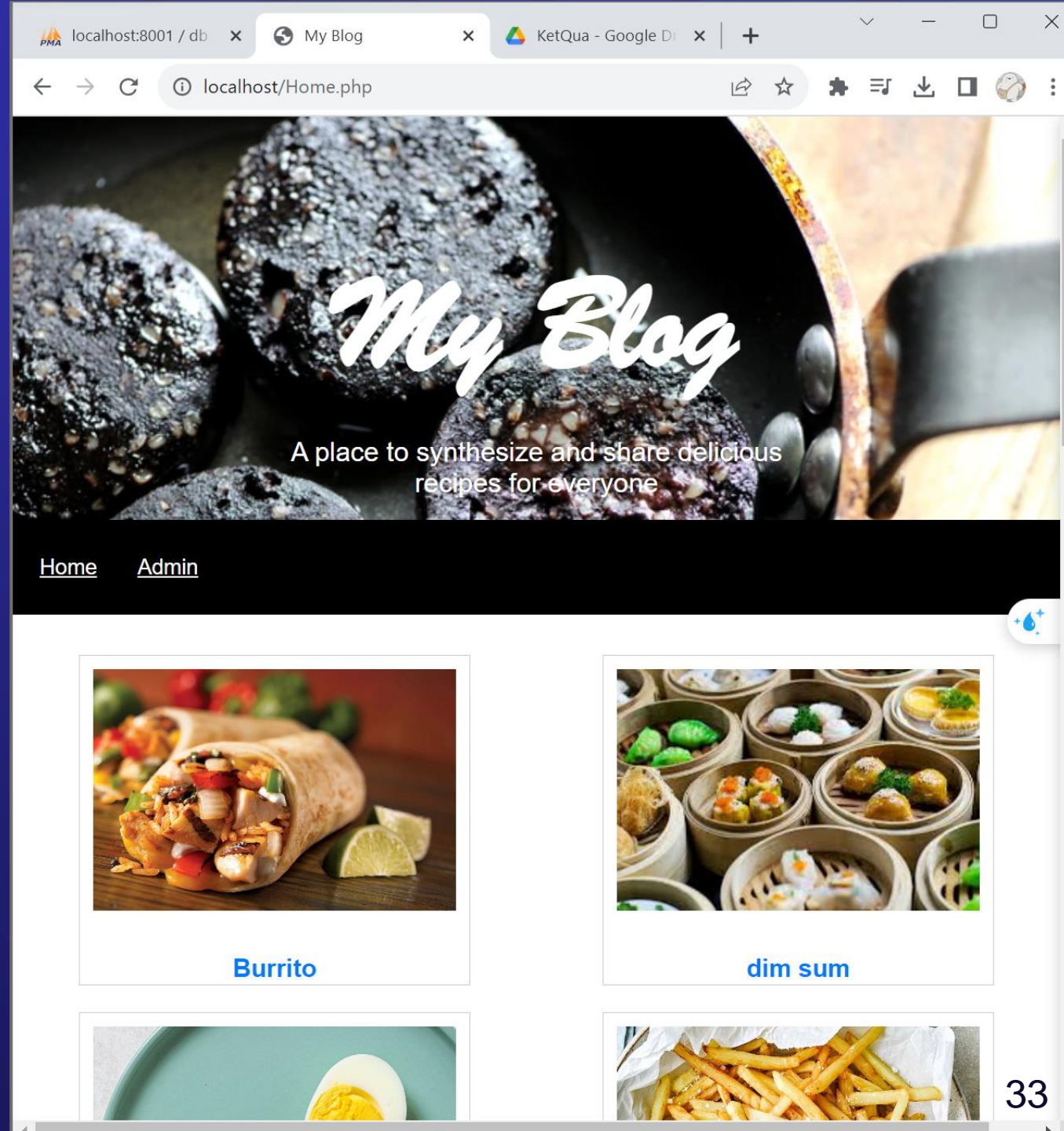


04 Ứng dụng

e. Ứng dụng tạo blog

Mục đích của blog cá nhân:

- Chia sẻ nội dung
- Kết nối với cộng đồng
- Giới thiệu và quảng bá sản phẩm

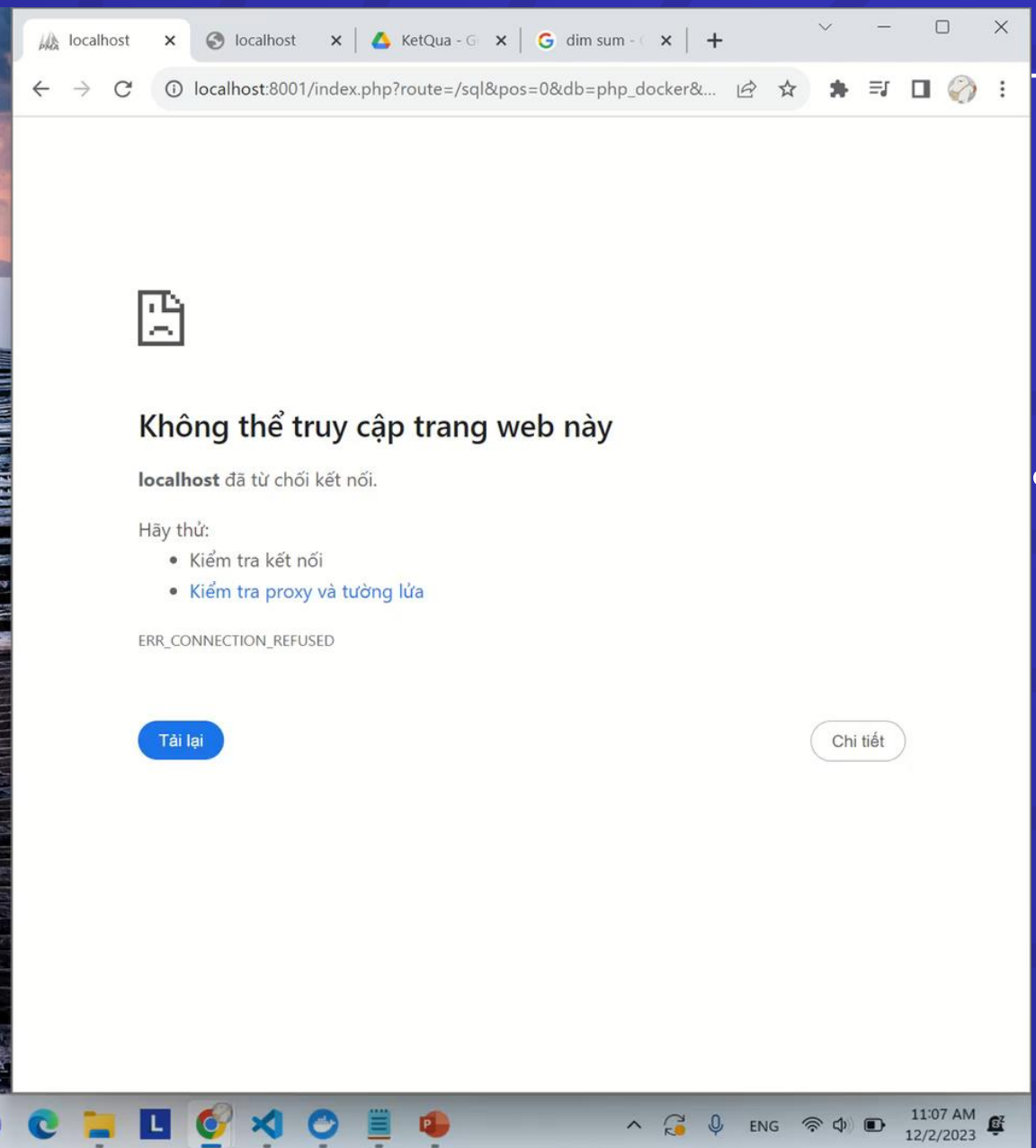


04 Ứng dụng

- docker-compose của web

D: > HKV > QuanTriMang > DoAn > docker-compose.yml

```
1 version: '3.8'
2 services:
3   db:
4     image: mysql:latest
5     environment:
6       - MYSQL_DATABASE=php_docker
7       - MYSQL_USER=php_docker
8       - MYSQL_PASSWORD=password
9       - MYSQL_ALLOW_EMPTY_PASSWORD=1
10    volumes:
11      - "./db:/docker-entrypoint-initdb.d"
12  www:
13    image: php:apache
14    volumes:
15      - ".: /var/www/html"
16    ports:
17      - 80:80
18      - 443:443
19  phpmyadmin:
20    image: phpmyadmin/phpmyadmin
21    ports:
22      - 8001:80
23    environment:
24      - PMA_HOST=db
25      - PMA_PORT=3306
```

A futuristic blue neon tunnel with an astronaut at the end. The tunnel is composed of many concentric, glowing blue lines that create a sense of depth and perspective. An astronaut in a white suit is standing at the far end of the tunnel, facing away from the viewer. The background is a dark blue gradient with some abstract geometric shapes and symbols scattered around.

Thank you!