**20090551 – Đỗ Gia Huy**

**THỰC HÀNH DOCKER COMPOSE**

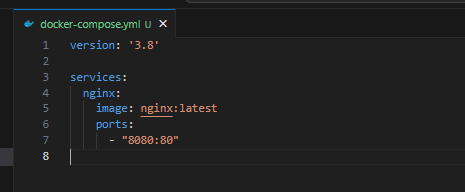
**Bài 1: Chạy một container đơn giản với Docker Compose**

Yêu cầu:

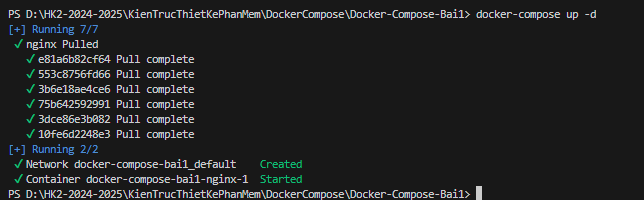
Tạo một container chạy Nginx bằng Docker Compose.

Map cổng 8080 của máy host với cổng 80 của container.

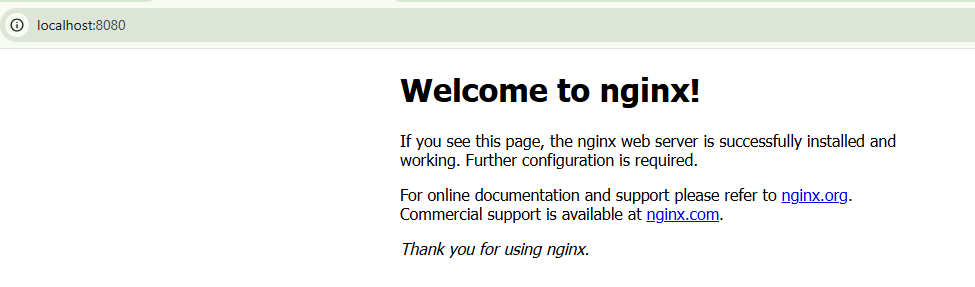
* **Tạo file cấu hình docker-compose.yml**

****

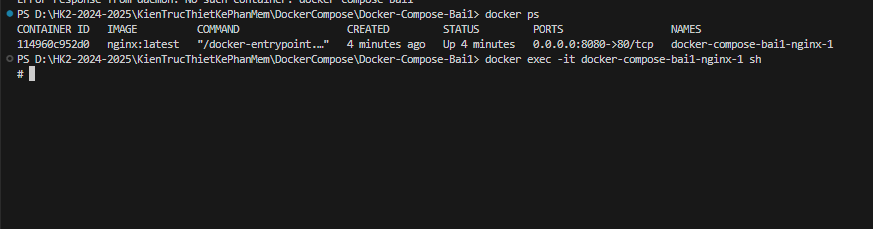
* **Chạy docker-compose, dùng lệnh *docker-compose up –d***

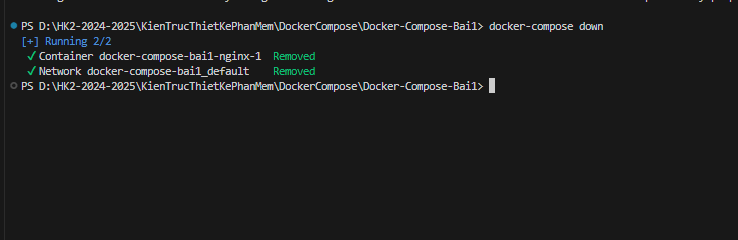
****

* **Mở trình duyệt và truy cập:** [***http://localhost:8080***](http://localhost:8080)

****

* **Một số thao tác với container nginx đã tạo (docker ps, docker exec -it docker-compose-bai1-nginx-1 sh)**

****

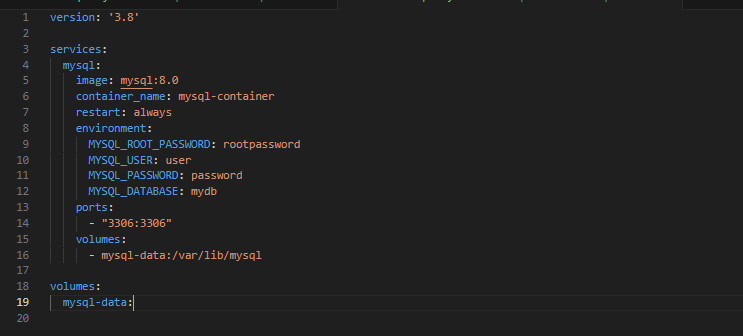


**Bài 2: Chạy MySQL với Docker Compose**

Yêu cầu: Tạo một container chạy MySQL phiên bản 8.0.

Đặt username là user, password là password và database là mydb

* **Tạo file cấu hình docker-compose.yml**



MYSQL\_ROOT\_PASSWORD Mật khẩu cho tài khoản root

MYSQL\_USER Tạo user mới tên user

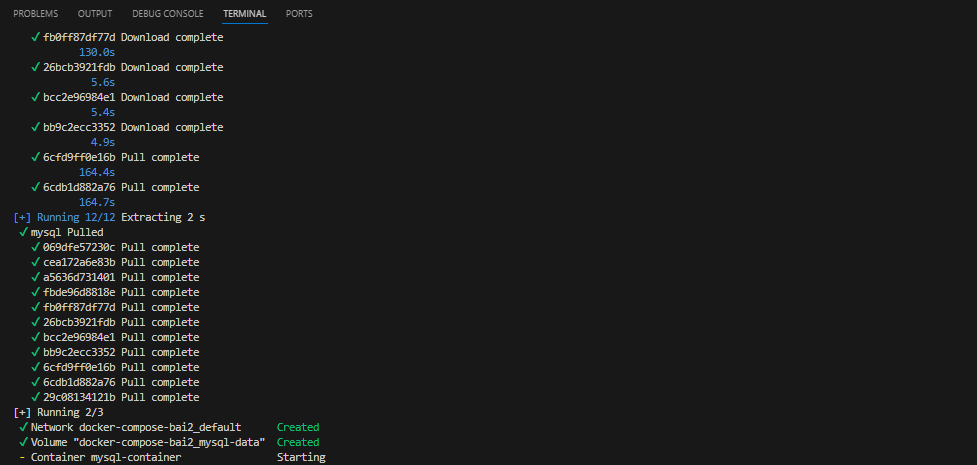
MYSQL\_PASSWORD Mật khẩu cho user

MYSQL\_DATABASE Tạo sẵn database tên mydb

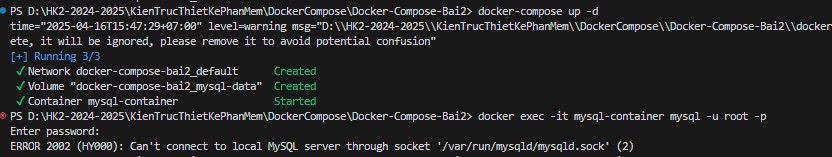
3306:3306 Map cổng 3306 của host với container

volumes Lưu dữ liệu MySQL tránh mất khi container bị xóa

* **Chạy docker-compose, dùng lệnh *docker-compose up –d***

****

* **Chạy docker exec -it mysql-container mysql -u user –p nhưng không kết nối được**

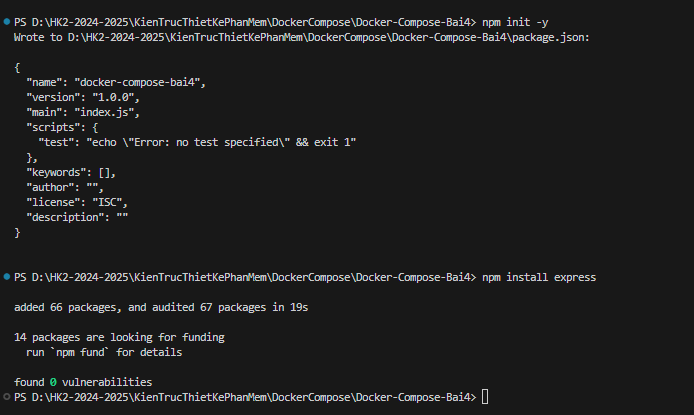
****

**Bài 4: Chạy ứng dụng Node.js với Docker Compose**

Yêu cầu:

Chạy một ứng dụng Node.js đơn giản với Express.

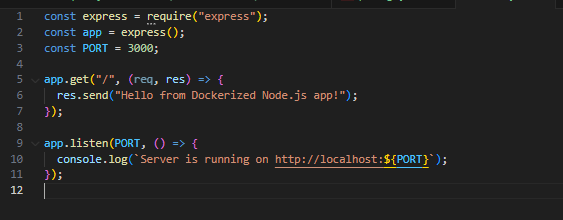
* **Khởi tạo dự án và install Express**



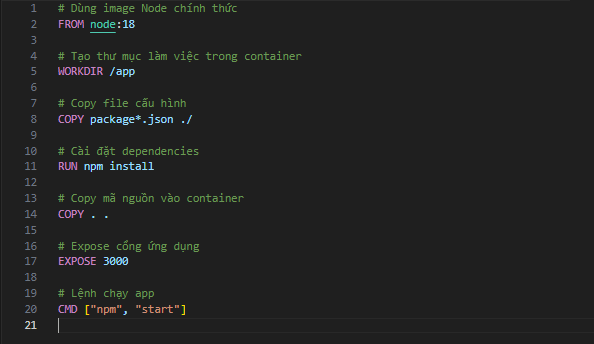
**Package-json**



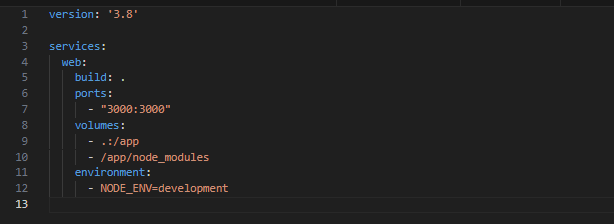
**Tạo index.js**

****

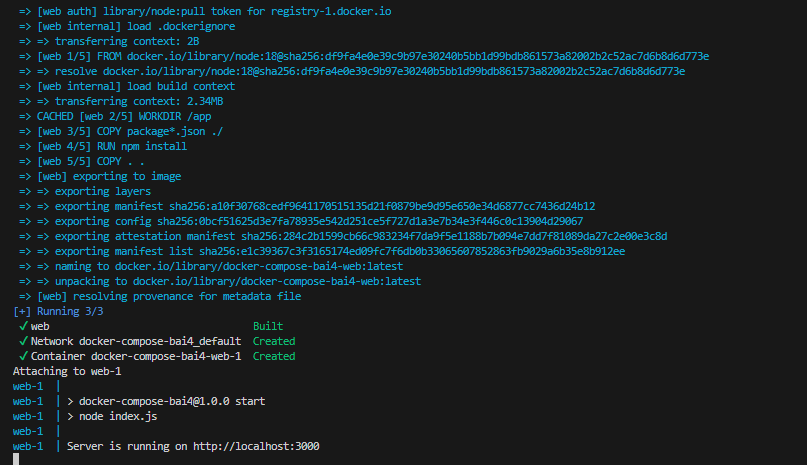
**Tạo Dockerfile**

****

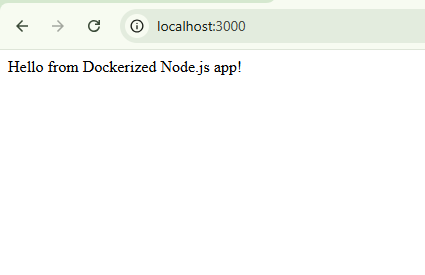
**Tạo Docker-compose.yml**

****

**Chạy ứng dụng *docker-compose up –build***

****

**Chạy thành công**

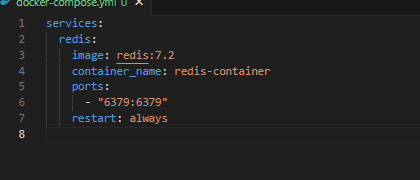
****

**Bài 5: Chạy Redis với Docker Compose**

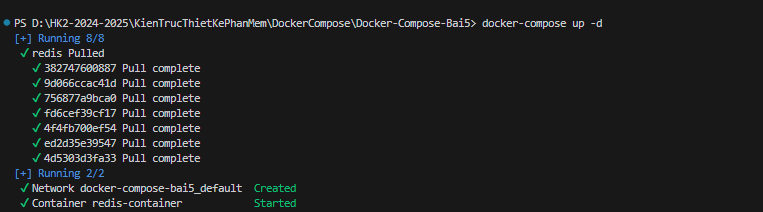
Yêu cầu:

Chạy một container Redis trên cổng 6379.

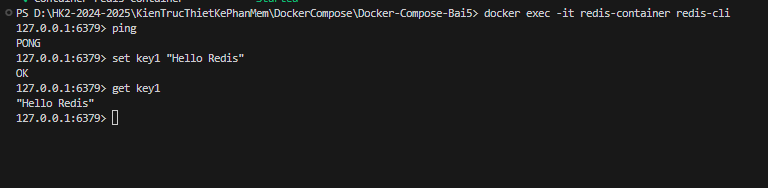
* **Tạo docker-compose.yml**



* **Chạy docker-compose up -d**



* **Kiểm tra Redis đang hoạt động**

****

Bài 6: Chạy WordPress với MySQL

Yêu cầu:

Chạy WordPress với MySQL bằng Docker Compose.

Bài 7: Chạy MongoDB với Docker Compose

Yêu cầu:

Chạy MongoDB và Mongo Express để quản lý.

Bài 8: Kết nối nhiều dịch vụ với Docker Compose

Yêu cầu:

Chạy Node.js kết nối với MySQL.

Bài 9: Chạy ứng dụng Python Flask với Docker Compose

Yêu cầu:

Chạy ứng dụng Flask đơn giản với Docker Compose.

Bài 10: Lưu trữ dữ liệu với Docker Volumes

Yêu cầu:

Chạy MySQL và gắn volume để dữ liệu không bị mất.

Bài 11: Chạy dịch vụ Postgres với Adminer

Yêu cầu:

Chạy PostgreSQL và Adminer (công cụ quản lý database) bằng Docker Compose.

PostgreSQL phải có database tên mydb, user là user, password là password.

Adminer chạy trên cổng 8083

Bài 12: Giám sát container với Prometheus và Grafana

Yêu cầu:

Chạy Prometheus, Grafana và Node Exporter bằng Docker Compose để giám sát hệ thống.

Bài 13: Chạy ứng dụng React với Nginx

Yêu cầu:

Chạy một ứng dụng React và serve nó bằng Nginx.

Bài 14: Cấu hình mạng riêng giữa các container

Yêu cầu:

Chạy 2 container có thể giao tiếp với nhau trong một mạng riêng.

Bài 15: Giới hạn tài nguyên cho container

Yêu cầu:

Giới hạn CPU và RAM cho một container Redis.

Phần 3: Docker Compose file

Bài tập 1: Triển khai WordPress với MySQL

Mục tiêu: Tạo stack WordPress kết nối với MySQL, sử dụng volumes để lưu trữ dữ liệu.

Yêu cầu:

1. Sử dụng image wordpress:latest (port 80).

2. Sử dụng image mysql:5.7 (port 3306).

3. Volume cho database (/var/lib/mysql).

4. Biến môi trường cho MySQL:

MYSQL\_ROOT\_PASSWORD, MYSQL\_DATABASE, MYSQL\_USER, MYSQL\_PASSWORD

Gợi ý:

WordPress cần khai báo depends\_on MySQL.

Sử dụng network tùy chỉnh để kết nối giữa 2 service.

Bài tập 2: Ứng dụng Node.js + MongoDB

Mục tiêu: Triển khai ứng dụng Node.js (lưu dữ liệu vào MongoDB) và MongoDB với volume.

Yêu cầu:

1. Viết Dockerfile cho ứng dụng Node.js (ví dụ: REST API đơn giản).

2. Sử dụng image mongo:latest (port 27017).

3. Volume cho MongoDB (/data/db).

4. Đảm bảo Node.js service khởi động sau MongoDB (depends\_on + healthcheck).

Bài tập 3: Load Balancing với Nginx + Flask

Mục tiêu: Cân bằng tải giữa 2 instance Flask dùng Nginx.

Yêu cầu:

1. 2 service Flask (sử dụng app.py từ bài tập trước, port 5000).

2. 1 service Nginx (port 8080) cấu hình làm reverse proxy:

Chuyển request / đến các Flask instance (round-robin).

3. Tạo custom network và Nginx config.

Bài tập 4: Prometheus + Grafana Monitoring

Mục tiêu: Giám sát Docker containers dùng Prometheus và Grafana.

Yêu cầu:

1. Service Prometheus (port 9090) với file cấu hình thu thập metrics từ Docker.

2. Service Grafana (port 3000) kết nối đến Prometheus.

3. Volume để lưu dữ liệu Prometheus và Grafana.

Bài tập 5: Multi-tier Voting App

Mục tiêu: Triển khai ứng dụng voting gồm 5 services (Tham khảo từ Docker Docs).

Yêu cầu:

1. Frontend: vote (Python, port 5000).

2. Backend: result (Node.js, port 5001).

3. Redis (lưu tạm vote).

4. Worker (Java) xử lý vote từ Redis sang DB.

5. Postgres (lưu kết quả).