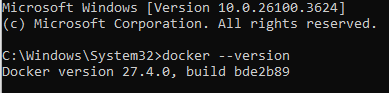
|  |  |
| --- | --- |
| **Họ và Tên** | Kiều Anh Hải |
| **MSSV** | 21056271 |

# **Phần 1: Các lệnh cơ bản thao tác với Docker**

## **1. Hiển thị phiên bản Docker hiện tại.**

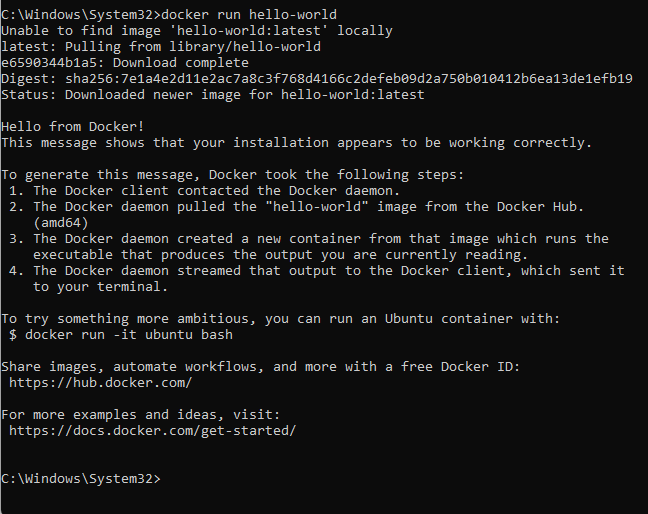
docker --version



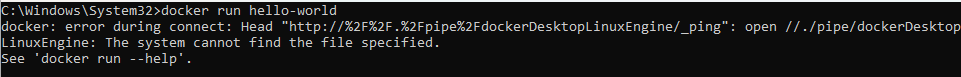
* Nếu lệnh này không hoạt động hoặc báo lỗi, có thể Docker chưa được cài đặt hoặc dịch vụ Docker chưa chạy.

## **2. Tạo và chạy một container từ image hello-world.**

docker run hello-world

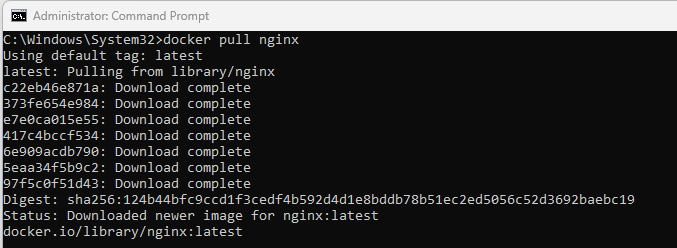


* Nếu gặp lỗi này, có thể là chưa mở app Docker Desktop.



## **3. Tải (pull) image Nginx từ Docker Hub về máy.**

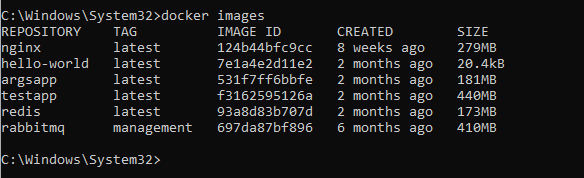
docker pull nginx



🡪 Lệnh này tải image Nginx từ Docker Hub về máy nhưng không chạy nó ngay lập tức.

## **4. Hiển thị danh sách tất cả các images.**

docker images



🡪 Đây là danh sách tất cả các images đã tải về máy cục bộ.

## **5. Chạy container từ image nginx.**

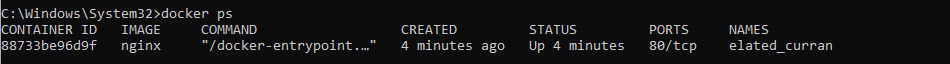
docker run -d nginx



🡪 Lệnh này dùng để chạy container từ image nginx trong chế độ nền (detached mode), tức là không chiếm terminal – container vẫn chạy ngay cả khi bạn đóng terminal.

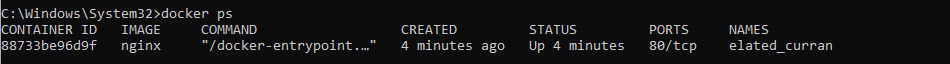
## **6. Liệt kê tất cả các container đang chạy.**

docker ps



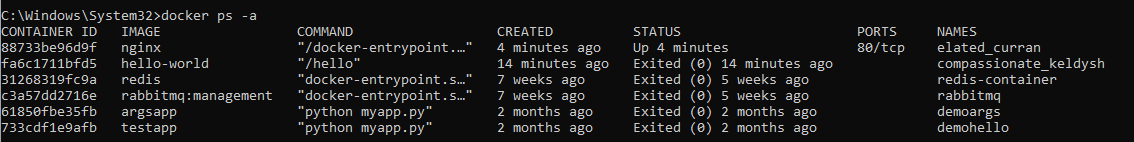
* **Giải thích các cột:**
* CONTAINER ID: ID của container (ngắn gọn, chỉ cần dùng 1 phần ID là đủ).
* IMAGE: Image mà container này được tạo từ đó (ví dụ: nginx).
* COMMAND: Lệnh được chạy khi container khởi động (ví dụ: /docker-entrypoint. …).
* STATUS: Trạng thái hiện tại của container (ví dụ: Up 4 minutes có nghĩa là container đã chạy được 5 giây).
* PORTS: Cổng mà container đang mở ra ngoài (nếu có cấu hình).
* NAMES: Tên của container (nếu không chỉ định, Docker tự động gán một tên ngẫu nhiên).

🡪 Nếu không có container nào, sẽ không thấy gì – hoặc hiển thị như này:



## **7. Liệt kê tất cả các container trên máy.**

docker ps -a

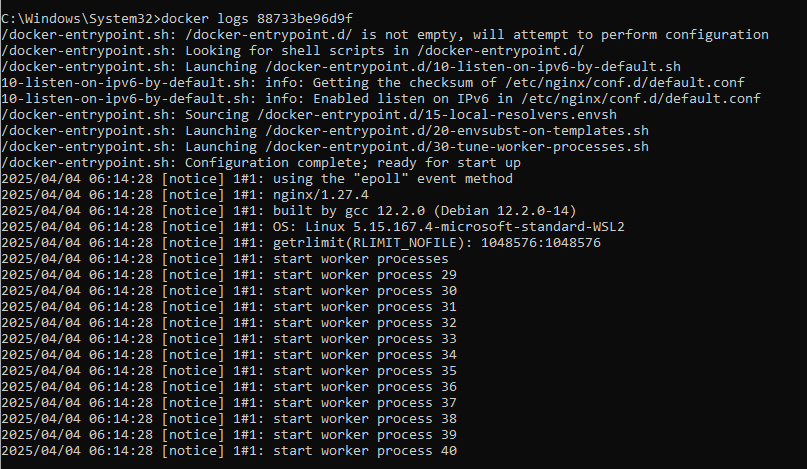


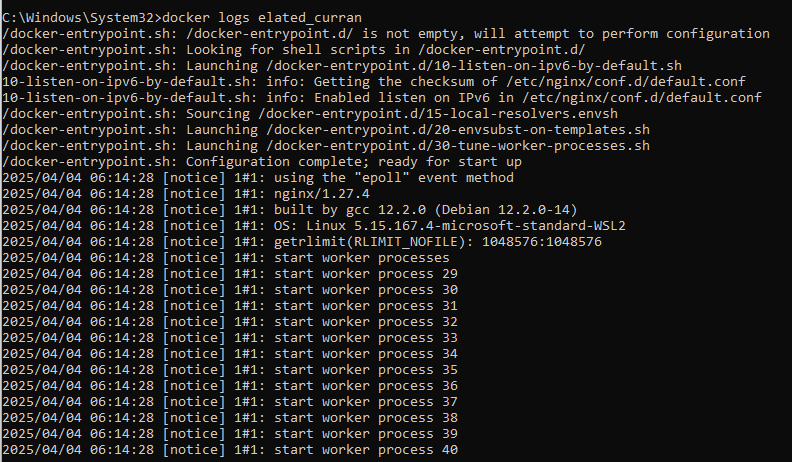
* Các cột được giải thích tương tự câu lệnh docker ps.

🡪 Lệnh này liệt kê tất cả các container trên máy bao gồm cả những container đang chạy và đã dừng.

## **8. Xem log (dữ liệu ghi lại) của một container cụ thể .**

docker logs <container\_id>

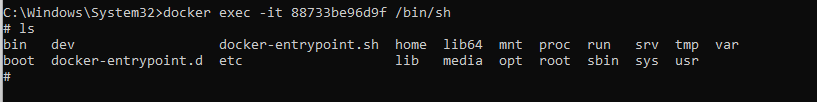




🡪 Có thể xem log dựa trên container\_id hoặc container\_name.

## **9. Chạy một shell (terminal) trong container đang chạy**

docker exec -it <container\_id> /bin/sh



Khi sử dụng Docker, bạn có thể mở shell trong container với docker exec -it <container\_id> bash hoặc docker exec -it <container\_id> sh.

**Bash** là shell mạnh mẽ và phổ biến, hỗ trợ tính năng như hoàn thành tự động, lịch sử lệnh, và cú pháp phức tạp trong scripting. Nó lý tưởng khi cần môi trường đầy đủ với nhiều tính năng.

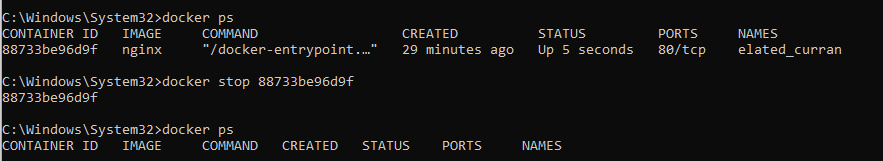
**Sh**, ngược lại, là shell cổ điển và đơn giản, không có các tính năng nâng cao như **bash**, nhưng nhẹ hơn và tiêu tốn ít tài nguyên, phù hợp cho các container yêu cầu sự tối giản.

Tóm lại, chọn **bash** cho môi trường đầy đủ tính năng, còn **sh** cho các tác vụ đơn giản và container nhẹ.

🡪 Lệnh này cho phép thực thi lệnh trong container mà không cần phải dừng hoặc khởi động lại container.

## **10. Dừng một container đang chạy trên Docker.**

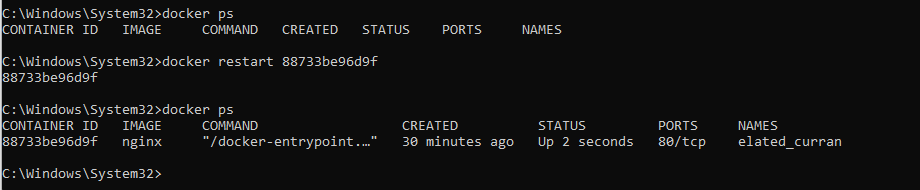
docker stop <container\_id>



🡪 Lệnh này rất hữu ích để dừng một container đang chạy mà không làm gián đoạn các container khác. Có thể dừng một hoặc nhiều container cùng lúc và tùy chỉnh thời gian dừng nếu cần thiết.

## **11. Khởi động lại một container đang chạy hoặc đã dừng.**

docker restart <container\_id>

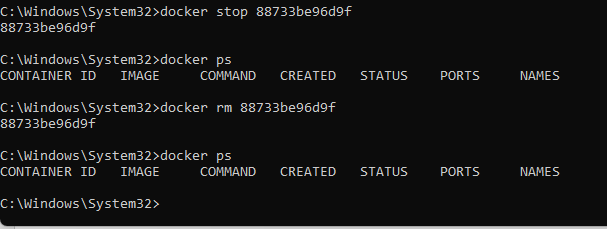


🡪 Lệnh này dùng để khởi động lại một container đang chạy hoặc đã dừng. Docker sẽ dừng container hiện tại và ngay lập tức khởi động lại container đó.

🡪 Lệnh docker restart giúp dừng và khởi động lại một container, rất hữu ích khi cần áp dụng thay đổi cấu hình hoặc khắc phục sự cố. Có thể chỉ định thời gian chờ và khởi động lại nhiều container cùng lúc.

## **12. Xóa một container đã dừng.**

docker rm <container\_id>



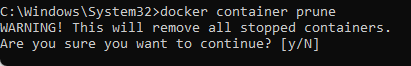
* Khi gặp lỗi này – có nghĩa là container này chưa dừng hoạt động – cần dừng trước khi xóa.



🡪 Lệnh docker rm giúp xóa một container đã dừng, giải phóng tài nguyên hệ thống.

## **13. Xóa tất cả các container đã dừng trong Docker.**

docker container prune



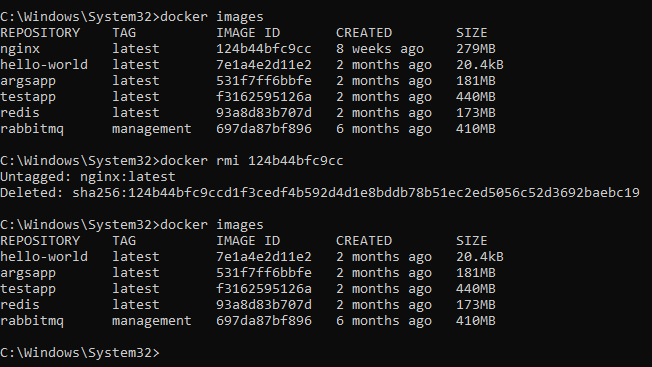
* Nếu muốn xóa thì nhập ‘y’, không muốn thì nhập ‘N’.

🡪 Lệnh docker container prune giúp xóa tất cả các container đã dừng, giúp dọn dẹp tài nguyên hệ thống. Lệnh này rất hữu ích khi muốn giải phóng không gian mà không cần phải xóa từng container một cách thủ công.

Nếu muốn dọn dẹp hệ thống Docker một cách tổng quát, có thể kết hợp lệnh docker system prune để xóa các tài nguyên không sử dụng, bao gồm cả images, volumes, và networks.

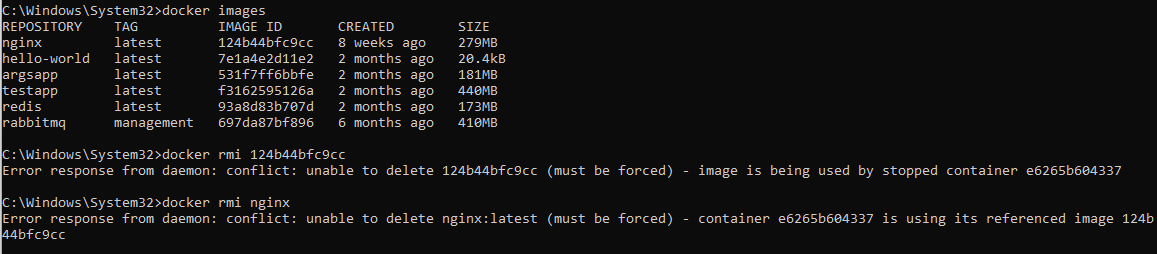
## **14. Xóa một image khỏi hệ thống Docker.**

docker rmi <image\_id>



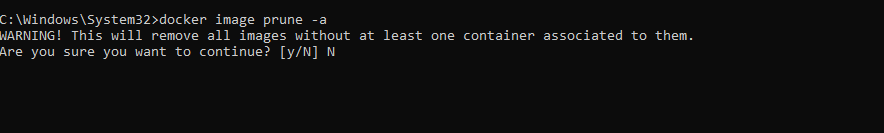
🡪 Lệnh docker rmi giúp xóa một image khỏi hệ thống Docker, giải phóng không gian lưu trữ. Tuy nhiên, nếu image đang được sử dụng bởi container, bạn sẽ không thể xóa image này cho đến khi container được xóa hoặc ngừng sử dụng image đó. Bạn cũng có thể sử dụng tuỳ chọn -f để buộc xóa image ngay cả khi có container đang sử dụng.

* Nếu gặp lỗi – có thể do images này đang được sử dụng bởi 1 container khác, cần dừng và xóa container đang hoạt động để có thể xóa images này.



## **15. Xóa tất cả các image không sử dụng khỏi hệ thống Docker.**

docker image prune -a

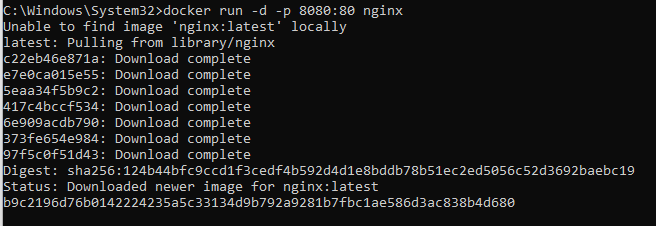


* Nếu muốn xóa thì nhập ‘y’, không muốn thì nhập ‘N’.

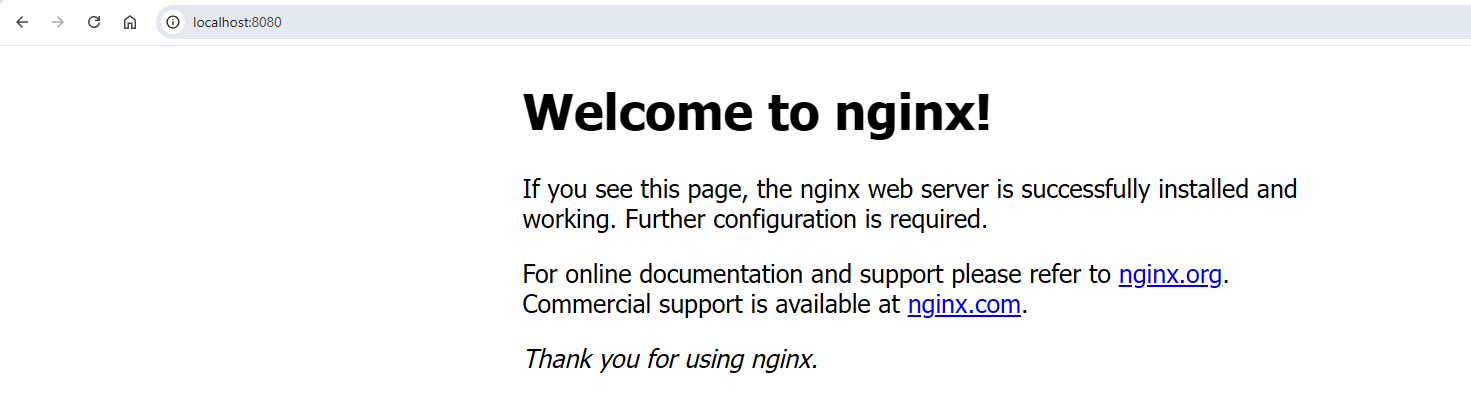
🡪 Lệnh docker image prune -a được sử dụng để xóa tất cả các image không sử dụng khỏi hệ thống Docker, bao gồm cả những image không còn được container nào sử dụng và những image không phải là "dangling" (image không gắn với bất kỳ container nào).

## **16. Tạo một container mới từ image nginx và chạy ở chế độ background và map port 8080 trên máy chủ tới port 80 trong container.**

docker run -d -p 8080:80 nginx

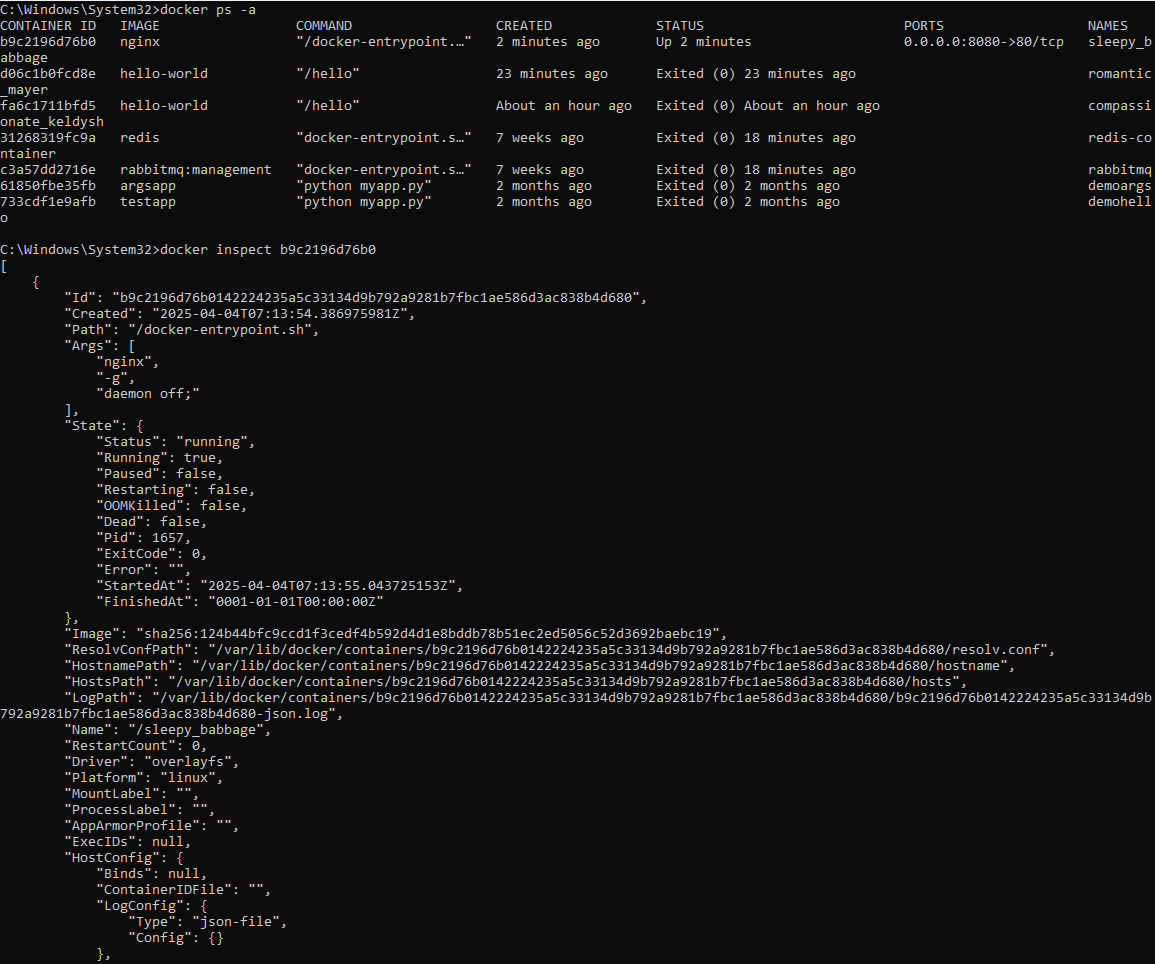


🡪 Lệnh docker run -d -p 8080:80 nginx sẽ tạo một container mới từ image nginx, chạy ở chế độ background và map port 8080 trên máy chủ tới port 80 trong container. Bạn có thể truy cập vào nginx qua <http://localhost:8080>.



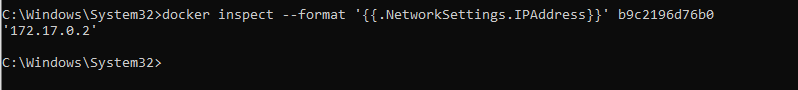
## **17. Lấy thông tin chi tiết về container.**

docker inspect <container\_id>



🡪 Lệnh docker inspect <container\_id> giúp lấy thông tin chi tiết về container, bao gồm cấu hình mạng, volumes, môi trường, và trạng thái của container.

🡪 Có thể sử dụng tuỳ chọn --format để lọc thông tin cụ thể.



## **18. Chạy một container nginx từ image nginx ở chế độ background và gắn kết một volume từ máy chủ vào container.**

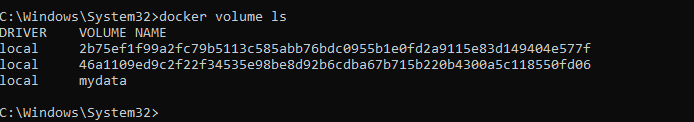
docker run -d -v mydata:/data nginx



🡪 Lệnh này được sử dụng để chạy một container nginx từ image nginx ở chế độ background và gắn kết một volume từ máy chủ vào container. Volume mydata được gắn vào thư mục /data trong container, cho phép lưu trữ dữ liệu ngoài container, giúp bảo vệ dữ liệu khỏi bị mất khi container bị xóa.

## **19. Liệt kê tất cả các volumes hiện có trên hệ thống Docker.**

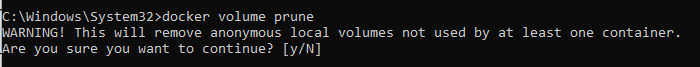
docker volume ls



🡪 Lệnh docker volume ls giúp liệt kê tất cả các volumes hiện có trên hệ thống Docker, cho phép kiểm tra và quản lý các volume được Docker sử dụng để lưu trữ dữ liệu.

## **20. Xóa tất cả các volumes không còn được sử dụng bởi bất kỳ container nào.**

docker volume prune



* Nếu muốn xóa thì nhập ‘y’, không muốn thì nhập ‘N’.

🡪 Lệnh docker volume prune giúp xóa các volumes không sử dụng, giải phóng dung lượng ổ đĩa và giúp hệ thống Docker của bạn gọn gàng hơn.

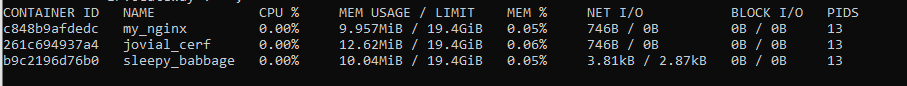
## **21. Tạo một container từ image nginx, chạy ở chế độ background và đặt tên cho container là my\_nginx.**

docker run -d --name my\_nginx nginx



## **22. Hiển thị thông tin thống kê thời gian thực về tài nguyên hệ thống mà các container Docker đang sử dụng**

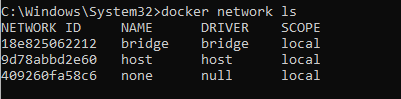
docker stats



🡪 Lệnh docker stats được sử dụng để hiển thị thông tin thống kê thời gian thực về tài nguyên hệ thống mà các container Docker đang sử dụng. Thông tin bao gồm CPU, RAM, mạng, và I/O của mỗi container, giúp giám sát và tối ưu hiệu suất của hệ thống Docker.

## **23. Liệt kê tất cả các mạng (networks) mà Docker đang quản lý trên hệ thống.**

docker network ls



* Giải thích cột :
* **NETWORK ID**: ID duy nhất của mạng.
* **NAME**: Tên của mạng.
* **DRIVER**: Loại driver mạng (ví dụ: bridge, host, none).
* **SCOPE**: Phạm vi của mạng, thường là local đối với các mạng Docker.

🡪 Lệnh docker network ls được sử dụng để liệt kê tất cả các mạng (networks) mà Docker đang quản lý trên hệ thống của bạn. Các mạng này cho phép các container Docker giao tiếp với nhau và với bên ngoài.

## **24. Tạo một mạng Docker mới với tên là my\_network.**

docker network create my\_network



🡪 Lệnh docker network create my\_network được sử dụng để tạo một mạng Docker mới với tên là my\_network. Mạng này sẽ cho phép các container kết nối với nhau, và có thể cấu hình các thuộc tính mạng như kiểu driver và phạm vi.

## **25. Tạo và chạy một container từ image nginx, kết nối container vào mạng my\_network, và đặt tên cho container là my\_container**

docker run -d --network my\_network --name my\_container nginx



🡪 Lệnh docker run -d --network my\_network --name my\_container nginx tạo và chạy một container từ image nginx, kết nối container vào mạng my\_network, và đặt tên cho container là my\_container để dễ dàng quản lý.

## **26. Kết nối container có tên my\_nginx vào mạng Docker có tên my\_network.**

docker network connect my\_network my\_nginx



🡪 Lệnh docker network connect my\_network my\_nginx kết nối container có tên my\_nginx vào mạng Docker có tên my\_network, giúp container có thể giao tiếp với các container khác trong mạng đó.

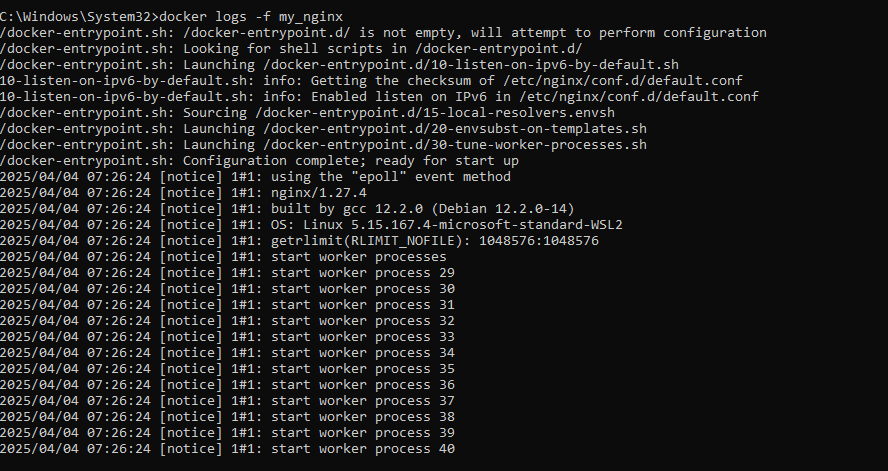
## **27. Tạo và chạy một container từ image nginx, thiết lập biến môi trường MY\_ENV với giá trị hello\_world và chạy container ở chế độ background.**

docker run -d -e MY\_ENV=hello\_world nginx



## **28. Theo dõi log của container**

docker logs -f my\_nginx



🡪 Lệnh docker logs -f my\_nginx sẽ giúp **theo dõi log** của container my\_nginx trong thời gian thực. Có thể sử dụng lệnh này để xem các thông báo hoặc lỗi mà container tạo ra trong suốt thời gian chạy.

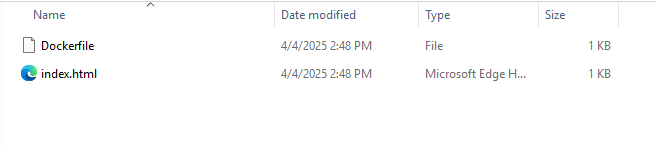
## **29. Tạo Dockerfile**

Bước 1: Tạo file Dockerfile chứa nội dung.

FROM nginx

COPY index.html /usr/share/nginx/html/index.html

Bước 2: Tạo file index.html đơn giản.

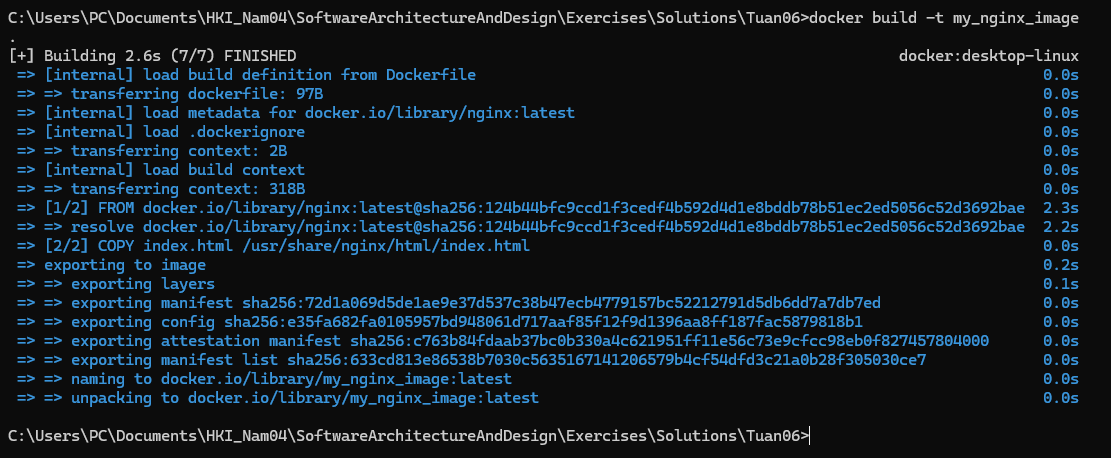


* 2 file phải nằm cùng vị trí.

Bước 3: cmd tại vị trí 2 file để làm bước 30.

## **30. Xây dựng một Docker image từ Dockerfile trong thư mục hiện tại (. đại diện cho thư mục hiện tại).**

docker build -t my\_nginx\_image .



## **31. Chạy một container từ Docker image**

docker run -d -p 8080:80 my\_nginx\_image



🡪 Lệnh docker run -d -p 8081:80 my\_nginx\_image được sử dụng để chạy một container từ Docker image đã xây dựng trước đó (my\_nginx\_image) và ánh xạ cổng 8081 của máy chủ đến cổng 80 của container. [http://localhost:8081](http://localhost:80810)

