1 docker –version

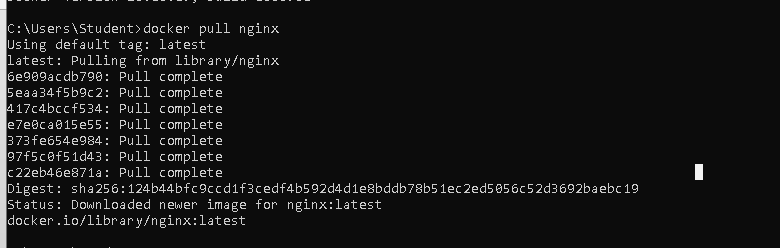


Check version cua docker

2 docker run hello-world

Run 1 container với docker

3 docker pull nginx



Pull images nginx

4 docker images



Show ra các images có trong docker engine

5 docker run -d nginx



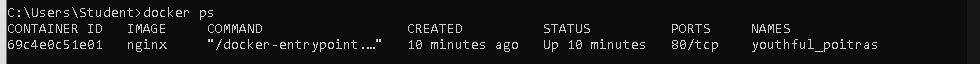
Chạy imamge nginx vừa pull

sử dụng -d, Docker sẽ chạy container ở chế độ nền và trả về ID của container, giúp bạn có thể tiếp tục làm việc với terminal mà không bị gián đoạn bởi container đang chạy

 docker run không chỉ chạy một **container**, mà còn tự động tạo ra một **container** từ một **image**. Thực tế, khi bạn dùng docker run, Docker sẽ **lấy image** (nếu nó chưa có trên máy) và tạo một **container** từ image đó để chạy ứng dụng hoặc dịch vụ bên trong container.

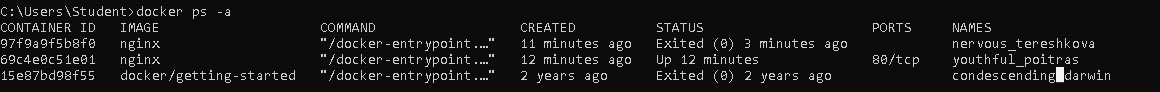
 Vậy, docker run là cách bạn **khởi động một container từ một image**. Docker sẽ dùng image để tạo ra một bản sao của môi trường và chạy nó dưới dạng container.

6 docker ps



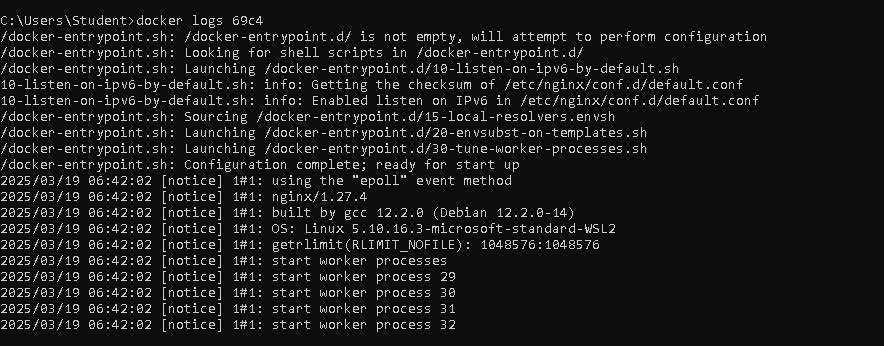
Check các container đang chạy

7 docker ps –a



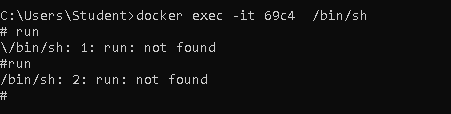
List các docker container trong docker engine

8 docker logs <container\_id>



Kiểm tra logs của 1 container

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



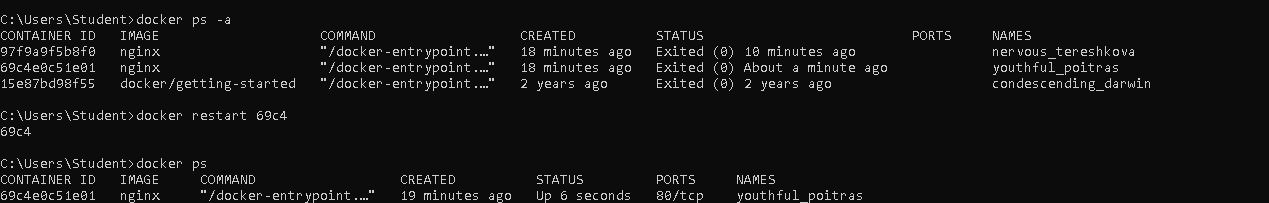
sẽ mở một terminal **interactive shell** bên trong container với ID hoặc tên là <container\_id>. Bạn có thể sử dụng lệnh này để **truy cập vào môi trường** của container và thực hiện các thao tác, như kiểm tra các file hoặc cài đặt các phần mềm khác

10 docker stop <container\_id>



Dừng 1 container nó sẽ trả về id của container đã stop

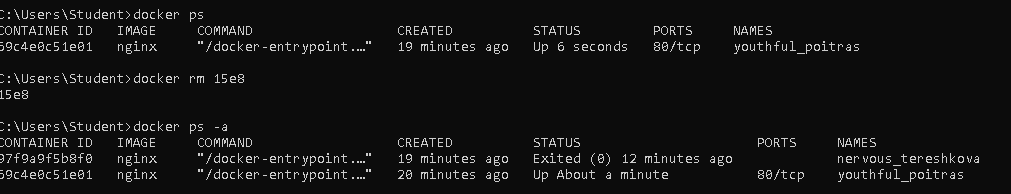
11 docker restart <container\_id>



Khởi động lại 1 container có id tương ứng nó sẽ trả về id của container docker đã run

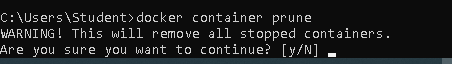
12 docker rm <container\_id>

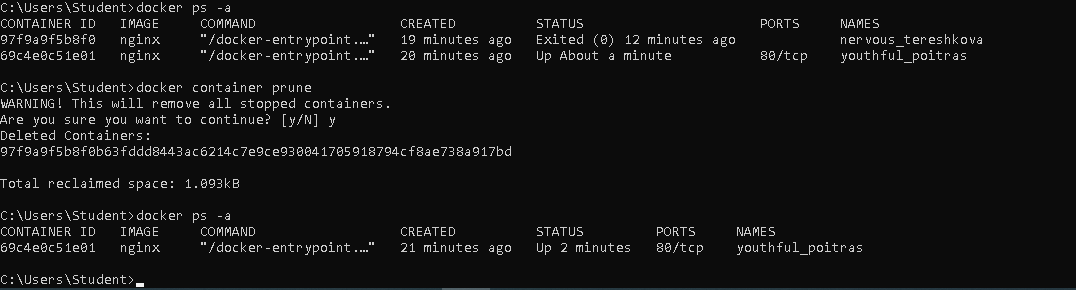




Xóa 1 container và trả về id có container tương ứng

13 docker container prune





Xóa tất cả các container đang stop

14 docker rmi <image\_id>



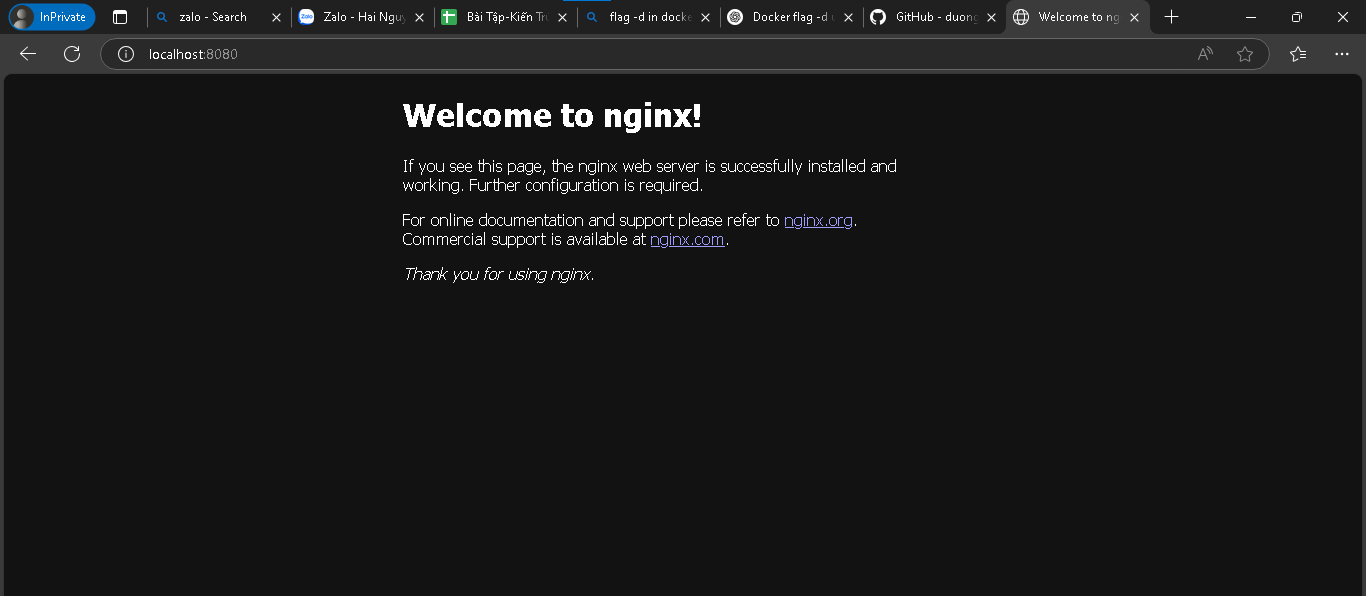
Xóa 1 images có id tương ứng hiện tại không thể xóa vì nó đang được sử dụng để chạy container có id là 69c4

15 docker image prune –a



Xóa tất cả các image không có container nào sử dụng để chạy

16 docker run -d -p 8080:80 nginx





Flag –d Chạy container này dưới nền cho phép tiếp tục thao tác trên console mà không bị gián đoạn

Flag –p là port nó sẽ chạy

Khi bạn truy cập vào cổng 8080 trên máy, Docker sẽ chuyển tiếp (forward) traffic đến cổng 80 của container (nơi nginx đang chạy).

17 docker inspect <container\_id>



Hiển thị các thông tin của container đó

18 docker run -d -v mydata:/data nginx

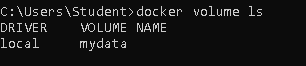


Lệnh docker run -d -v mydata:/data nginx sẽ:

1. Tạo một container từ image nginx và chạy nó ở chế độ nền (-d).
2. Gắn một volume mydata vào thư mục /data trong container, giúp dữ liệu trong /data được lưu trữ bên ngoài container, đảm bảo rằng dữ liệu không bị mất khi container bị xóa hoặc restart.
   * Nếu volume mydata chưa tồn tại, Docker sẽ tự động tạo nó.

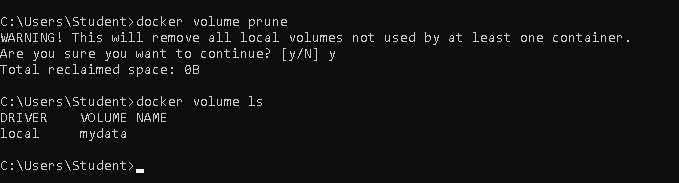
**Volume** trong Docker là cách để lưu trữ dữ liệu bên ngoài container, giúp dữ liệu không bị mất khi container bị xóa hoặc restart.

19 docker volume ls



Coi các volume trong docker

20 docker volume prune

Xóa tất cả các volume không sử dụng đến   
  


21 docker run -d --name my\_nginx nginx

chạy một container từ một image(nginx) và đặt tên cho container này



22 docker stats

docker stats trong Docker dùng để **hiển thị các thông tin thống kê thời gian thực** về tài nguyên hệ thống mà các container đang sử dụng. Các thống kê này bao gồm **CPU usage**, **Memory usage**, **Network I/O**, **Disk I/O**, và các chỉ số khác liên quan đến hiệu suất của container.

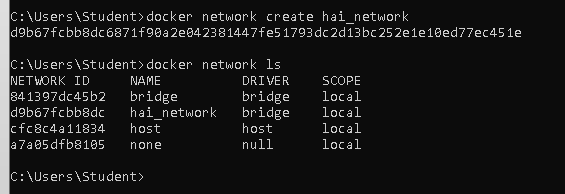
Docker sẽ hiển thị thông tin thống kê cho tất cả các container đang chạy trên hệ thống của bạn. Thông tin này sẽ được cập nhật liên tục trong quá trình chạy.

23 docker network ls



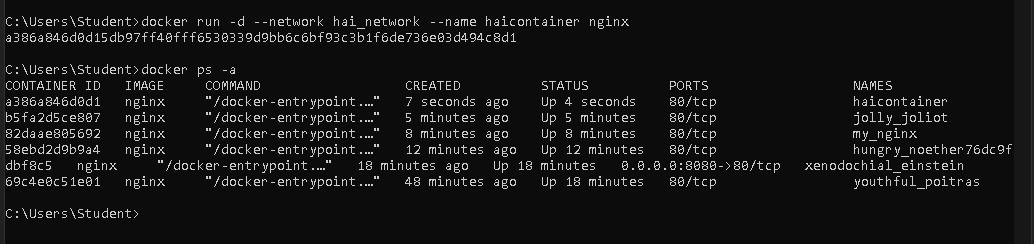
Coi các network có trong docker

24 docker network create my\_network



Tạo ra 1 network có tên hải

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



chạy một container từ image nginx dạng ngầm với flag –d

**--network my\_network**: Tùy chọn này xác định **mạng (network)** mà container sẽ tham gia

my\_network: Đây là tên của mạng Docker mà bạn muốn kết nối container vào

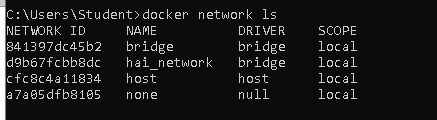
 **Tạo và chạy một container** từ image nginx.

 Chạy container trong **chế độ nền** (-d).

 Kết nối container vào **mạng Docker** có tên my\_network.

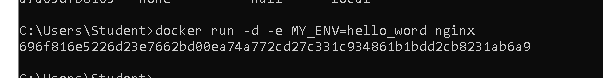
 Đặt tên cho container là **my\_container** thay vì sử dụng tên ngẫu nh

26 docker network connect my\_network my\_nginx

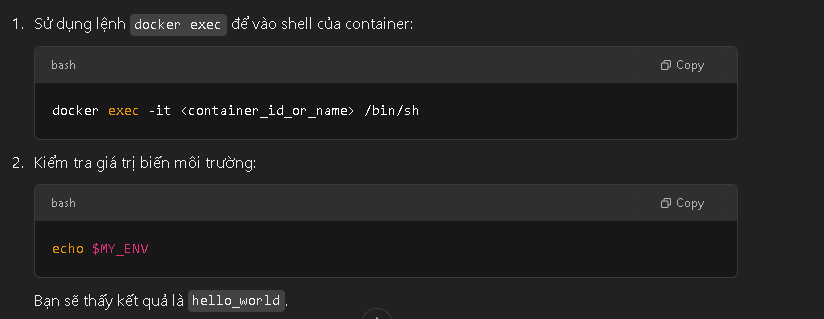


Lệnh này sẽ kết nối **container my\_nginx** vào **mạng** my\_network (mạng này có thể là mạng bridge, host, hoặc một mạng tùy chỉnh trong Docker). Sau khi thực hiện lệnh này, container my\_nginx sẽ có thể giao tiếp với các container khác trong mạng my\_network

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



**thiết lập biến môi trường** (environment variable) cho container khi nó được khởi động với giá trị là hello word của container nginx



28 docker logs -f my\_nginx

sẽ hiển thị **log** (dữ liệu ghi lại) của container có tên là my\_nginx và theo dõi log đó trong thời gian thực.

Lệnh này sẽ thấy các log mới được cập nhật ngay lập tức khi chúng xuất hiện, giống như đang "streaming" log của container. Nó rất hữu ích khi muốn giám sát hoạt động của container một cách liên tục.

29 FROM nginx

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

30 docker build -t my\_nginx\_image .

Build 1 một **Docker image** từ một **Dockerfile** và các tài nguyên có sẵn trong thư mục hiện tại

**-t my\_nginx\_image**: Tùy chọn -t dùng để **đặt tên (tag)** cho image mới. Ở đây, image sẽ được gán tên là my\_nginx\_image.

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

Run container này trên port 8080 với container vừa tạo