**Nguyễn Đình Bình MSSV: 23021479**

**Docker** là một nền tảng containerization (đóng gói ứng dụng trong container) (chạy từng ứng dụng trong container độc lập.).

Container là một môi trường độc lập, nhẹ, chạy được trên mọi máy (Linux, Windows, Mac) mà không lo bị khác môi trường.

Thay vì cài trực tiếp app + dependency vào máy, bạn viết một file Dockerfile → định nghĩa app cần gì (ngôn ngữ, thư viện, port, dữ liệu, …).

Sau đó build thành Docker Image (giống như snapshot), rồi chạy nó thành Container (instance chạy thực tế).

Ưu điểm:

Portable: chạy được ở bất kỳ đâu có Docker (dev, test, production).

Nhẹ hơn máy ảo: dùng chung kernel hệ điều hành, khởi động nhanh.

Tái sử dụng, dễ triển khai: chỉ cần docker run là chạy app.

**Docker Compose** là một công cụ để định nghĩa và chạy nhiều container cùng lúc. Bạn viết file docker-compose.yml → mô tả cách các container kết nối với nhau (ví dụ: app backend, frontend, database). Chỉ cần 1 lệnh docker-compose up -d → toàn bộ hệ thống chạy (backend, frontend, database…).

**Cả Linux, Unix, BSD, \*nix, và macOS** đều thuộc loại hệ điều hành giống Unix (Unix-like). Chúng có chung một số đặc điểm và triết lý thiết kế cơ bản được phát triển từ hệ điều hành Unix ban đầu.

**Unix:** Đây là hệ điều hành gốc, được phát triển tại Bell Labs vào những năm 1970. Nó là nền tảng cho sự ra đời của các hệ điều hành khác, nhưng các phiên bản của nó thường là độc quyền và thương mại. Cấu trúc hệ thống tệp phân cấp, các công cụ dòng lệnh mạnh mẽ, và triết lý "mọi thứ đều là tệp tin"

**\*nix:** Đây là một thuật ngữ chung, bao gồm tất cả các hệ điều hành có chức năng và giao diện tương tự Unix. Linux, BSD, và macOS đều được xếp vào nhóm này.

**Linux:** Đây là một nhân (kernel) (là phần cốt lõi của hệ điều hành.) mã nguồn mở và miễn phí. Các bản phân phối Linux như Ubuntu hay Fedora là các hệ điều hành hoàn chỉnh được xây dựng trên nhân này.

**BSD (Berkeley Software Distribution):** Ban đầu là một nhánh của Unix, nhưng đã phát triển thành một hệ điều hành độc lập với mã nguồn riêng, cũng là mã nguồn mở. Các biến thể nổi tiếng là FreeBSD, OpenBSD và NetBSD.

**macOS:** Là một hệ điều hành độc quyền của Apple dành cho máy tính Mac, được xây dựng trên một nhân giống Unix có tên là Darwin. Dù có giao diện đồ họa khác biệt, lõi của macOS vẫn tuân thủ nhiều tiêu chuẩn của Unix.

**Alpine và Ubuntu** đều là các bản phân phối Linux, nhưng chúng có triết lý thiết kế và mục đích sử dụng rất khác nhau. Dưới đây là thông tin chi tiết về từng loại và so sánh giữa chúng.

1. Ubuntu

Nguồn gốc: Dựa trên Debian (2004, Canonical Ltd).

Đặc điểm:

Thân thiện, dễ dùng, có giao diện đồ họa đẹp (GNOME).

Phổ biến nhất trong thế giới Linux desktop.

Rất nhiều tài liệu, cộng đồng đông đảo.

Hỗ trợ đầy đủ: server, desktop, cloud, IoT.

Package manager: APT (Advanced Package Tool).

Dung lượng image Docker: khá nặng (200–300MB cho ubuntu:20.04), vì chứa nhiều thư viện, công cụ mặc định.

Phù hợp dev: cài gì cũng có, gần giống môi trường production server truyền thống.

Ưu điểm:

Dễ học, dễ tiếp cận cho người mới.

Kho phần mềm khổng lồ, hỗ trợ phần cứng tốt.

Có cả bản LTS (Long Term Support) ổn định 5 năm.

Hỗ trợ nhiều ngôn ngữ lập trình, framework, database.

Dễ debug (có sẵn nhiều công cụ như bash, curl, apt-get).

Thường được chọn làm base image cho các ứng dụng enterprise.

Nhược điểm:

So với distro tối giản → khá nặng (cài sẵn nhiều package).

Không tối ưu tuyệt đối cho hiệu năng hay bảo mật tối giản.

Nặng → tốn dung lượng, khởi động container lâu hơn Alpine.

Attack surface lớn hơn (nhiều gói → nhiều khả năng dính lỗ hổng bảo mật).

2. Alpine Linux

Nguồn gốc: 2005, cộng đồng phát triển độc lập.

Đặc điểm:

Distro siêu nhẹ, tối giản.

Dùng musl libc (khác glibc thường gặp) và BusyBox (thay thế GNU coreutils).

Không dành cho người mới, thiên về môi trường server, container, bảo mật.

Package manager: apk.

Ưu điểm:

Dung lượng nhỏ, khởi động nhanh.

Ít thành phần dư → attack surface nhỏ → bảo mật cao.

Lý tưởng cho embedded system, router, firewall, Docker image base.

Image Docker mặc định chỉ ~5MB (alpine:3.19 chẳng hạn).

Nhược điểm:

Ít phần mềm pre-install → nhiều thứ phải cài thêm.

Một số app không tương thích (do musl thay glibc).

Ít thân thiện với người mới, không phù hợp desktop.

Một số thư viện (nhất là Python, Java, TensorFlow, PyTorch…) có thể không tương thích, build khó hơn.

Debug bất tiện (nhiều tool không có sẵn, phải tự cài thêm).

Khi compile từ source, có thể phát sinh lỗi khó trace hơn so với Ubuntu/Debian.

- Dùng Ubuntu khi: Làm việc với ứng dụng phức tạp, nặng, cần nhiều dependency (Python ML, Java, Node.js + nhiều package gốc). Muốn môi trường gần giống server production (đa phần production server Linux đều dựa trên Debian/Ubuntu hoặc CentOS). Dev/debug thường xuyên, cần đủ công cụ cài sẵn.

- Dùng Alpine khi: Làm microservices nhỏ, không phụ thuộc nhiều thư viện C phức tạp. Ưu tiên tối ưu kích thước image, giảm thời gian pull/push (CI/CD, serverless). Triển khai trên cloud, muốn tăng bảo mật, giảm attack surface. Ứng dụng Go, Rust, C#, Node.js (ít native dependency).

**VNC (Virtual Network Computing)** là một công nghệ truy cập và điều khiển máy tính từ xa qua giao diện đồ họa (GUI). Nó hoạt động theo mô hình client–server:

VNC Server: chạy trên máy cần điều khiển.

VNC Viewer (Client): chạy trên máy dùng để điều khiển.

→ Khi kết nối, người dùng nhìn thấy desktop từ xa và thao tác như đang ngồi trực tiếp trước máy đó (chuột, bàn phím).

VNC dựa trên RFB (Remote Frame Buffer Protocol). Cách thức:

Server gửi hình ảnh màn hình (framebuffer) đến Client.

Client hiển thị hình ảnh đó cho người dùng.

Các sự kiện chuột, bàn phím từ Client gửi ngược về Server.

Mọi thao tác được đồng bộ theo thời gian thực.

Đặc điểm của VNC

Đa nền tảng: chạy được trên Windows, Linux, macOS, thậm chí di động.

Không phụ thuộc hệ điều hành: chỉ cần VNC Viewer là có thể kết nối đến bất kỳ OS nào có VNC Server.

Bảo mật: bản gốc VNC không mã hóa mạnh (thường dùng kết hợp SSH hoặc VPN để bảo mật hơn).

Hiệu suất: vì truyền cả framebuffer → đôi khi chậm hơn RDP (Remote Desktop Protocol của Microsoft).

Ứng dụng thực tế

Quản trị hệ thống: admin điều khiển server/PC từ xa qua GUI.

Hỗ trợ kỹ thuật (remote support): IT hỗ trợ người dùng từ xa.

Truy cập máy cá nhân từ xa: ví dụ, dùng điện thoại để điều khiển máy ở nhà.

Môi trường nhúng/IoT: truy cập GUI của hệ thống chạy headless.