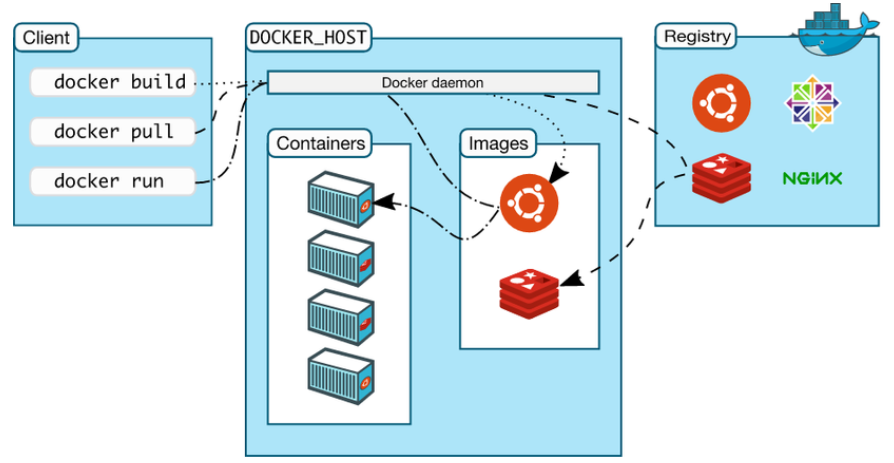
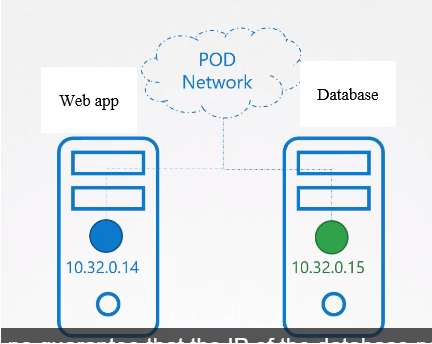
**Khái niệm Docker**

Mục đích: Không muốn tốn khá nhiều thời gian để config máy local và server cùng một môi trường để chạy được ứng dụng. Bạn chỉ cần build 1 lần chạy ở nhiều nơi mà thôi.



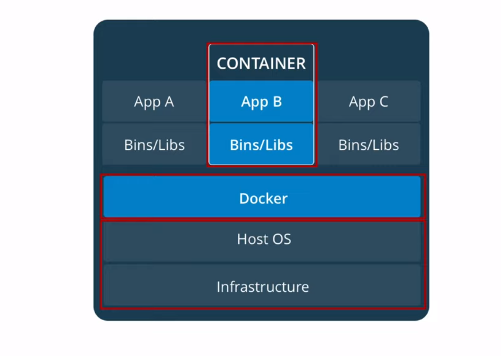
* **Docker Engine** : là thành phần chính của Docker, như một công cụ để đóng gói ứng dụng
* **Docker Hub** : là một “github for docker images”. Trên DockerHub có hàng ngàn public images được tạo bởi cộng đồng cho phép bạn dễ dàng tìm thấy những image mà bạn cần. Và chỉ cần pull về và sử dụng với một số config mà bạn mong muốn.
* **Images**: là một khuôn mẫu để tạo một container. Thường thì image sẽ dựa trên 1 image có sẵn với những tùy chỉnh thêm. Ví dụ bạn build 1 image dựa trên image Centos mẫu có sẵn để chạy Nginx và những tùy chỉnh, cấu hình để ứng dụng web của bạn có thể chạy được. Bạn có thể tự build một image riêng cho mình hoặc sử dụng những image được chia sẽ từ cộng đồng Docker Hub. Một image sẽ được build dựa trên những chỉ dẫn của Dockerfile.
* **Container**:  hoạt động giống như một thư mục (directory), chứa tất cả những thứ cần thiết để một ứng dụng có thể chạy được. Mỗi một docker container được tạo ra từ một docker image. Các thao tác với một container : chạy, bật, dừng, di chuyển, và xóa
* **Docker Client**: là một công cụ giúp người dùng giao tiếp với Docker host.
* **Docker Daemon**: lắng nghe các yêu cầu từ Docker Client để quản lý các đối tượng như Container, Image, Network và Volumes thông qua REST API. Các Docker Daemon cũng giao tiếp với nhau để quản lý các Docker Service.
* **Dockerfile**: là một file chứa tập hợp các lệnh để Docker có thể đọc và thực hiện để đóng gói một image theo yêu cầu người dùng
* **Volumes**: là phần dữ liệu được tạo ra khi container được khởi tạo.
* **Docker registries** : Là kho chứa images. Người dùng có thể tạo ra các images của mình và tải lên đây hoặc tải về các images được chia sẻ



Pod 1 là web app

Pod 2 là database

Web app có thể trỏ đến database bằng network internal của cluster nhưng ko đảm bảo rằng ip của data luôn được giữ nguyên



Máy vật lý

Host OS, Infrastructure gọi là máy host (máy tính), máy host này cài docker

Khi docker chạy, chúng ta muốn 1 ứng dụng chạy trên mtruong của docker, thì docker tạo ra môi trường cách ly gọi là các container, trong các container chứa đầy đủ các thư viện, các gói ứng dựng nhằm đảm bảo ứng dụng chạy được trong docker, container là môi trường cách ly để chạy các ứng dụng và các container ko có sự tương tác trực tiếp qua lại vs nhau

Như vậy docker cài đặt vào máy host và tạo ra các container khác nhau để chạy các ứng dụng khác nhau rất giống như việc tạo ra cách máy ảo khác nhau để chạy các ứng dụng khác nhau.

**Lưu ý:** khi cài docker trên window, vào **features** của window bật **hyper -V** lên

Imager - Docker

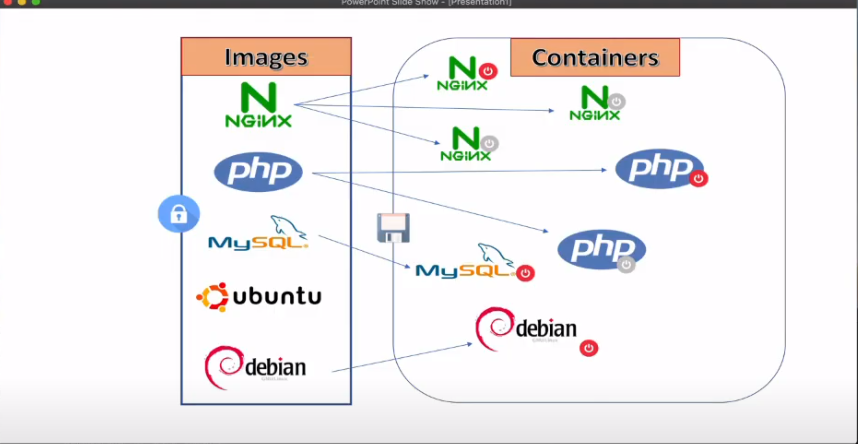


Image trong docker là những phần mềm được đóng gói và quản lý bởi docker

Vd imager đóng gói phần mềm PHP, hệ điều hành ubuntu

Trong docker các image chỉ có thể đóng, ko thể sửa đổi, khi image được docker khởi chạy thì mảng thực thi của image được gọi là các container

Khi image khởi chạy thì phiên bản thực thi của image là các container

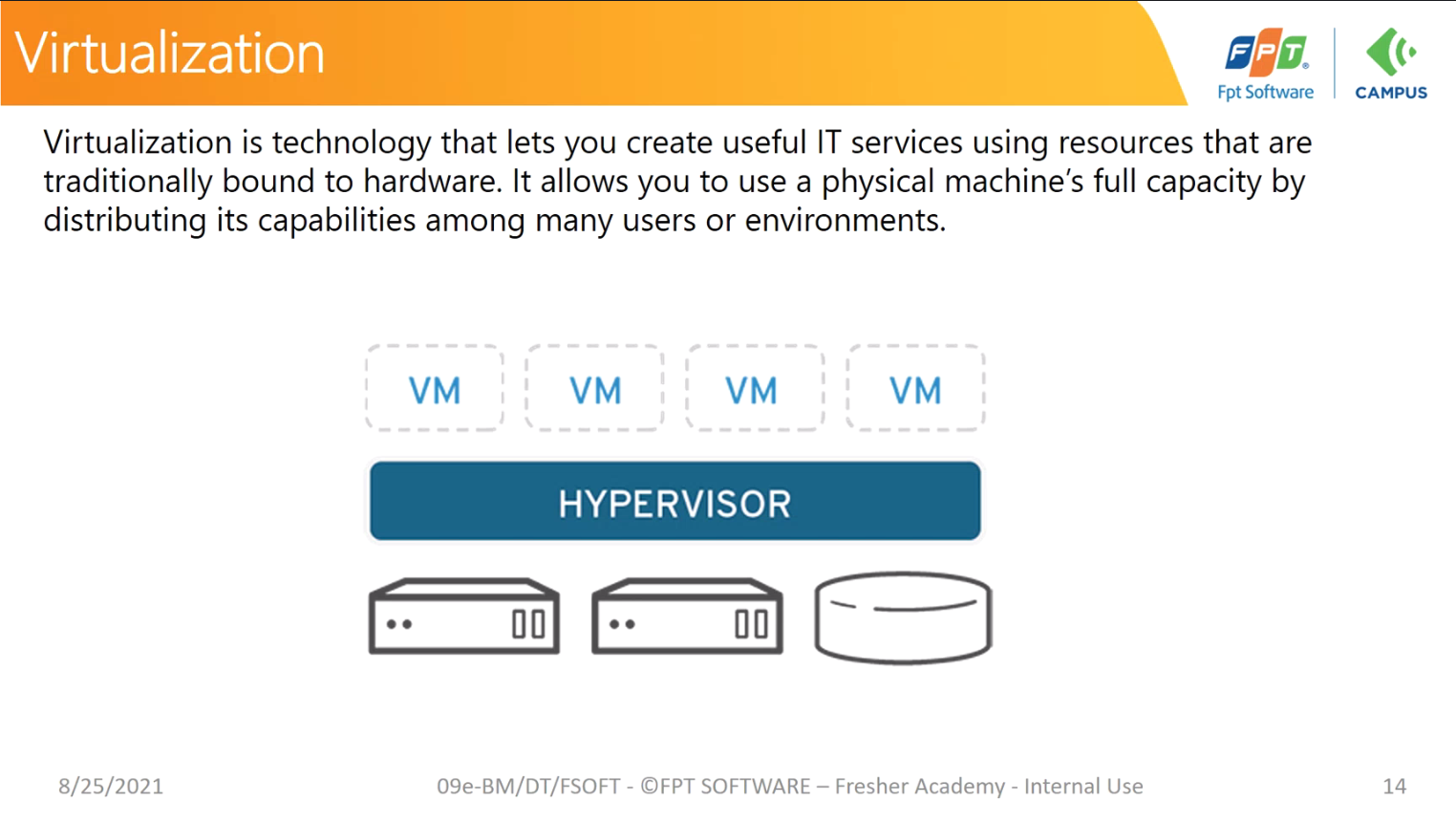
Các container có thể ghi được các dữ liệu vào trong nó

* Như vậy để có được container, qua đó chạy các ứng dụng thì chúng ta phải có các image trước

*- Docker engine*  
Docker engine là thành phần chứa các tool. Cho phép người dùng đóng gói các ứng dụng. Cũng như tạo và khởi chạy Docker container từ những Docker image.  
  
*- Docker image*  
Docker image là thành phần tương tự như file .gho dùng để ghost win. Trong một Docker image có chứa các OS và môi trường lập trình đã được cài đặt sẵn. Người dùng cũng có thể tải các image từ người khác.

Docker là phần mềm mã nguồn mở giúp chúng ta tạo ra các container để cta phát triển, triển khai ứng dụng, nói cách khác là n cho phép các ứng dụng chạy trong mtruong cách ly gọi là container. Ta tạm coi container như 1 máy ảo

**Virtualization**

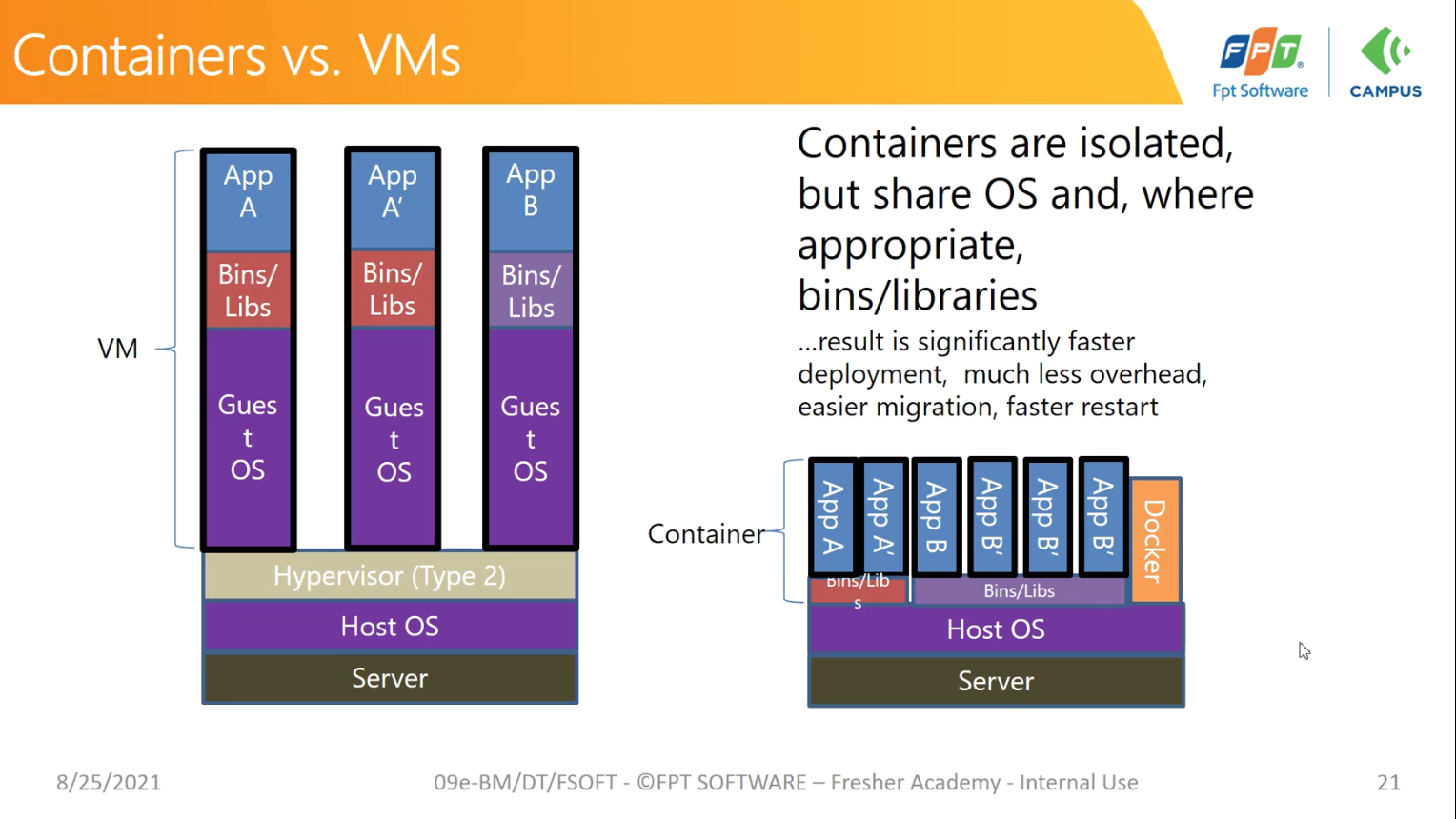


Virtualization là công nghệ được sử dụng trong lĩnh vực IT sd resource phần cứng 1 cách hợp lý nhất

Tức là sử dụng tối đa tài nguyên phần cứng vật lý ở dưới thông qua **hypervisor**

**Hypervisor:** là phần mềm, software nằm trên phần cứng vật lý, là nơi (tương tự VMware) để cho các ứng dụng ảo hóa (máy ảo) chạy nên trên phần cứng vật lý

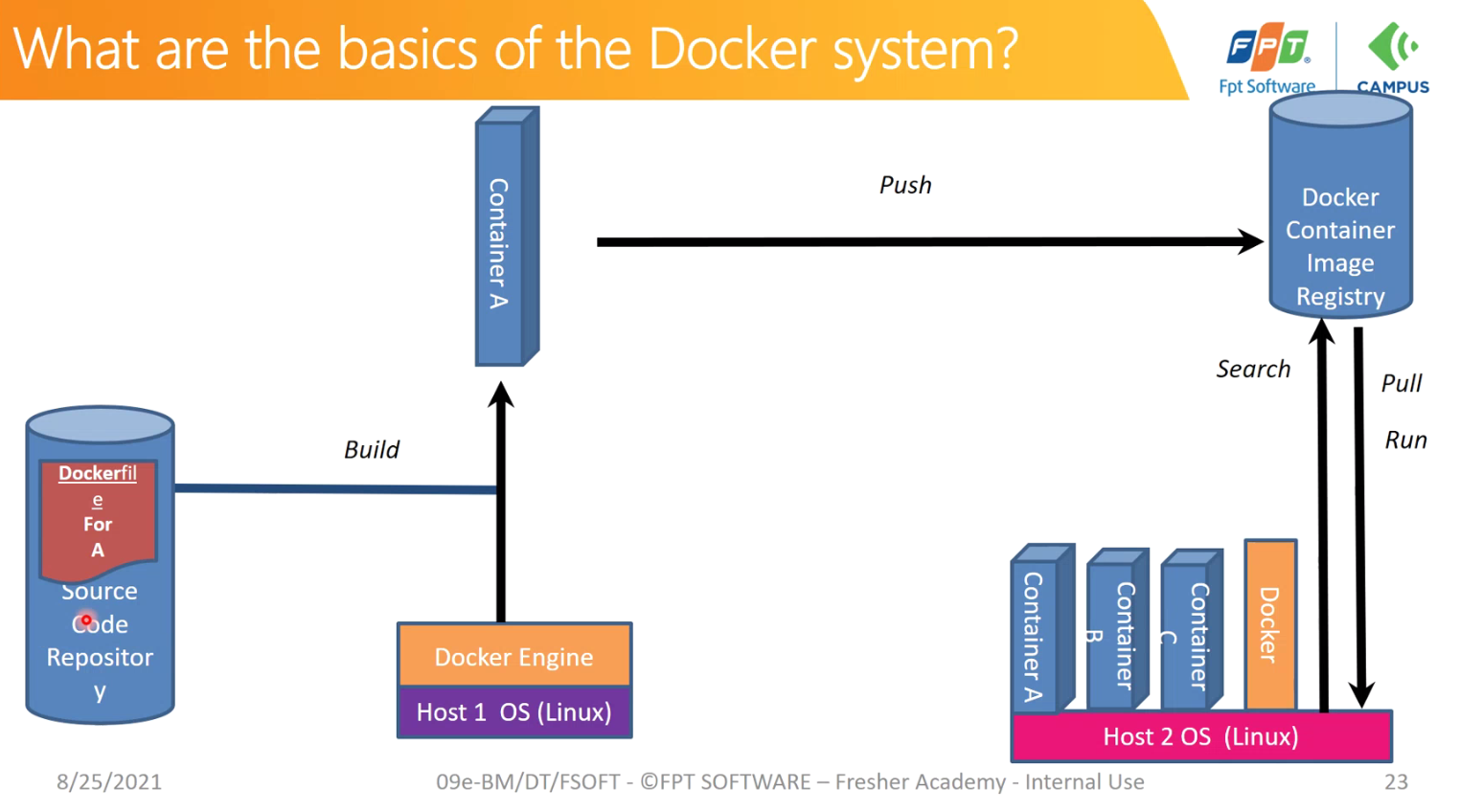
Hypervisor (Vmware, virtualbox) là môi trường chạy máy ảo, chạy hệ điều hành khác song song với máy vật lý



Khi tạo ra 1 môi trường, hay chạy 1 ứng dụng trên các OS (ubuntu, centos, linux) thì trên mỗi máy ảo vật lý chỉ tạo ra 1 OS nhất định và n sẽ chiến ram, CPU làm nặng máy

Từ đó người ra mới sinh ra khái niệm docker để đóng gói các ứng dụng, thư viện và chạy chúng trên 1 môi trường cách ly gọi là các container. Thì các container này hoàn toàn độc lập với nhau và việc triển khai rất dễ dàng mà ko chiếm quá nhiều ram, cpu, làm nặng máy như là tạo ra 1 mtruong máy ảo riêng biệt.

**Hệ thống docker**



Container image

Có 1 suorce code lưu trên docker-file

Container image là pagekage được đóng gói application, OS, library

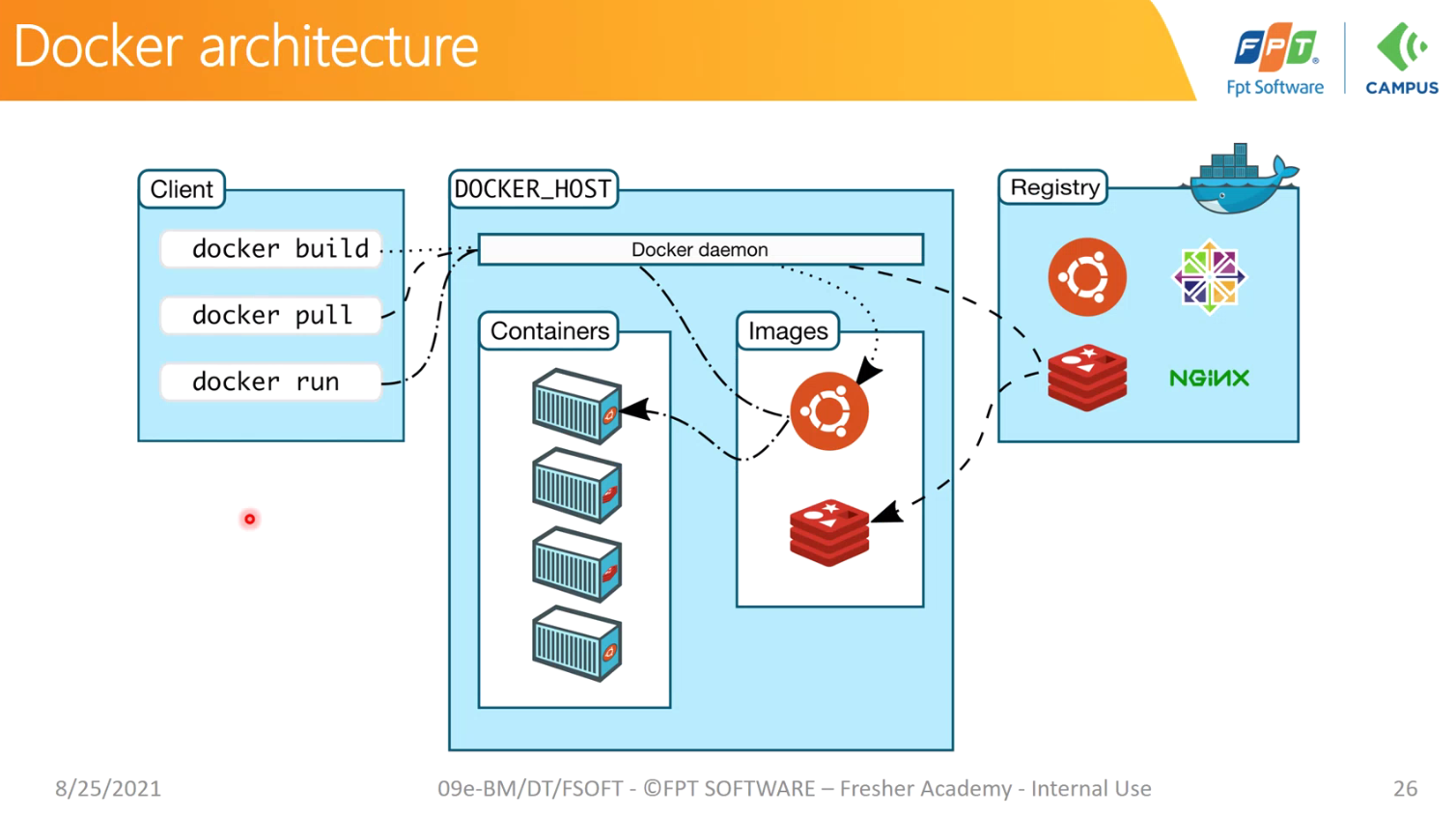
1. Build 1 container image bằng cách sử dụng docker file
2. Push container image lên docker registry

Tương tự up 1 bản AMI lên docker hub, sau đó 1 máy client có sử dụng docker có thể pull cái bản AMI, docker image này về máy (hđ tương tự như AMI aws)

1. Server chạy docker có thể tìm kiếm và pull container image về để chạy thành các container ở trên server đó

Container tương tự như VM để ta log-in vào để ta tương tác với application

**Docker Architecture**



Docker server

Cài docker trên máy client (máy ảo ubuntu) là docker engine

docker engine này gồm 2 phần

* Docker client
* Docker daemon (docker server)

**1.Docker Image**

Là thành phần bắt buộc để chạy lên các container tương ứng, docker image này đóng gói các application, thư viện

Muốn 1 container image có thể chạy 1 application python thì docker image phải đóng gói 1 app python với hđh ubuntu chẳng hạn

**2.Docker Container**

Môi trường như máy ảo để chạy các application

**3.Docker Engine**

Là container engine cho phép tạo ra

**4.Registry Service (Docker Hub (public)) or Docker Trusted Registry (Private)**

1 số cty ko muốn đưa lên docker hub vì n ở public nên họ tự tạo ra Private Registry và khai báo cho thằng docker server trỏ đến cái thằng private Registry

Registry là nơi quản lý, lưu trữ docker image