**الفقرة Containers vs VMs (محمد):**

**شريحة VMs vs. Containers**

هلأ قبل ما نحكي الفرق بين الكونتانير والفي أم وير خلينا نحكي الشي المشترك بيناتهن، عملياً التنتين ألهن نفس الهدف يلي هو عزل التطبيق أو النظام ونخليه يشتغل بوحدة مستقلة فينا نشغلها وين ما بدنا، والتنتين عملو Abstraction وريحونا من هم ال Physical Hardware وصرنا عم نشتغل بطبقة فوق الفيزيكال كلو هاد أدى إلى استخدام الموارد بكفاءة عالية.  
هلأ الفرق بيناتهن هو من الناحية المعمارية architectural.

**شريحة VMs**

VMs متل ما منعرفها هي محاكاة لكومبيوتر حقيقي، وبتشتغل من خلال شي اسمو Hypervisor وأشهر مثال عليه هو ال Vmware و Virtual Box يلي كلياتنا منستخدمهن ومنزلينهن على أجهزتنا، لما منزل Hypervisor مثل فرتشوال بوكس مثلاً على جهازنا هون بيصير اسمو Host Machine وهو بيعطي جزء من مواردو (رام، CPU) لل Hypervisoer وهي ببتولى مهمة توزيعها وادارتها بين VMs يلي منسميهن Guest Machine.

ومتل ما منعرف كل VM أو GM هي بحد ذاتها نظام تشغيل مستقل بكلشي يعني ألها System Bins و Libraries و Network Adapter الخاص فيها، فاذا بدنا نشوفها من الداخل منشوفها كوحدة مستقلة ألها مواردها الخاصة، أما من الخارج فنحنا منعرف أنو مجموعة هل GMs كلهن عم يتشاركو الموارد تبع Host Machine.

ففينا نشبه ال VM ب House متل يلي بالصورة او نحنا منسميها فيلا، فالهاوس متل ما منعرف بيكون كلشي فيها مستقل بالبنية التحتية كالتدفئة والكهرباء والمي..الخ، فكل الهاوسات بتلاقي فيها كحد أدنى غرفة نوم وغرفة معيشة وحمام ومطبخ فلما الواحد بدو يشتري هاوس على طول بيشتري شي أكثر من حاجتو بدون ما يستفيد منو.

ومتل لما بدنا هاوس جديد فنحنا مضطرين نبني واحد من الصفر كمان ال VM لما بدنا وحدة جديدة فلازم نعمل deploy لنسخة كاملة منها بما تحويها من نظام تشغيل وبرامج يعني كلشي تقريباً.

وال VM فيها هدر للموارد لأنو من أجل كل instance عندي نسخة كاملة من نظام التشغيل.

مما سبق نستنتج أنو VMs قدمتلي Hardware Virtualization.

**شريحة Containers**

الكونتاينر بتأمنلي Virtulizatoin على مستوى نظام التشغيل، يعني عندي مجموعة من الكونتاينرز بتتشارك ال Kernal تبع نظام التشغيل المضيف، ومتل مو شايفين بالصورة كل كونتاينر بتحوي فقط lib و bin يلي بتحتاجها بعلمها بحسب الأنظمة او البرامج يلي فيها وبتتشارك بنفس الوقت مع غيرها بل Kernal تبع نظام التشغيل.

ففينا نشبها بالشقة Appartment فالشقق بالبناية كلها بتتشارك بالبنية التحتية متل التدفئة المركزية، والشقق عادة بتجي بحجموم وقياسات متعددة فالواحد بيحسن يلاقي شقة على كيفو بحسب حاجتو ونفس الشي بالنسبة للكونتاينر.

**شريحة They’re different**

طيب السؤال يلي بيخطر على البال هل يمكن الدمج بين ال VM و الكونتاينرز ؟

يتبع...

**الفقرة What & Why (باسل):**

شو هو الدوكر ؟ الدكور عبارة عن مشروع اوبن سورس مكتوب بلغة Go بيعتمد كلياً على لينوكس كونتاينرز والغرض الأساسي منو هو تسهيل عملية تطوير التطبيقات والنظم داخل الكونتاينرز يلي شفنا مبدئها من شوي، وهو لحتى يشتغل بيستخدم ميزات أساسية للينكس كيرنال منها namespace و control groups لحتى يحسن يعمل كونتاينرز على نظام التشغيل العادي.

هو عبارة عن بيئة عمل خاصة بالمطورين ومديرين الانظمة تسهل عملية تطوير التطبيق ونقلة وتثبيته في اسرع وقت ممكن، كما يُمكنك من اختبار وتثبيت التطبيق في بيئة العمل الفعلية في اسرع ما يمكن، فهو يقوم بإنشاء حاويات (containers)، بحيث أن هذه الحاويات تلعب دورغلاف حاوي للتطبيق، بحيث يصبح قائما بذاته ، مكتفي ذاتياً.

والحاوية هي غلاف يستخدم لعزل مجموعة عمليات عن باقي عمليات النظام في التعامل مع الـ processor و RAM و القراءة والكتابة من ال hard disk و التعامل مع الشبكة… ، وبالتالي يكون في معزل تام عن باقي الـ processes في النظام. نفس فكرة الـsand-boxing.

الكونتاينرز مو فكرة جديدة متل ما شفنا، غوغل مثلاً كانت عم تستخدم لسنوات تكنولوجيا خاصة فيها لتحقق مبداً الكونتاينر، وفي العديد من التقينات باللينوكس بتحقق نفس المبداً منها Zones و BSD jails.

**الفقرة Containers components (آلاء):**

هلأ رح نعمل جولة سريعة عن المكونات الأساسية يلي بتشكل الـ Docker

Image: وهية المكون الأساسي بالدوكر. بالحقيقة هي الأبلكيشن كاامل مع كل المكاتب أو الأشياء اللي ممكن يستخدمها هل الأبلكيشن أو يعتمد عليها بحيث يصير يقدر هل برنامج يشتغل عأي جهاز دور الحاجة لتنصيب أي شي آخر. وهية نفس فكرة ملفات الـ iso

بتم خلق هل امج باستخدام Dockerfile

Container: هي المكان يلي بحتضن الأمج واللي بيتم تنفيذ الأمج ضمنها وبالتالي هية البيئة التنفيذية. بالحقيقة فينا نقول انو الكونتينر هو runtime instance من الأمج.من الممكن أن والتطبيق عبارة عن حاوية واحده او عدة حاويات تتواصل فيما بينها تحتوي علي كل ما يحتاجه المشروع

Engine: هو المسؤول عن خلق وإنشاء الكونتينرات بدء من إمج.

Registry Service: مهمتها التحكم بتخزين وتوزيع الأمجات.

طيب شو الفرق بين الأمج والكونتينر:

الأمج ما بيتم التعديل عليه بتاتاً، بعكس الكونتينر.

بنية الأمج بتكون بحيث كل ما منضيف فانكشناليتي جديدة علأمج بتنضاف طبقة read only جديدة وهي الطبقات غير قابلة للتعديل

أما لما منعمل new instance من container فالاضافة لبطقات الكونينر الأصلية بتنخلق طبقة read /write layer وبتنضاف على سطح طبقات الأمج الغير قابلة للتعديل. وبالتالي أي تغيير بيطرأ على الكونينر أثناء التنفيذ، مثلا عملية كتابة بملف اللوغ فبينعكس هاد الشي على طبقة ال read/write يلي انضافت مؤخراً أما الطبقات الأخرى فمتل ما حكينا ما بيصير علها أي تعديل.

وبالتالي صار فينا نشبه الأمج بال class والكونتينر بكون object أو instance من هل class

استخدام طبقات الكونتينر بيوفر كتير ميزات منها السرعة وانقاص مسحة التخرين المطلوبة، فلما منخلق كونتينر جديدة ما بيتم نسخ كامل طبقات الأمج وانما بيخلق طبقات الكونتينر وبيربطها مع طبقات الأمج الموجودة. وهذا الشي بيخلقلي سرعة كبيرة.

-ميزة كتير مهمة بالدوكر انه مو بس بيتم التشارك بنفس الصورة بين عدة أمجات وانما حتى بتم التشارك بنفس الطبقات بين عدة أمجات. فمثلاً إذا شركة بستخدم نظام تشغيل معين مع كل الامجات بالتالي بتكون طبقة نظام التشغيل مشتركة. فمتل ما عم شنفو بالصورة اذا بدنا نخلق امجين وحدة php والتانية Node.js بس التنين بيعتنمدوا عنفس نظام التشغيل بتم خلق طبقة نظام تشغيل وحدة وعليها بينبى أمجين بطقات مختلفة حسب الوظيفة يلي بدها تأديها كل أمج.

* أما عن إمكانية إنشاء عدة نسخ من الكونتيتنر فتم عن طريق ال docker compose. فببساطة منحط بال Compose file الكونفغريشنز لكل السيرفسز وبعدين بتعملية وحدة بتم خلق وإقلاع كافة الكونتينرات بدء من الملف المعرف سابقاً.
* نتيجة لإمكانية إيجاد مجموعة كبيرة من الكونتينرات ظهرت مشكلة إدارة هل كونتينرات وخاصة لما بتكون على هوست واحد.. لهيك تم إيجاد شي بيقدر يدير هل كونتينرات اسمه docker swarm من خلاله بيقدر المستخدم يعنقد أو يجمع مجموعة من العقد بالدوكر وبالتالي بتصير كسيتم افتراضي واحد. كما انه بيعمل على توزيع الور لود وبالتالي بيخففلي التعقيد.