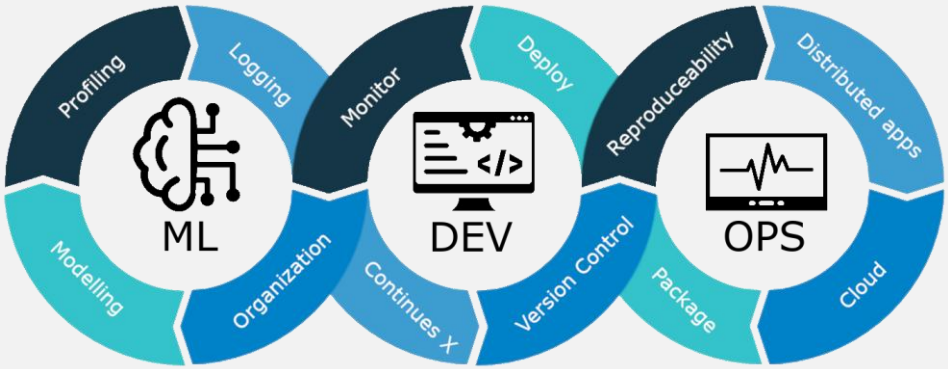
Docker

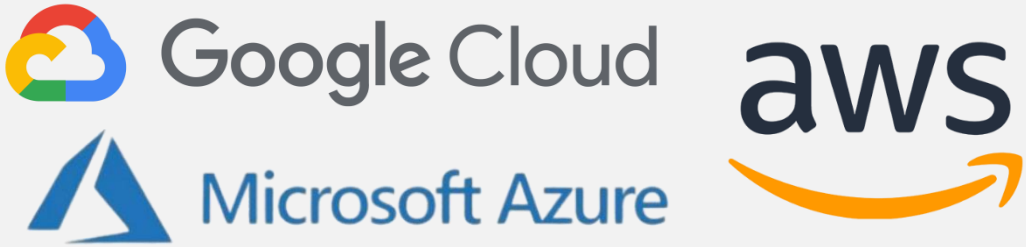
Créer et gèrer des conteneurs pour exécuter des applications de manière isolée et portable.

UP ASI
Département informatique
Bureau: E204



La virtualisation

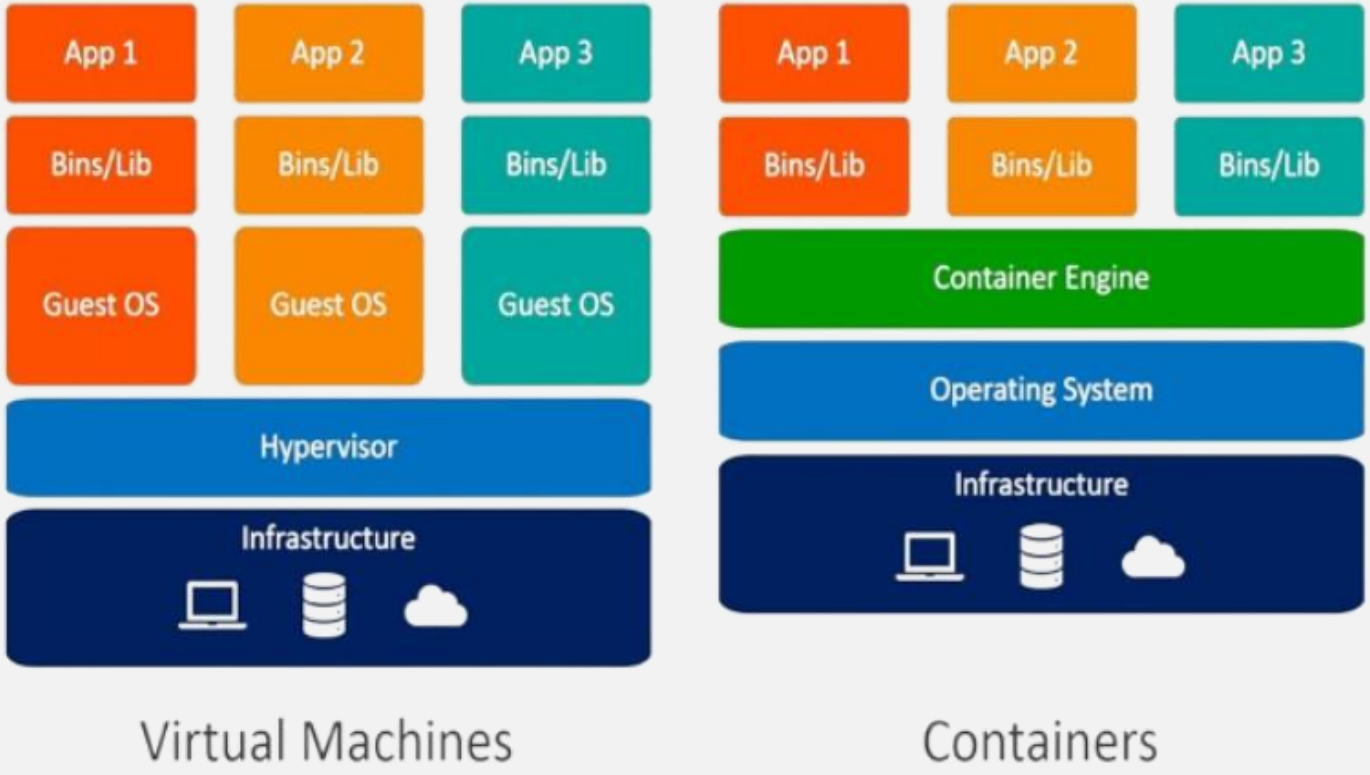
- ✓ La virtualisation est une technologie permettant de créer et d'exécuter une ou plusieurs représentations virtuelles d'un ordinateur et de ses différentes ressources sur une même machine physique.
- ✓ La virtualisation a eu le succès grâce au **cloud computing**: C'est un Data center ou une infrastructure offerte par un fournisseur dans laquelle la puissance de calcul et le stockage sont gérés par des serveurs distants auxquels les usagers se connectent via une liaison Internet sécurisée.
  - ✓ Elasticité rapide
  - ✓ Modèle Pay as You Go



Les Data Center

- La capacité de stockage des data Center est de 175 zettaoctets en 2025. Pour mémoire, un zettaoctet équivaut à mille milliards de milliards d'octets, soit un milliard de disques durs d'un téraoctet
- Dans ces données, une partie que l'on estime de 60 % à 80 % est dite 'froide' : données que l'on utilise rarement, voire jamais, comme par exemple les documents légaux ou des vieux emails.
- Certains dataCenter présentent un bilan carbone équivalent à celui d'une ville entière.

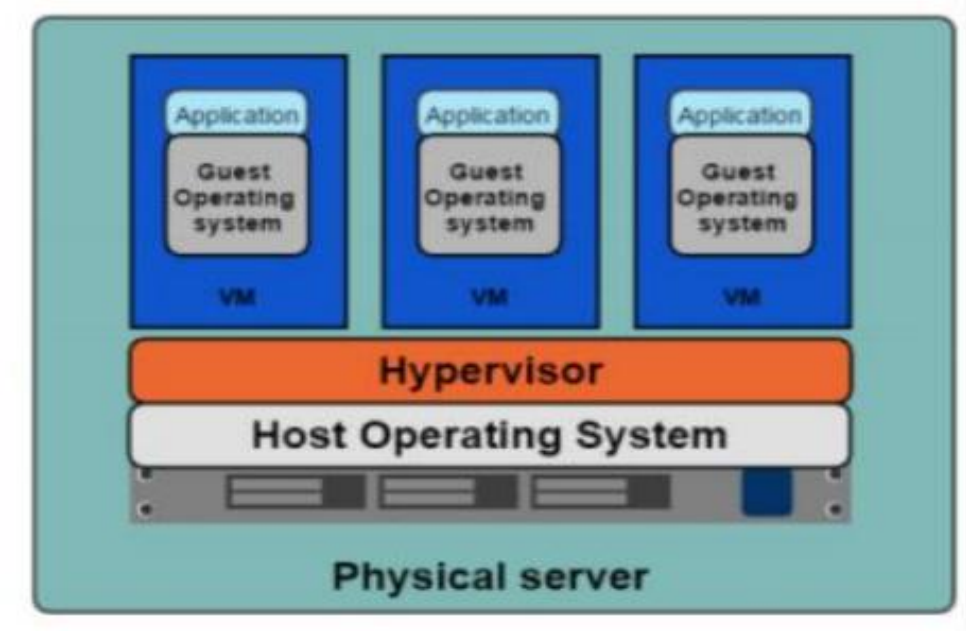
La virtualisation



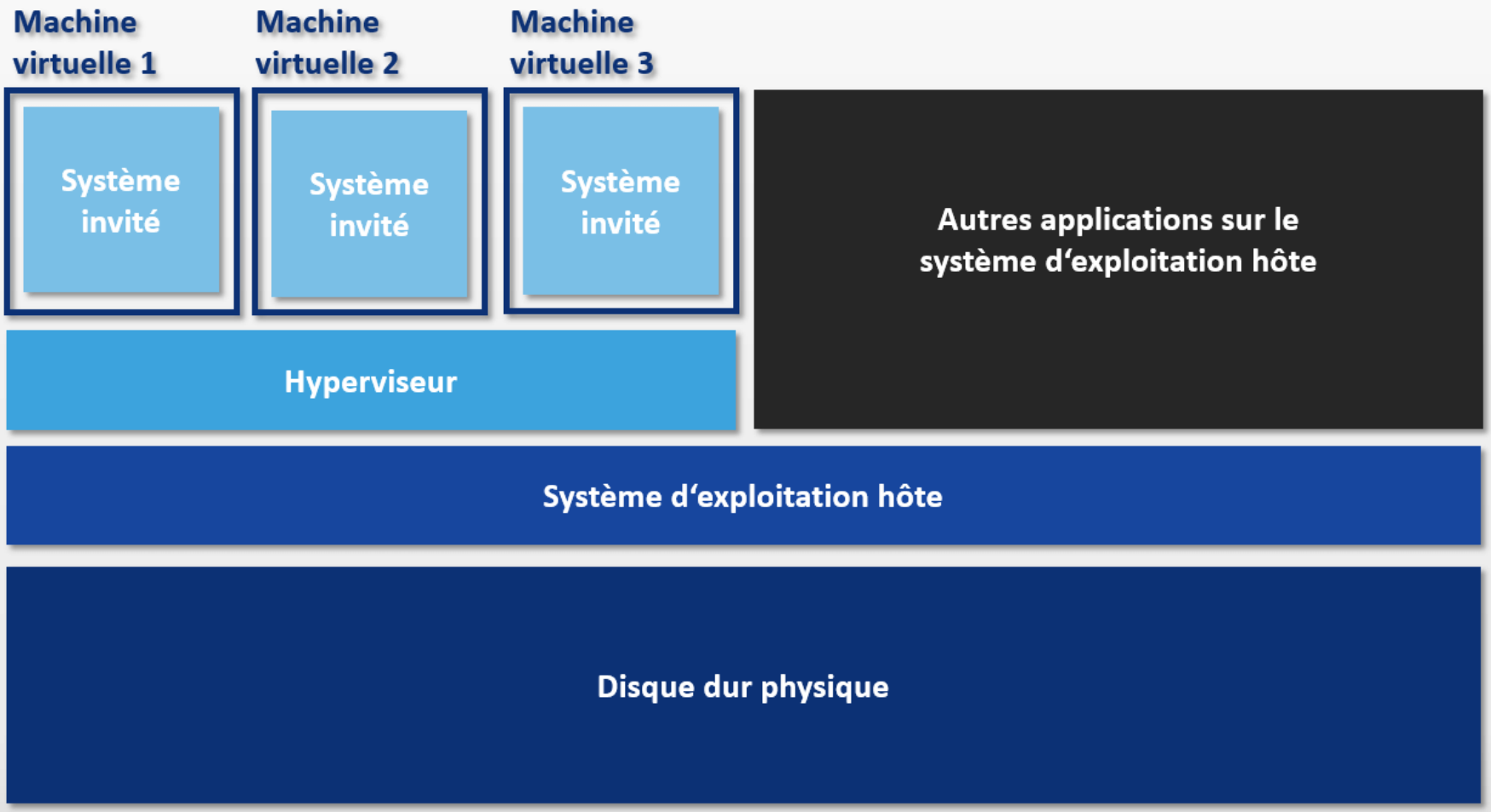
- On distingue deux types de virtualisation tels que:
  - ✓ La virtualisation lourde (VM)
  - ✓ La virtualisation légère (Conteneurs)

La virtualisation lourde

- **La virtualisation Lourde ou à base de VM (utilisant un Hyperviseur):** Elle permet de simuler une ou plusieurs machines physiques, et les exécuter sous forme de machines virtuelles (VM) sur un serveur. Ces VM intègrent un **OS** sur lequel des applications sont exécutées.



La virtualisation lourde



La virtualisation lourde

- **Avantages des VMs**
  - ✓ Une meilleure exploitation des ressources
  - ✓ Une machine physique est divisée en plusieurs machines virtuelles
- **Limites des VMs**
  - ✓ Chaque VM a besoin des ressources (CPU, Stockage (Disque), RAM, OS invité)
  - ✓ En augmentant les VMs, on demande plus de ressources
    - Chaque OS invité alloue ses propres ressources
    - Gaspillage
    - Portabilité d'applications n'est pas garantie
- **Outils de virtualisation**

La virtualisation légère

La virtualisation légère ou à base des conteneurs

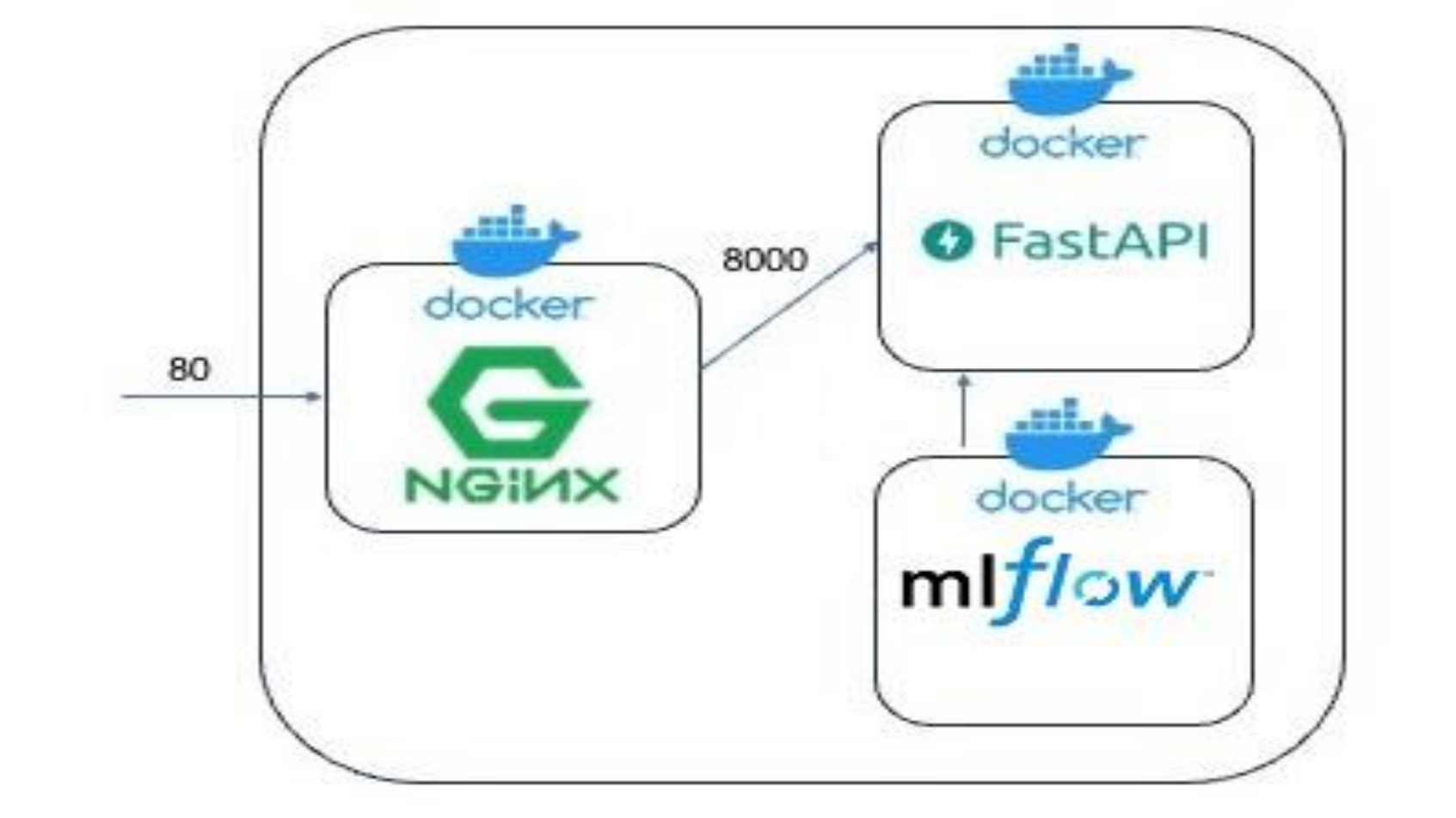
La virtualisation légère

La virtualisation légère ou à base des conteneurs:

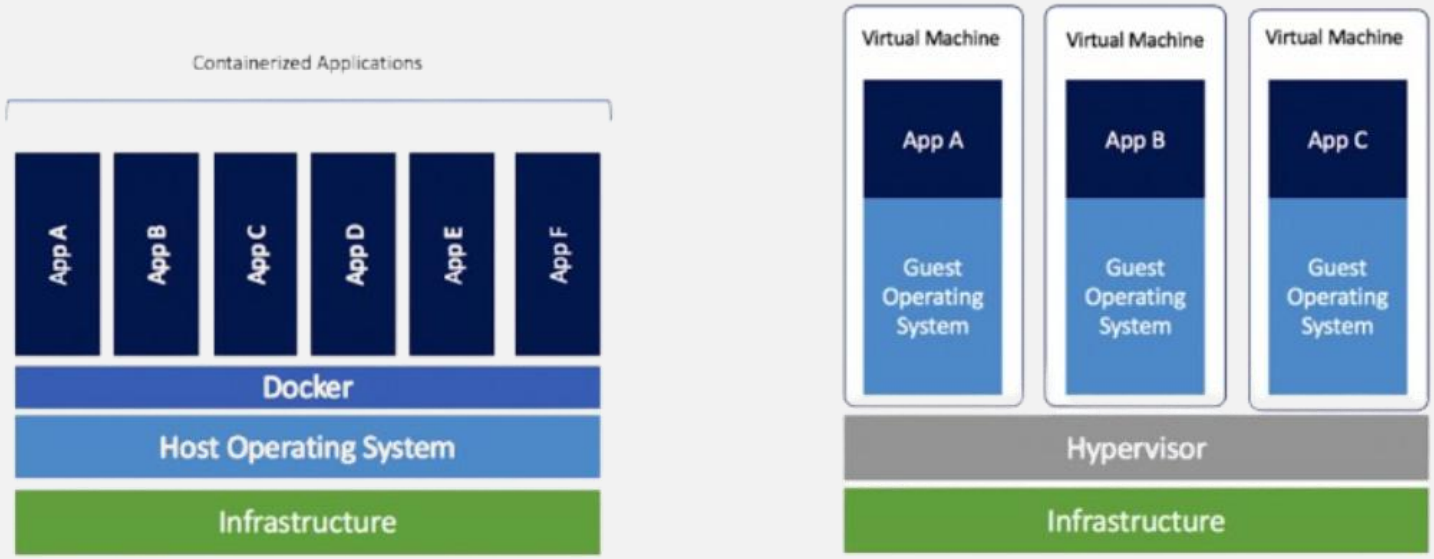
- Offre tous les services, scripts, API, librairies dont une application a besoin favorisant ainsi une juste utilisation des ressources.
- Repose sur la création de conteneurs isolés les uns des autres sur un **OS Commun**.
- Les conteneurs indépendants partagent un **OS commun** et un même **espace mémoire**.

- **Simplifiant les choses:** Un conteneur est une enveloppe (emballage) contenant toutes les ressources nécessaires pour faire fonctionner une application donnée (Environnement d'exécution comme JDK, livrable de l'application, dépendances nécessaires)

La virtualisation lourde vs. légère

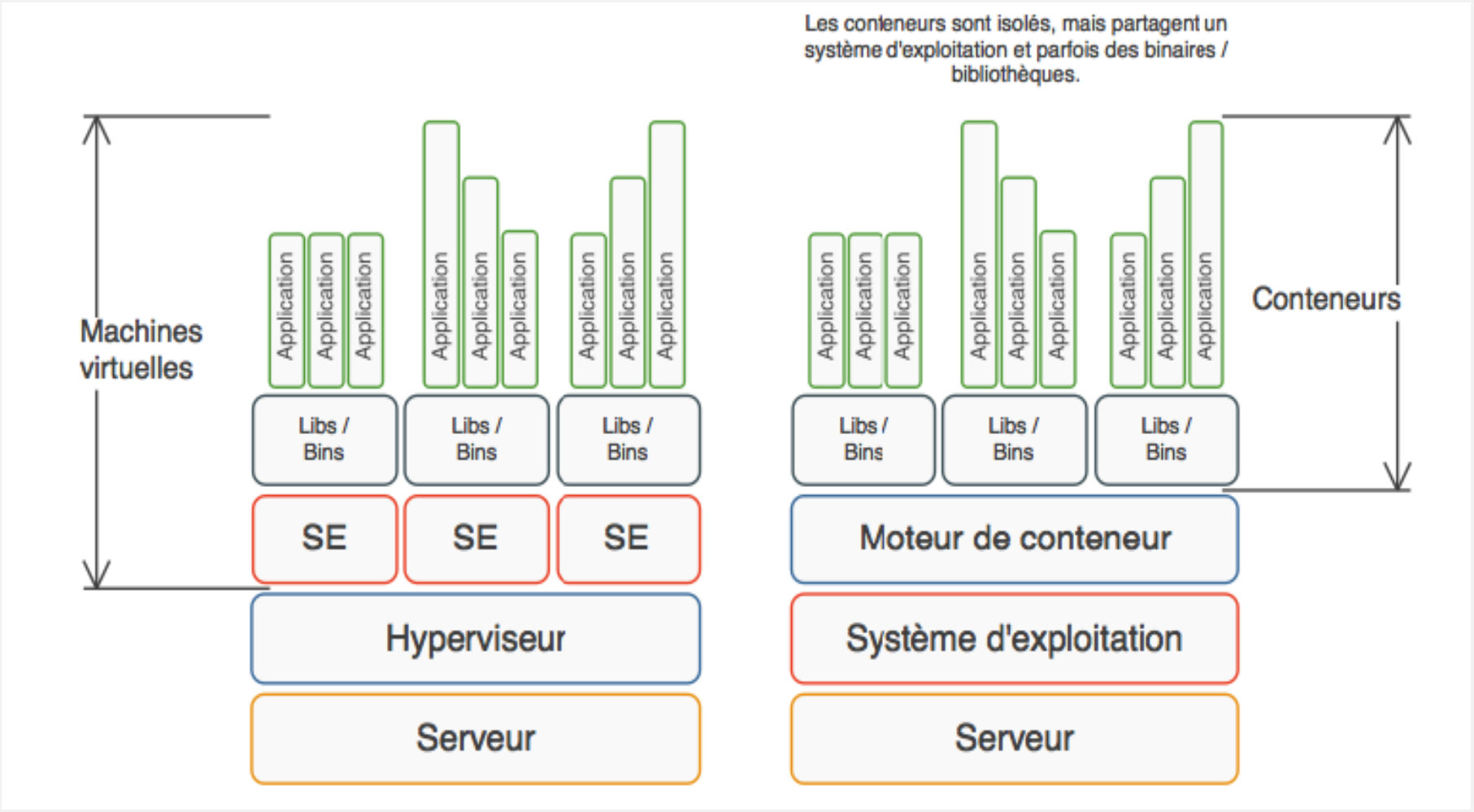


La virtualisation lourde vs. légère



- Une machine virtuelle va recréer un serveur complet pour chaque application avec son propre système d'exploitation
- Le conteneur va isoler l'application tout en utilisant le système d'exploitation de son hôte. Ce même système d'exploitation peut être utilisé par d'autres conteneurs ayant des taches totalement distinctes.

La virtualisation lourde vs. légère

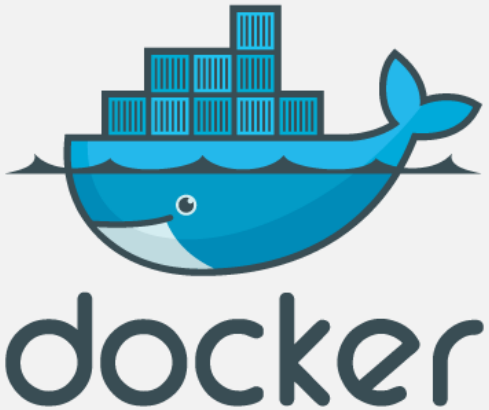


La virtualisation lourde vs. légère

VM	Container
Une virtualisation lourde	Une virtualisation légère
À base de VM	À base des Conteneurs
Virtualisation des ressources **hardware** (CPU, RAM, disque, …)	Virtualisation au niveau du système d'exploitation (de l'**OS**)
Machine invité (machine virtuelle)	Conteneur
Image ISO (OS Complet)	Librairies nécessaires
Hyperviseur sur la Machine hôte	Moteur de conteneur sur la machine hôte
Démarrage plus lent	Démarrage en quelques secondes
Entièrement isolé et donc plus sécurisé	Isolation au niveau du processus et donc moins sécurisée

Docker

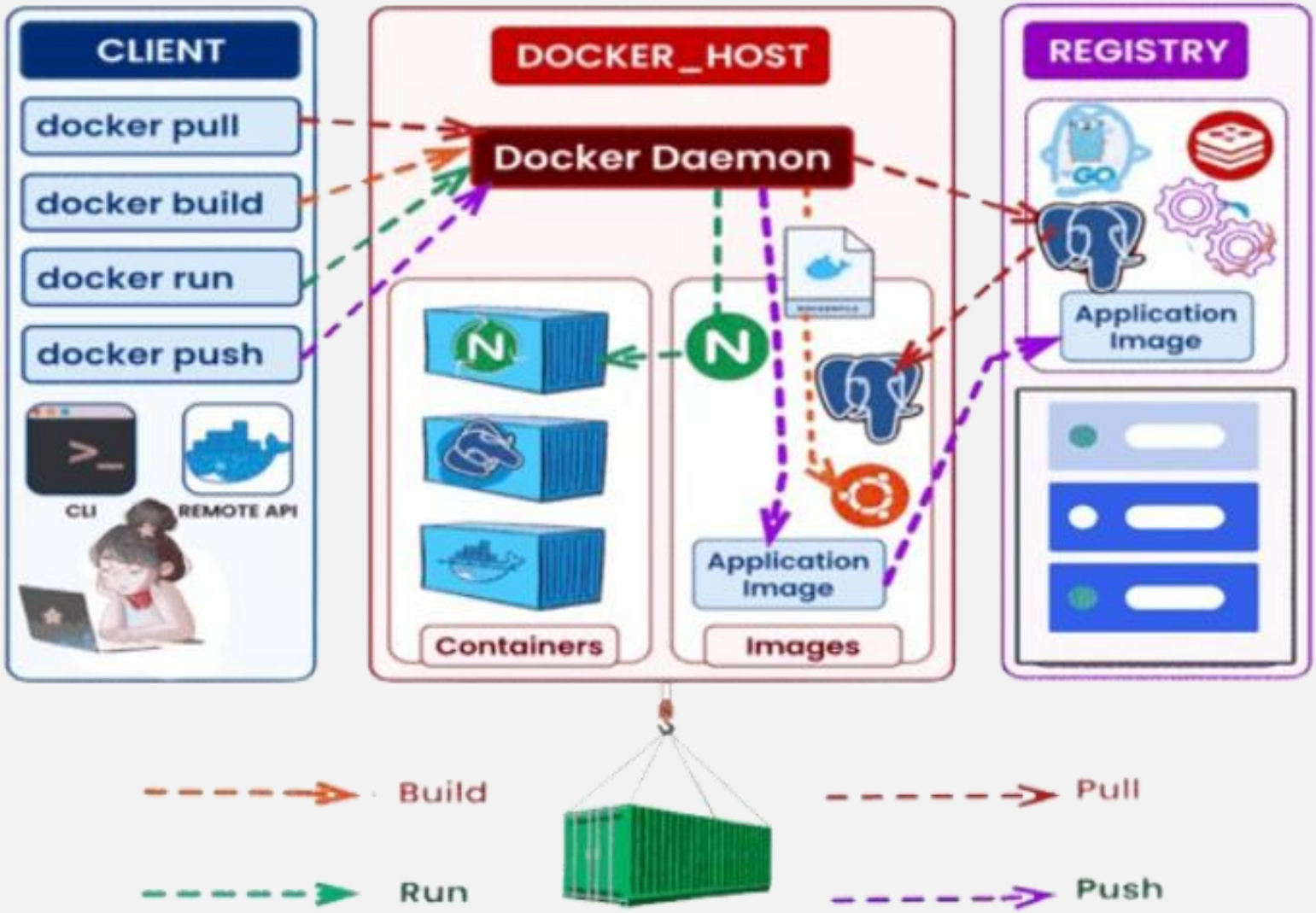
- Docker est une plateforme permettant de développer, expédier et exécuter des applications dans des conteneurs légers.
- Pourquoi Docker ?
  - → Rapidité.
  - → Isolation des applications.
  - → Facilité de déploiement.
- On distingue deux versions de docker:
  - Docker **E**ntreprise **E**dition : Docker EE payante
  - Docker **C**ommunity **E**dition: Docker CE gratuite



Docker

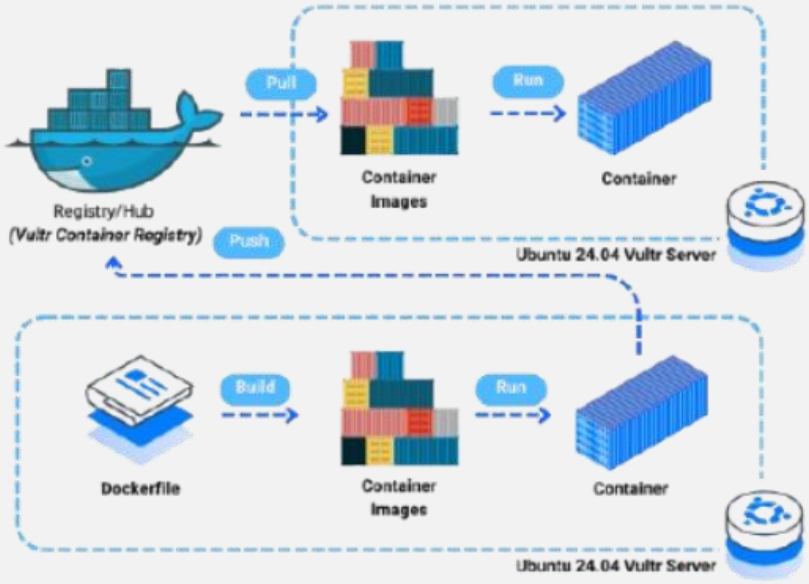
- **Docker** a connu une adoption massive dans le monde de l'informatique. Les données suivantes illustrent la popularité et l'adoption de Docker :
- Utilisation répandue : Selon le Docker Hub, plus de **12 millions de développeurs** utilisent activement Docker pour gérer leurs conteneurs.
- Adoption en entreprise : **89 %** des grandes entreprises utilisent Docker dans leurs processus de développement et de déploiement, selon une étude de l'entreprise Docker.
- Popularité sur GitHub : Docker compte plus de **60 000 étoiles sur GitHub**, ce qui en fait l'un des projets open source les plus populaires.
- Intégration dans les CI/CD : Plus de **75 %** des organisations qui adoptent des pipelines CI/CD utilisent Docker pour la conteneurisation de leurs applications.

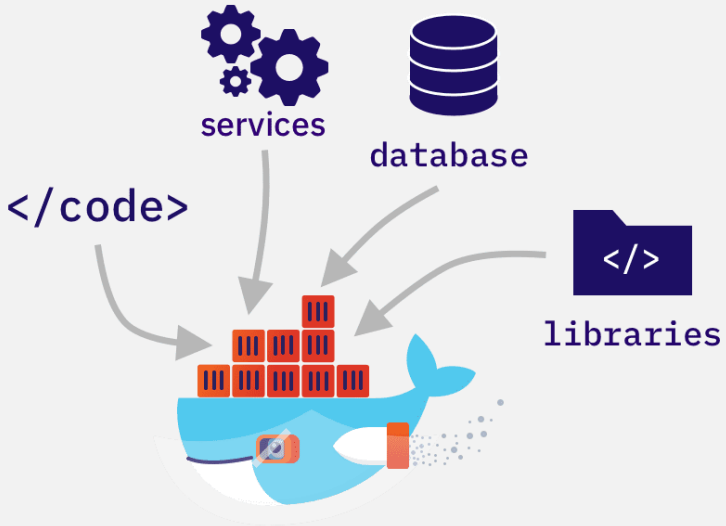
Docker - Architecture



Docker - Architecture

- Docker Engine : Le moteur principal pour exécuter Docker.
- Images Docker : Modèles pour créer des conteneurs.
- Conteneurs Docker : Instances exécutables des images.
- Docker Hub : Registre public pour partager les images Docker.



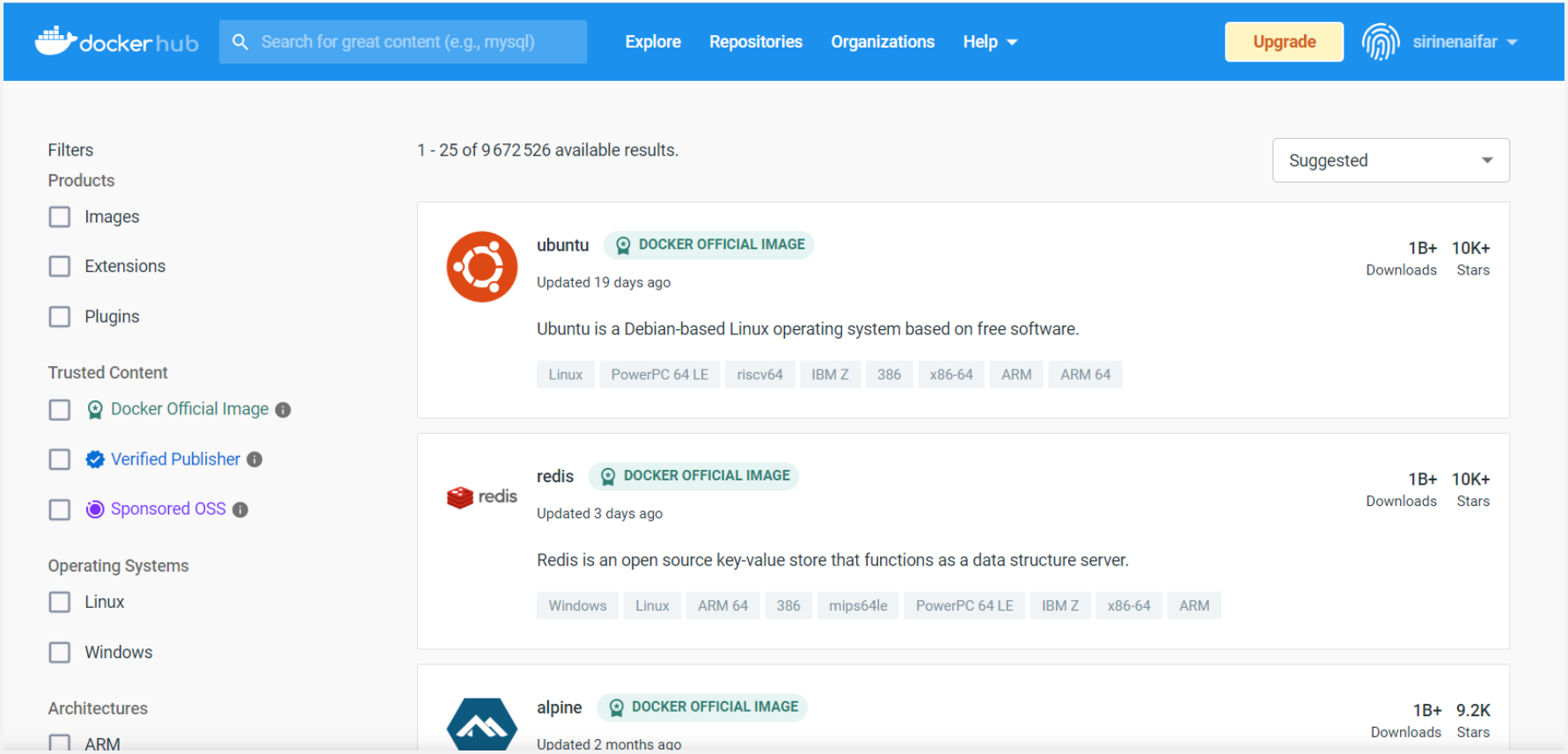


Docker – Docker Hub

- DockerHub est le registre officiel de Docker
- C'est un répertoire **SaaS** (Software as a Service ou logiciel en tant que service) regroupant des applications containerisés (images) fournis par des développeurs/opérationnels et accessibles.
- Ces images peuvent également être fournies par Docker.
- Il est possible de télécharger ces images et de partager les vôtres.

Docker – Docker Hub

https://hub.docker.com/



Docker - Image et conteneur

- Une **image** est un modèle figé contenant tous les fichiers nécessaires pour exécuter une application (code, dépendances, configuration).
- Un **conteneur** est une instance d'une image en cours d'exécution, un environnement actif basé sur cette image.

→ Le conteneur est une instance d'une image exécutée

- **Analogie simple :**
  - Une image est comme une recette de cuisine (instructions, ingrédients).
  - Un conteneur est le plat préparé à partir de cette recette.

→ Vous pouvez exécuter plusieurs plats (conteneurs) à partir de la même recette (image).

Docker - Image

→ Comment créer et utiliser une image Docker ?

- Une image Docker est construite à partir d'un fichier de configuration appelé **Dockerfile**. Ce fichier contient les instructions nécessaires pour configurer l'environnement de l'application.
- Il est basé sur une image standard auquel on ajoute les éléments propres à l'application que l'on veut déployer.

Docker - Image

```
FROM python:3.9.1

RUN apt-get install wget

COPY ./requirements.txt ./
RUN pip install -r requirements.txt
RUN pip install psycopg2 pyarrow fastparquet

WORKDIR /app
COPY ingest_data.py ingest_data.py

ENTRYPOINT [ "python", "ingest_data.py" ]
```

Docker - Image

→ Comment créer et utiliser une image ?

Les instructions de base que peut contenir un DockerFile sont les suivantes :

- **FROM** permet de définir depuis quelle base votre image va être créée
- **LABEL** permet de définir l'auteur de l'image
- **RUN** permet de lancer une commande
- **ADD** permet de copier un fichier depuis la machine hôte ou depuis une URL
- **EXPOSE** permet d'exposer un port du container vers l'extérieur
- **CMD** détermine la commande qui sera exécutée lorsque le container démarrera
- **ENTRYPOINT** permet d'ajouter une commande qui sera exécutée par défaut
- **WORKDIR** permet de définir le dossier de travail pour toutes les autres commandes
- **ENV** permet de définir des variables d'environnements

Docker - Commandes

- Pour extraire une image ou un référentiel d'un registre:

  docker pull « nom de l'image » : « version »

  → Si on ne spécifie rien, docker téléchargera la dernière version

```
[root@adsl-172-10-0-35 docker]# docker pull alpine
Using default tag: latest
Trying to pull repository docker.io/library/alpine ...
latest: Pulling from docker.io/library/alpine
Digest: sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad
Status: Image is up to date for docker.io/alpine:latest
```

```
[root@adsl-172-10-0-35 docker]# docker pull nginx:1.15.12
Trying to pull repository docker.io/library/nginx ...
1.15.12: Pulling from docker.io/library/nginx
743f2d6c1f65: Pulling fs layer
6bfc4ec4420a: Pulling fs layer
688a776db95f: Pulling fs layer
```

Docker - Commandes

- Pour construire une image docker définie dans un fichier Dockerfile , vous devez exécuter la commande:

docker build -t « username dockerHub » / « Nom de l'image à créé »

« Path vers le fichier »

- Vous pouvez également spécifier .(point) lorsque vous utilisez le fichier docker à partir du répertoire actuel, mais s'il se trouve dans un autre répertoire, vous devez spécifier le chemin complet.
- De plus, si vous n'utilisez pas le nom de dockerfile par défaut, l'option -f est requise

Docker - Commandes

```
[root@localhost docker]# docker build -t sirinenaifar/alpine:1.0.0 .
Sending build context to Docker daemon 2.048 kB
Step 1/4 : FROM alpine
Trying to pull repository docker.io/library/alpine ...
latest: Pulling from docker.io/library/alpine
213ec9aee27d: Pull complete
Digest: sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad
Status: Downloaded newer image for docker.io/alpine:latest
 ---> 9c6f07244728
Step 2/4 : RUN apk add openjdk11
 ---> Running in 7fd26d1641c7

fetch https://dl-cdn.alpinelinux.org/alpine/v3.16/main/x86_64/APKINDEX.tar.gz
fetch https://dl-cdn.alpinelinux.org/alpine/v3.16/community/x86_64/APKINDEX.tar.gz
(1/30) Installing java-common (0.5-r0)
(2/30) Installing libffi (3.4.2-r1)
(3/30) Installing p11-kit (0.24.1-r0)
(4/30) Installing libtasn1 (4.18.0-r0)
(5/30) Installing p11-kit-trust (0.24.1-r0)
(6/30) Installing ca-certificates (20220614-r0)
(7/30) Installing java-cacerts (1.0-r1)
(8/30) Installing openjdk11-jre-headless (11.0.16.1_p1-r0)
```

Docker - Commandes

```
[root@localhost docker]# docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
sirinenaifar/alpine     1.0.0               698f1a073864        18 minutes ago      275 MB
docker.io/alpine        latest              9c6f07244728        6 weeks ago         5.54 MB
```

- L'image «sirinenaifar/alpine » est local (Accessible et utilisé par le créateur seulement). Pour rendre cette image accessible pour tous les utilisateurs de l'outil Docker, il faut la déposer dans le Docker Hub.
- La première chose à faire est de s'authentifier à partir du terminal: docker login –username=userName

```
[root@adsl-172-10-0-35 docker]# docker login --username sirinenaifar
Password:
Login Succeeded
```

Docker - Commandes

- Récupérer la liste des images:

docker images

```
[root@adsl-172-10-0-35 docker]# docker images
REPOSITORY                  TAG                 IMAGE ID            CREATED
        SIZE
docker.io/alpine            latest              9c6f07244728        3 weeks ago
        5.54 MB
docker.io/hello-world       latest              feb5d9fea6a5        11 months ag
o       13.3 kB
docker.io/centos            8                   5d0da3dc9764        11 months ag
o       231 MB
docker.io/centos            latest              5d0da3dc9764        11 months ag
o       231 MB
docker.io/nginx             1.15.12             53f3fd8007f7        3 years ago
        109 MB
```

Docker - Commandes

- Créer un conteneur:

docker run « nom de l'image »

➔ Si docker ne trouve pas l'image, il la télécharge et après il crée le conteneur

```
[root@adsl-172-10-0-35 docker]# docker run centos
Unable to find image 'centos:latest' locally
Trying to pull repository docker.io/library/centos ...
latest: Pulling from docker.io/library/centos
Digest: sha256:a27fd8080b517143cbbbab9dfb7c8571c40d67d534bbdee55bd6c473f432b
177
Status: Downloaded newer image for docker.io/centos:latest
```

Docker - Commandes

Pour faire un tag d'une image docker, il faut utiliser la commande suivante :

docker tag id_image docker_hub repository_name/version

```
PS D:\devOps projects> docker tag feb5d9fea6a5 helloworld:version1.0
PS D:\devOps projects>
```

Nous pouvons faire le tag directement sur le nom de l'image :

docker tag nom_image docker_hub repository_name/version

```
PS D:\devOps projects> docker tag hello-world helloworld:version1.0
PS D:\devOps projects>
```

Docker - Commandes

- Récupérer la liste de tous les conteneurs:

  docker container ls ou bien docker ps –a

```
[root@adsl-172-10-0-35 docker]# docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS                        PORTS               NAMES
a15af1503a9d        centos              "/bin/bash"              41 minutes ago      Exited (0) 41 minutes ago                         hopeful_galileo
4f97a1e0e6fa        9c6f07244728        "/bin/sh -c 'yum i..."   About an hour ago   Exited (127) About an hour ago                    reverent_wilson
d556c59a5432        9c6f07244728        "/bin/sh -c 'yum i..."   2 hours ago         Exited (127) 2 hours ago                          jovial_tesla
ab90c4bfc1a6        5d0da3dc9764        "/bin/sh -c 'yum i..."   2 hours ago         Exited (1) 2 hours ago                            serene_keller
3e96099e1f5d        5d0da3dc9764        "/bin/sh -c 'yum u..."   2 hours ago         Exited (1) 2 hours ago                            flamboyant_turing
a1ef5bace358        5d0da3dc9764        "/bin/sh -c 'dnf u..."   2 hours ago         Exited (1) 2 hours ago                            dreamy_dubinsky
dfcc9ee52765        5d0da3dc9764        "/bin/sh -c 'dnf u..."   2 hours ago         Exited (1) 2 hours ago                            gifted_hamilton
971590f59e86        5d0da3dc9764        "/bin/sh -c 'yum u..."   2 hours ago         Exited (1) 2 hours ago                            mystifying_knuth
eadec32d97ef        5d0da3dc9764        "/bin/sh -c 'yum u..."   2 hours ago         Exited (1) 2 hours ago                            sleepy_einstein
b29d66b74ba7        5d0da3dc9764        "/bin/sh -c 'sudo ..."   2 hours ago         Exited (127) 2 hours ago                          jovial_shirleye87a
c97c75e1       5d0da3dc9764       "/bin/sh -c 'dnf u..."   3 hours ago         Exited (1) 3 hours ago                            fervent_bassi
```

- Récupérer la liste des conteneurs actifs :

  docker ps

Docker - Commandes

- Supprimer une image:

docker image rm «nom de l'image » - - force

ou bien docker rmi « nom de l'image »

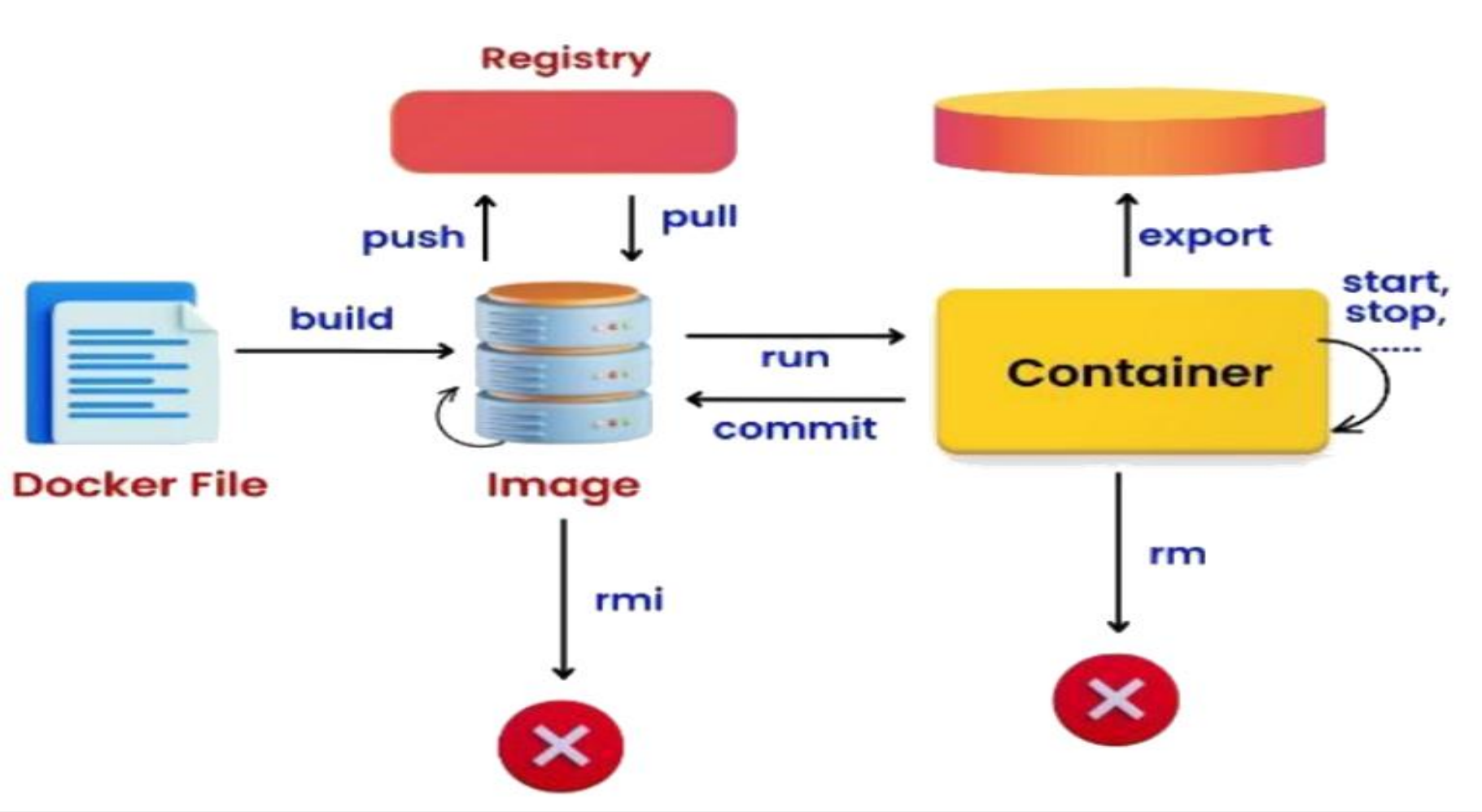
```
[root@adsl-172-10-0-35 docker]# docker image rm hello-world --force
Untagged: hello-world:latest
Untagged: docker.io/hello-world@sha256:7d246653d0511db2a6b2e0436cfd0e52ac8c066000264b3ce63331ac66dca625
Deleted: sha256:feb5d9fea6a5e9606aa995e879d862b825965ba48de054caab5ef356dc6b3412
```

- Supprimer un conteneur:

docker rm « Id du conteneur »

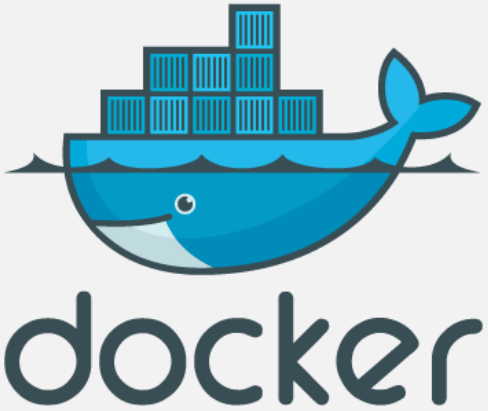
```
[root@adsl-172-10-0-35 docker]# docker rm 6b3337199055
6b3337199055
```

Docker - Commandes



Docker – Travail à faire

- Suivre les étapes de votre workshop pour conteneuriser votre projet (voir Atelier 6)



Docker

Si vous avez des questions, n'hésitez pas à nous contacter :

Département Informatique

UP Architectures des Systèmes d'Information

Bureau E204 /E304