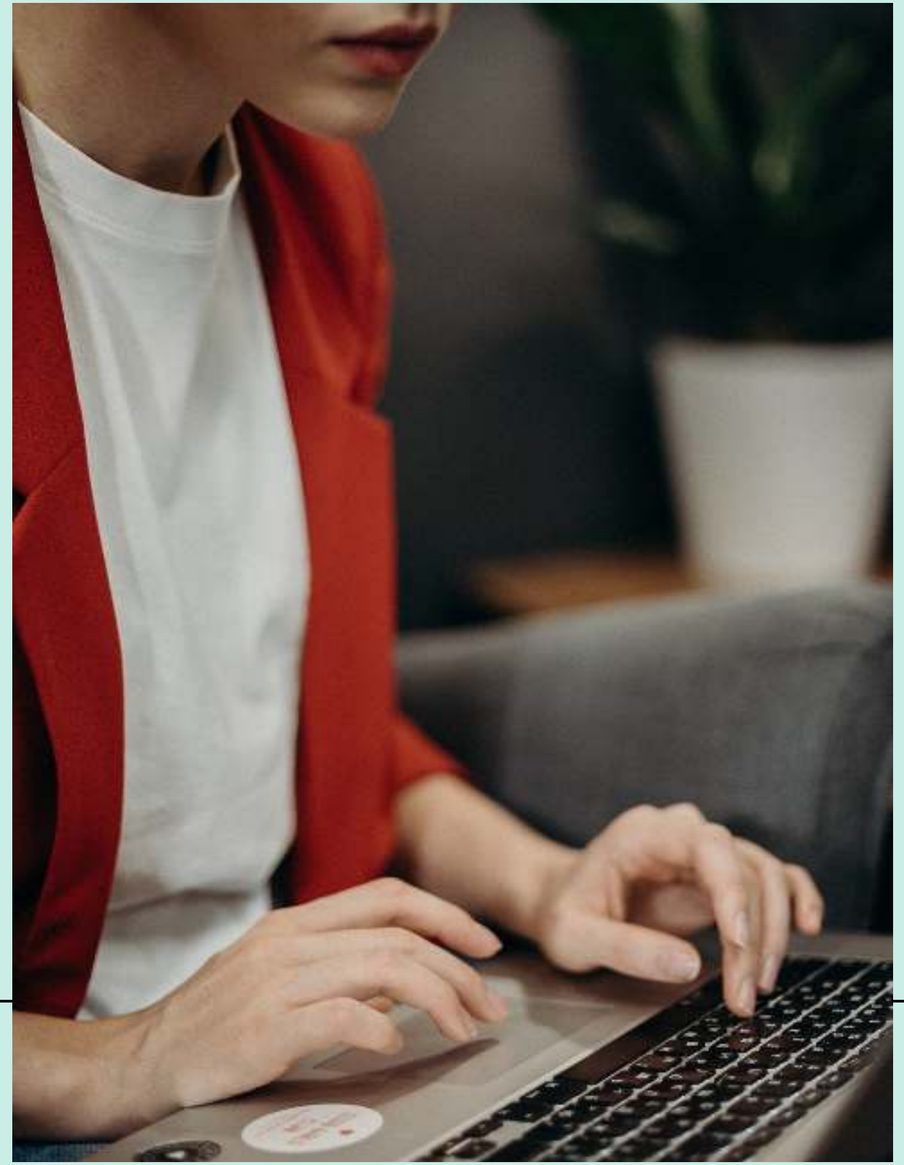
TD #2

Voir l’énoncé pdf



DOCKER TD#1

préparation

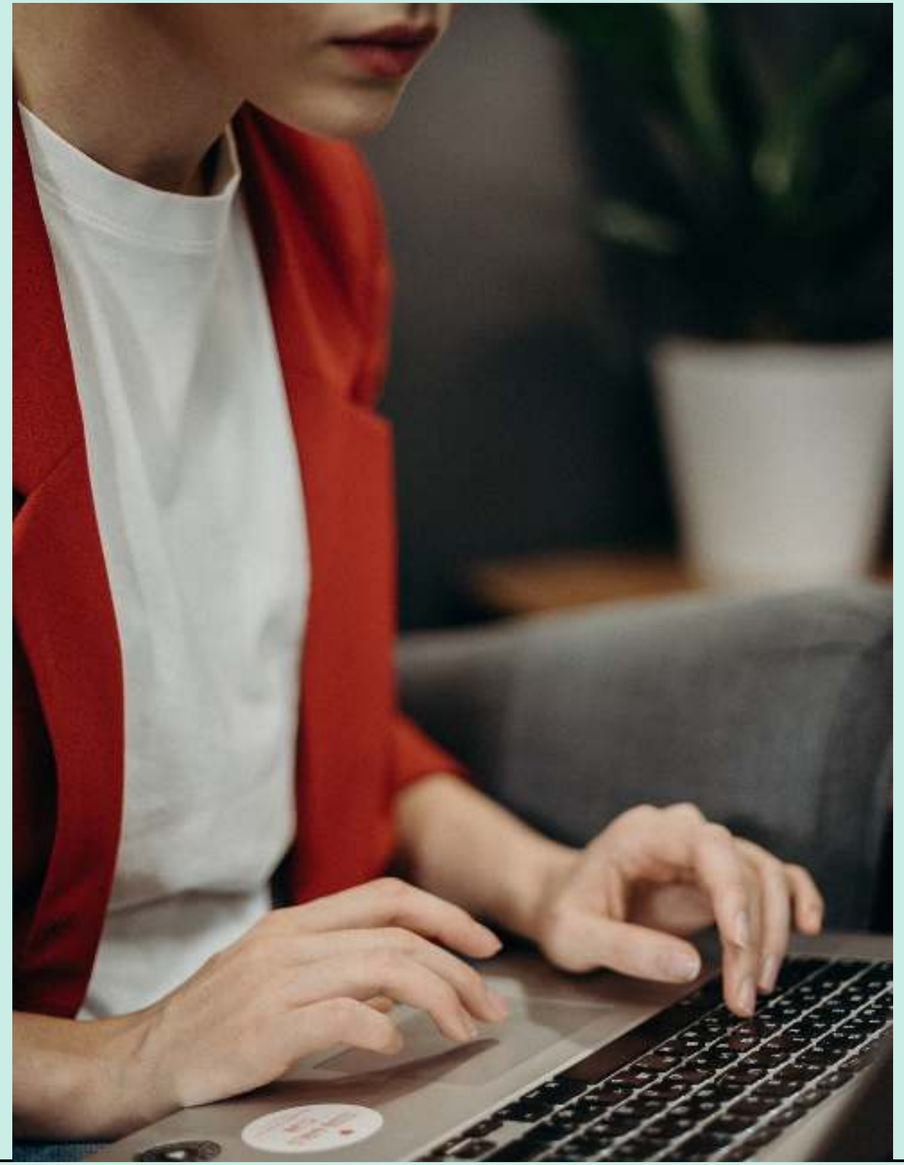
VM LINUX DEBIAN

- simuler un serveur de production
- simuler un serveur distant
- simuler un serveur dédié pour les runners

- avoir un environnement de travail commun

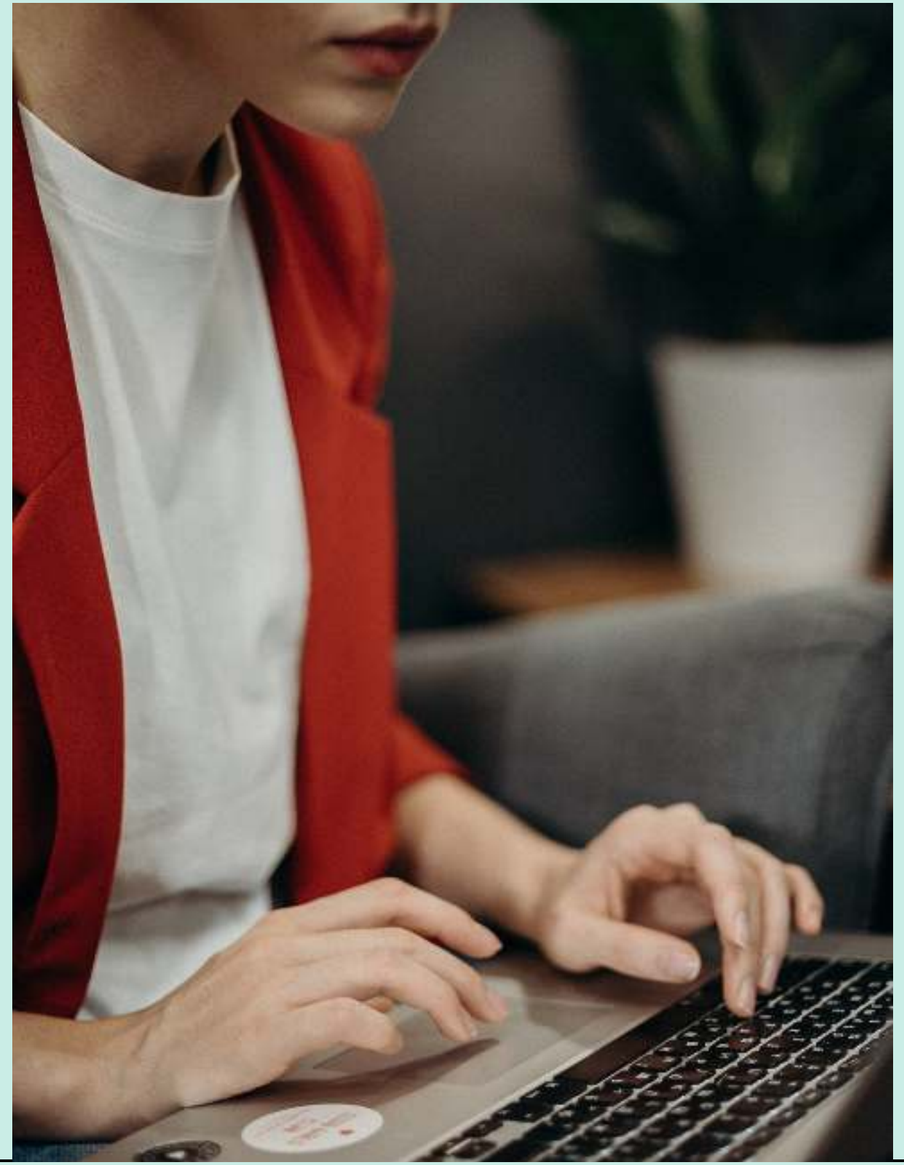
INSTALLATION VM

- installer VirtualBox sur la machine de travail (windows ou linux)
- copier les 2 fichiers source :
  - Linux.vbox
  - Linux.vdi
- ouvrir la VM : double-clic sur “Linux.vbox” (le fichier en bleu)



INSTALLATION VM

- installer VirtualBox sur la machine de travail (windows ou linux)
- copier les 2 fichiers source :
  - Linux.vbox
  - Linux.vdi
- ouvrir la VM : double-clic sur “Linux.vbox” (le fichier en bleu)



UTILISATION VM

- vérifier les configurations CPU et RAM, les augmenter dans la mesure des ressources disponibles
- démarrer la VM
- **NE PAS SE CONNECTER VIA L'ÉCRAN VIRTUAL BOX** (fond noir avec invite de commande “login: “)

UTILISATION VM

- **NE PAS SE CONNECTER VIA L'ÉCRAN VIRTUAL BOX** (fond noir avec invite de commande "login: ")
- sur un serveur distant vous n'avez pas accès à cet écran
- VirtualBox capture le clavier (et la souris), il faut une commande pour en sortir
- les copier-coller sont capricieux à même en fonctionnement

UTILISATION VM

- **NE PAS SE CONNECTER VIA L'ÉCRAN VIRTUAL BOX** (promis j'arrête de le répéter)
- lancer un terminal (gitbash, powershell)
- se connecter en ssh à la VM :
  - ssh -p 2222 user@localhost
  - mot de passe : user2023.
- passer en root :
  - sudo <commande>
  - sudo ls
  - mot de passe root : root2023.

DOCKER VM

NE PAS SUPPRIMER LES IMAGES PRÉCHARGÉES NOTAMMENT GITLAB

TRANSFERTS VM

1. utiliser une connexion sftp sur le port 2222
- avec filezilla ou autre
- spécifier port, user et password

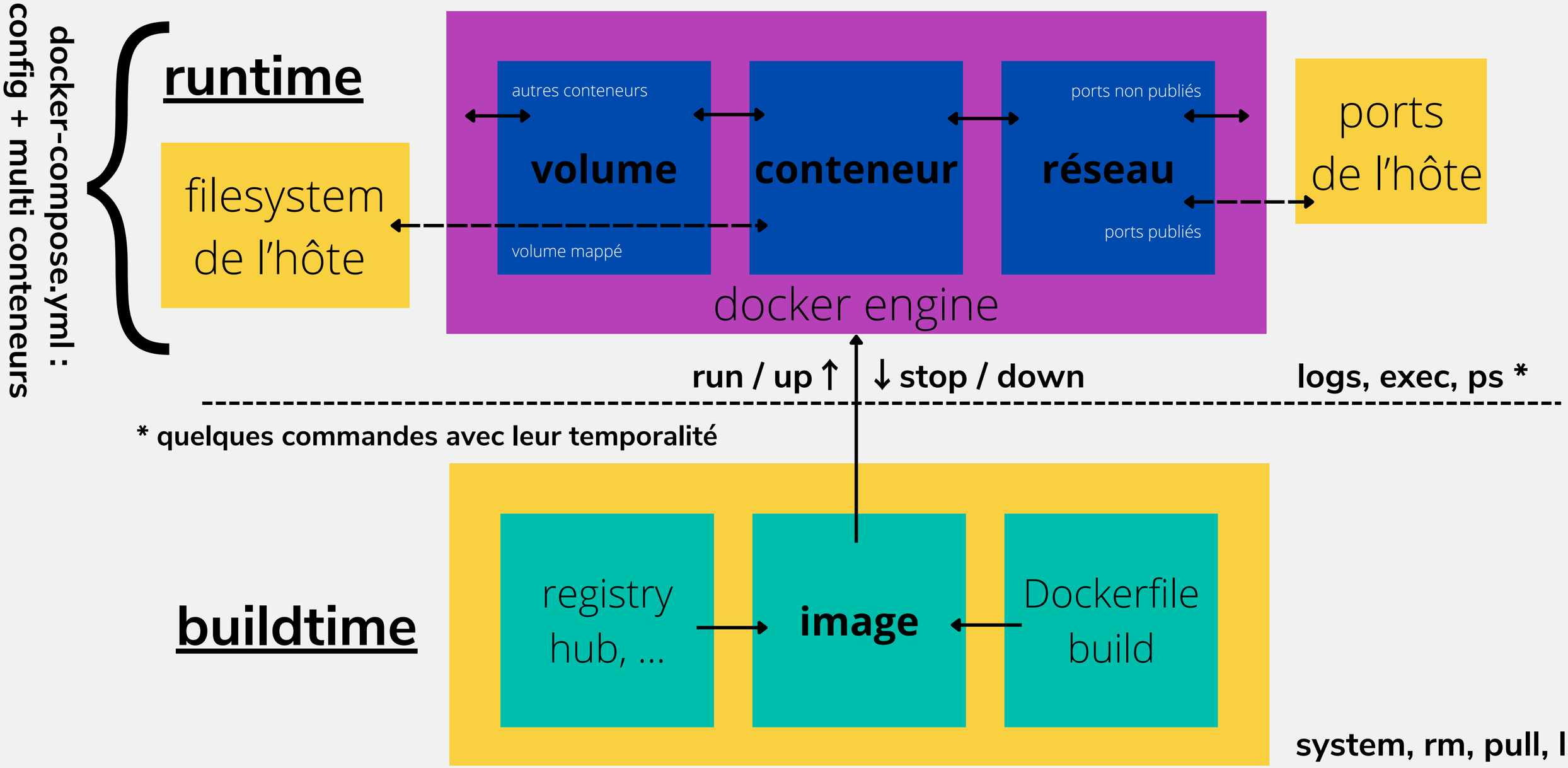
2. utiliser scp en ligne de commande (terminal) :
- copier des fichiers de l'hôte vers la VM :
  - scp /file/to/send username@remote:/where/to/put
  - scp -P 2222 test.txt user@localhost
- copier des fichiers de la VM vers l'hôte :
  - scp username@remote:/file/to/send /where/to/put
  - scp -P 2222 user@localhost:save.tar .

DOCKER TD#1

commandes de base

Écosystème docker

SOLSTICE
Coopérative d'entrepreneurs
Sébastien Grivolat



docker commandes courantes

SOLSTICE
Coopérative d'entrepreneurs
Sébastien Grivolat

- **run NOM_IMAGE**
  - télécharge si l'image n'est pas en local et démarre le conteneur : docker run hello-world
- **pull NOM_IMAGE**
  - télécharge l'image depuis le dépôt : docker pull ubuntu
- **ps**
  - statuts des conteneurs, par défaut les actifs : docker ps ou actifs et arrêtés : docker ps -a

docker commandes courantes

SOLSTICE
Coopérative d'entrepreneurs
Sébastien Grivolat

- **stop NAME/ID**
  - arrêter un conteneur : docker stop gitea
  - faire Ctrl-C si celui-ci n'est pas en tache de fond
- **start NAME/ID**
  - démarrer un conteneur : docker start gitea
- **kill NAME/ID**
  - force l'arrêt d'un conteneur : docker kill gitea

docker commandes courantes

SOLSTICE
Coopérative d'entrepreneurs
Sébastien Grivolat

- **logs NAME/ID**
  - traces d'un conteneur : docker logs gitea
- **rm NAME/ID**
  - supprimer un conteneur : docker rm gitea
- **exec OPTIONS NAME/ID CMD**
  - pour un conteneur actif, lancer une commande dans ce conteneur : docker exec -it gitea sh

tips linux

SOLSTICE
Coopérative d'entrepreneurs
Sébastien Grivolat

- **Ctrl-R** recherche de commande en arrière dans l'historique
- **history** afficher toutes les commandes déjà exécutées
- **wget localhost:XXX / curl localhost:XXX** pour vérifier si mon serveur web est actif, fonctionne avec une url
- **cat > MON_FICHIER.TXT** écrire un fichier, fin avec Ctrl-D
- **nano MON_FICHIER.TXT ou vi MON_FICHIER.TXT** éditer rapidement mon fichier sans éditeur type IDE disponible
- **ssh -p XX user@machine**
  - se connecter à distance ou à une VM depuis l'espace de travail habituel
  - permet aussi de partager des fichiers depuis une machine distante
- **sudo netstat -tulpa** voir les connexions réseau et les ports ouverts
- **sudo usermod -aG GROUP ${USER}** ajouter le groupe GROUP à l'utilisateur courant

tips linux

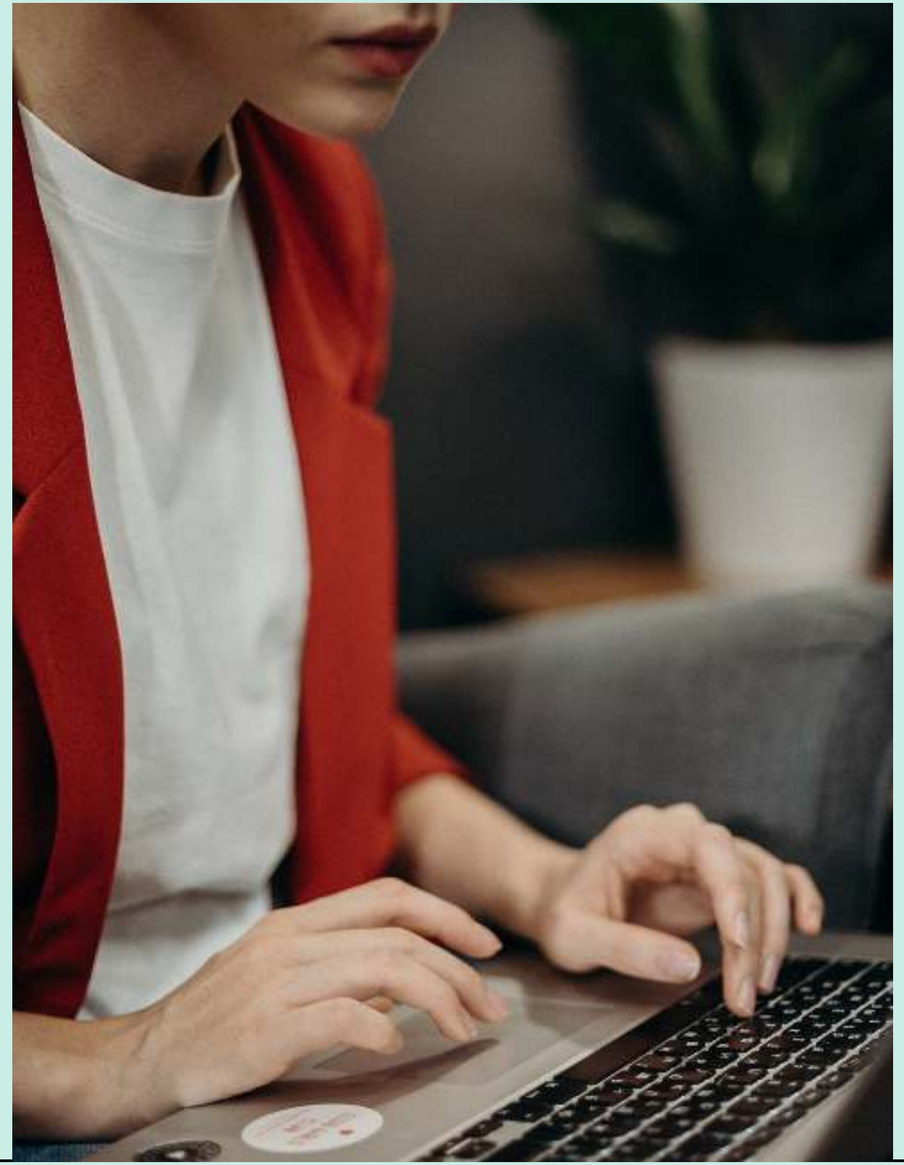
SOLSTICE
Coopérative d'entrepreneurs
Sébastien Grivolat

- **docker run --help | man COMMANDE** avoir l'aide en ligne sur une commande
- **sh ou bash** dans un conteneur (via exec) :
  - **ping** : vers les autres conteneurs, avec leur nom
  - **ip a** : adresse du conteneur
  - **ps aux** : lister les processus en cours dans le conteneur
  - **netstat -tulpa** : ports exposés par le conteneur
  - **/var/log ou ailleurs** : accéder à des logs internes
  - **changements internes** de conf ou de contenu : attention aux volumes en read-only, les modifs sont perdues au redémarrage du conteneur, pour tester en live uniquement
- **reboot** redémarre la machine sudo reboot
- **shutdown**
  - arrête la machine sudo shutdown -h now
  - à éviter à distance sans accès physique

DÉMARRAGE

- Lancer un terminal ssh (comme avant)

- vérifier l'installation et les droits
  - docker ps
  - docker run hello-world



REDIS & WEB

- Lancer 2 terminaux ssh (voir 3)

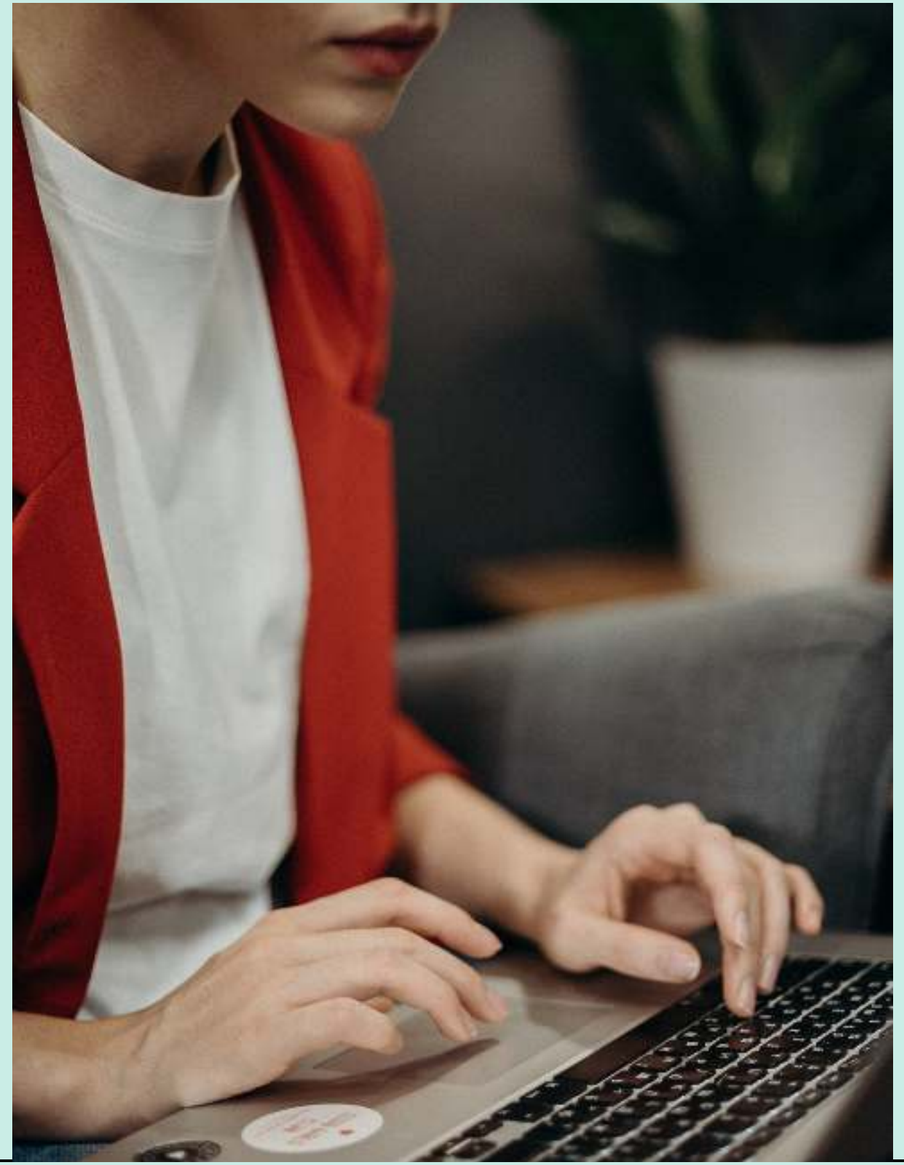
- Téléchargement et exécution

docker run redislabs/redismod
docker run phisit11/nginx-nodejs-redis-web1

- Monitor

docker ps

=> lancer 2 conteneurs, les surveiller



REDIS & WEB

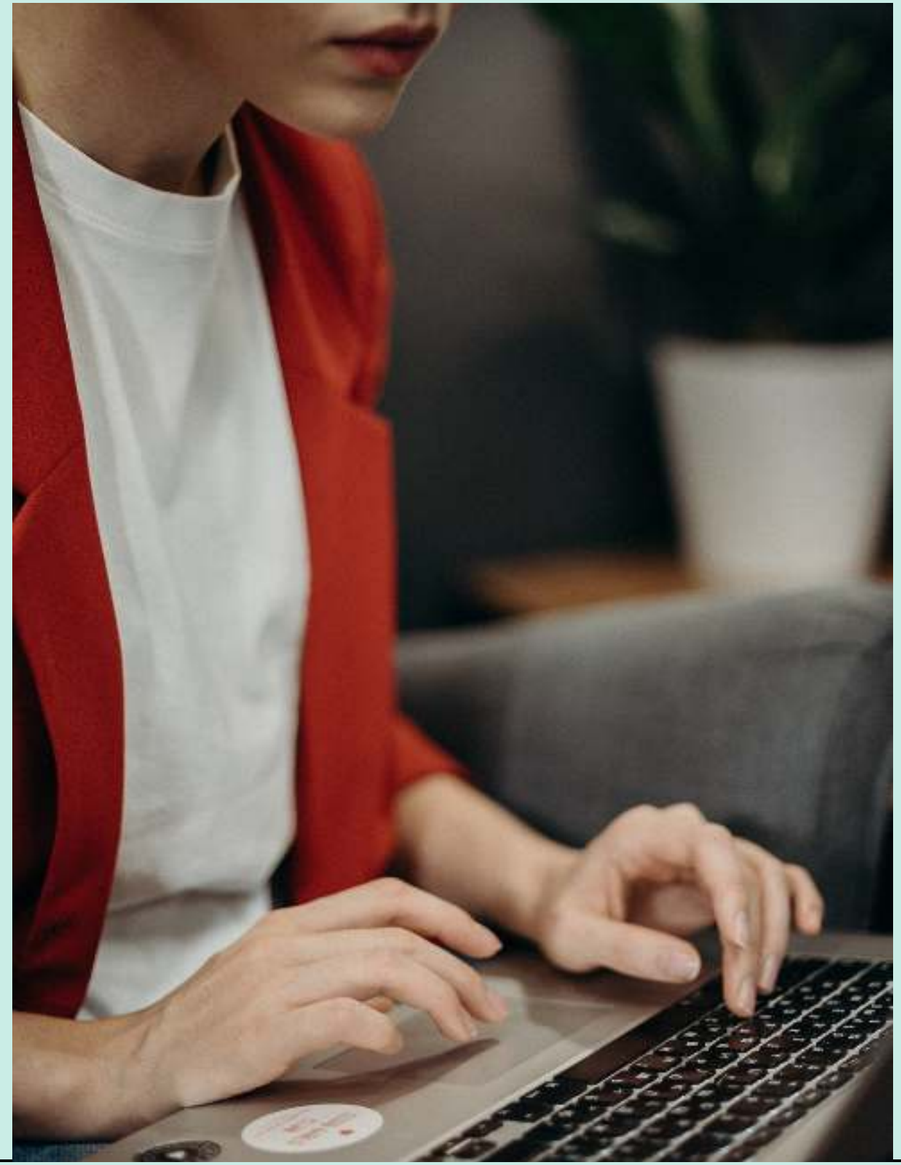
- Ajouter la version :

docker run phisit11/nginx-nodejs-redis-web1:0227

- Vérifier le ping :

docker run -it phisit11/nginx-nodejs-redis-web1:0227 sh
ping redis

=> lire les messages d'erreur



- Créer un réseau

docker network create docker

- Relancer les conteneurs :

docker run --network docker redislabs/redismod
docker run --network docker phisit11/nginx-nodejs-redis-web1:0227

- Vérifier le ping :

docker run -it phisit11/nginx-nodejs-redis-web1:0227 sh
ping redis

=> faire communiquer 2 conteneurs

REDIS & WEB

REDIS & WEB

- Ajouter les hostnames :

docker run --network docker --hostname redis redislabs/redismod

docker run --network docker --hostname web phisit11/nginx-nodejs-redis-web1:0227

- Vérifier le ping :

docker run -it phisit11/nginx-nodejs-redis-web1:0227 sh

ping redis

=> nommer ses conteneurs

REDIS & WEB

- Ouvrir le port redis :

docker run --network docker --hostname redis -p 6379:6379 redislabs/redismod

docker run --network docker --hostname web phisit11/nginx-nodejs-redis-web1:0227

=> ouvrir des ports réseau hors du moteur docker

REDIS & WEB

- Ouvrir le port web :

docker run --network docker --hostname redis -p 6379:6379 redislabs/redismod

docker run --network docker --hostname web -p 80:5000 phisit11/nginx-nodejs-redis-web1:0227

- Vérifier (plusieurs fois)

curl localhost

wget localhost

=> avoir un échange fonctionnel entre 2 conteneurs

REDIS & WEB

- Persister les données :

docker run --network docker -v redis_data:/data --hostname redis -p 6379:6379 redislabs/redismod
docker run --network docker --hostname web -p 8080:5000 phisit11/nginx-nodejs-redis-web1:0227

- tester le compteur, arrêter, relancer, retester

=> utiliser un volume

- lancer en tache de fond (detach)

docker run -d --network docker -v redis_data:/data --hostname redis -p 6379:6379 redislabs/redismod
docker run -d --network docker --hostname web -p 8080:5000 phisit11/nginx-nodejs-redis-web1:0227

- arrêter un conteneur

docker stop ID/NAME

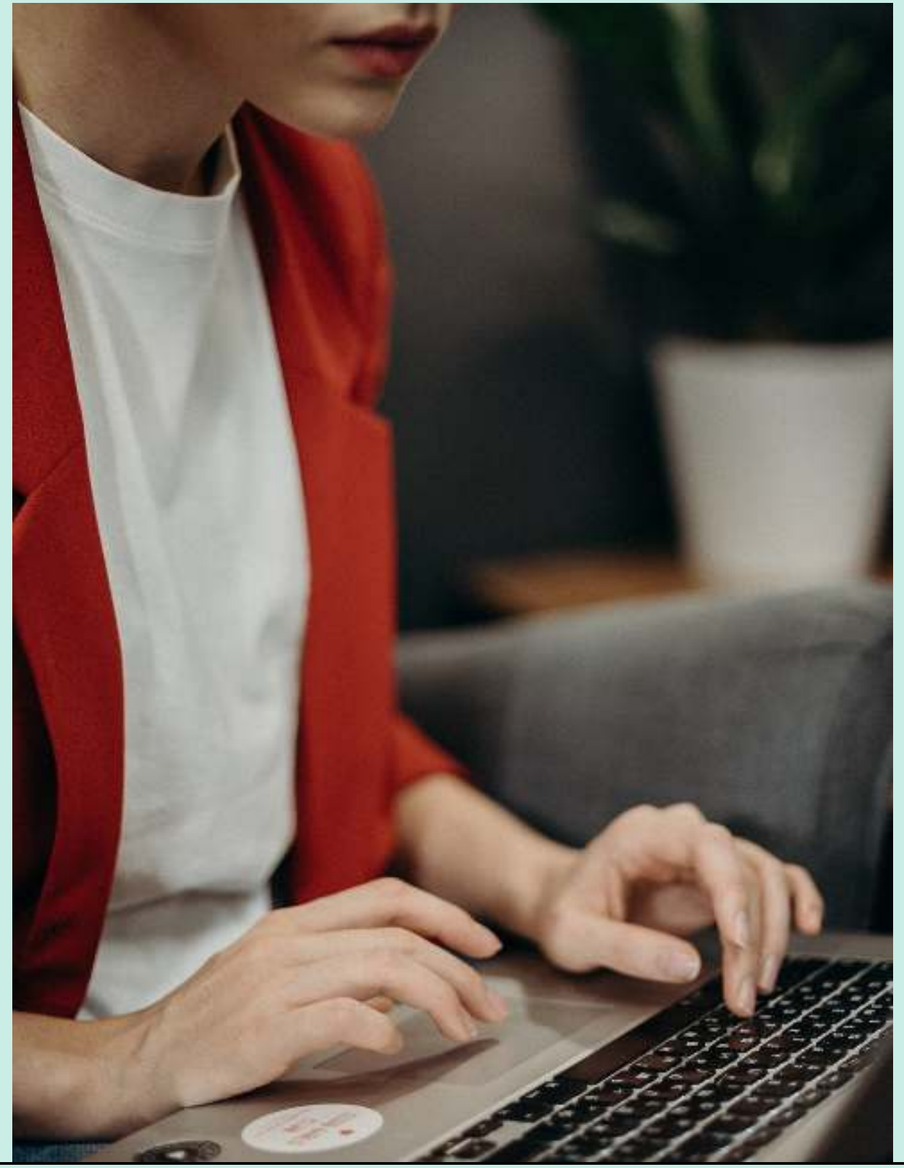
- voir les logs d'un conteneur

docker logs ID/NAME

=> utiliser quelques options et commandes de gestion docker

REDIS & WEB : CONCEPTS

- images + version
- conteneurs + images
- network
- volumes
- paramètres (hostname, ports)
- commandes (ps, stop, run)
- options (detach, it + cmd)
- linux (ping, curl, wget)

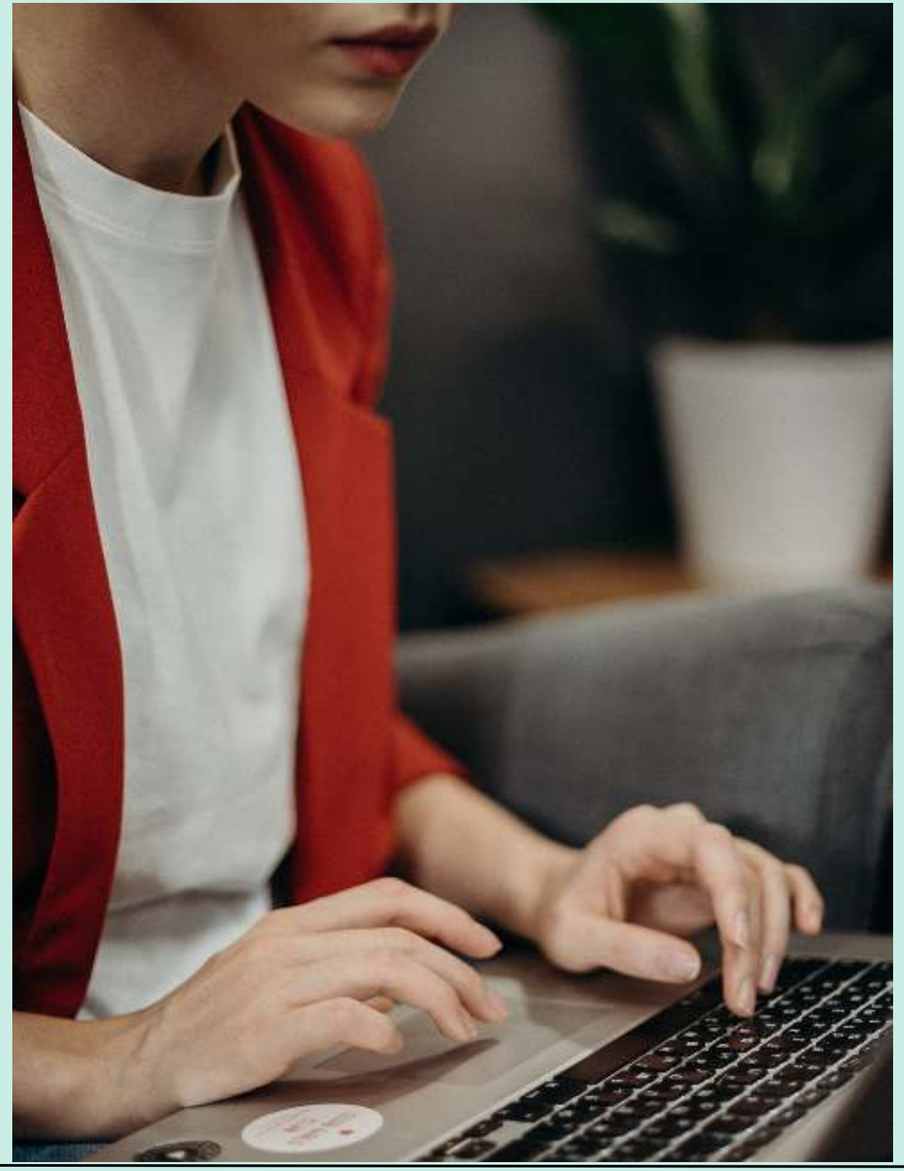


UBUNTU

docker run -it --network docker -v redis_data:/data --hostname ubuntu ubuntu
ls /data/

- accéder aux autres ressources docker
- tester une commande
- installer une version spécifique
- disposer d'un outil incompatible avec l'OS de l'hôte

=> utiliser un conteneur de secours pour débugger



COMMIT

Faire un commit qui contient toutes les étapes que vous avez effectuées et leurs explications.

Par exemple utiliser :
$ history >> ./journal.md
$ nano ./journal.md
=> expliquer ce que fait chaque commande (les plus importantes) en une phrase
$ git add *.md
$ git commit -m ‘TP2 commands for redis and ngix’
$ git push

docker run options standards

- quand ':' est utilisé à **gauche** se sont les **valeurs de l'hôte** et à **droite** du **conteneur**.
- **-i ou --interactive**
  - pour avoir shell, lancer une commande ou pour des images spécifiques : docker run -i ubuntu sh
- **-v ou --volume**
  - mapper le système de fichier de l'hôte dans le conteneur, soit un répertoire (par défaut) soit un fichier spécifique : docker run -it -v /home/${USER}:/home/ubuntu ubuntu
- **-d ou --detach**
  - exécution en tache de fond

docker run options standards

- **--network**
  - spécifier le réseau virtuel pour le conteneur : docker run --network host ubuntu
  - l'objectif est de partager un réseau entre plusieurs conteneurs
- **-p ou --publish**
  - une image peut exposer par défaut les ports du conteneur, si ce n'est pas le cas ou que l'on souhaite un port spécifique : docker run -p 8022:22 ubuntu
  - -P ou --publish-all ouvre tous les ports du conteneur sur des ports de hôtes aléatoires
  - **Attention si il y a une VM** qui est l'hôte docker il faut aussi mapper le port exposé par docker en dehors de la VM pour y accéder de l'extérieur (port pour ssh, pour un navigateur web).

docker gestion des "objets"

SOLSTICE
Coopérative d'entrepreneurs
Sébastien Grivolat

- **docker image** : gérer les images docker :
  - **pull IMAGE** télécharger une image : docker image pull gitea/gitea
  - **rm ID/NAME** supprimer une image : docker image rm gitea
  - **prune** supprimer les images (par défaut non utilisées) : docker image prune
  - **ls** lister les images : docker image ls
- **docker network** : gérer les réseaux docker :
  - **ls** lister les réseaux : docker network ls
  - **rm ID/NAME** supprimer un réseau : docker network rm host
- **docker container** : gérer les conteneurs docker :
  - **ls** lister les conteneurs actifs : docker container ls
  - **ls -a** lister tous les conteneurs : docker container ls -a
  - **rm ID/NAME** supprimer un conteneur : docker container rm gitea

docker gestion des "objets"

SOLSTICE
Coopérative d'entrepreneurs
Sébastien Grivolat

- **docker volume** :
  - **ls** lister les volumes : docker volume ls
  - **rm ID/NAME** supprimer : docker volume rm gitea
- **docker system** : gérer docker :
  - **df** voir l'occupation du disque dur : docker system df
  - **prune** supprimer les objets non utilisés : docker system prune
  - **prune --all** supprimer plus d'objets non utilisés : docker system prune --all
  - **df --help** avoir de l'aide sur df : docker system df --help
- **Tip** :
  - arrêter tous les conteneurs :
  - docker stop $(docker container ls -q)
  - supprimer toutes les images et les conteneurs pas de confirmation :
  - docker image rm -f $(docker image ls -q)

DOCKER BDD / IHM WEB

mysql + phpmyadmin avec docker

d'après

https://hub.docker.com/_/mysql

https://hub.docker.com/_/phpmyadmin

objectif :

=> mettre en place une bdd avec admin

COMMIT

Faire un commit qui contient toutes les étapes que vous avez effectuées, commentées.

Voir la suggestion du commit précédent.

DOCKER WALLABAG

bdd + wallabag avec docker

d'après

https://github.com/wallabag/docker

choisir un des bdd précédentes

commiter

objectif :

=> mettre en place un appli et une bdd

DOCKER - OPTIONNEL

mysql + mariadb + phpmyadmin avec docker

https://hub.docker.com/_/mariadb

commiter

objectif :

=> mettre en place 2e bdd similaire avec admin

DOCKER - OPTIONNEL

postgre + phppgadmin

https://hub.docker.com/_/postgres

https://hub.docker.com/r/dpage/pgadmin4

commiter

objectif :

=> mettre en place une autre bdd avec admin