Containers contd

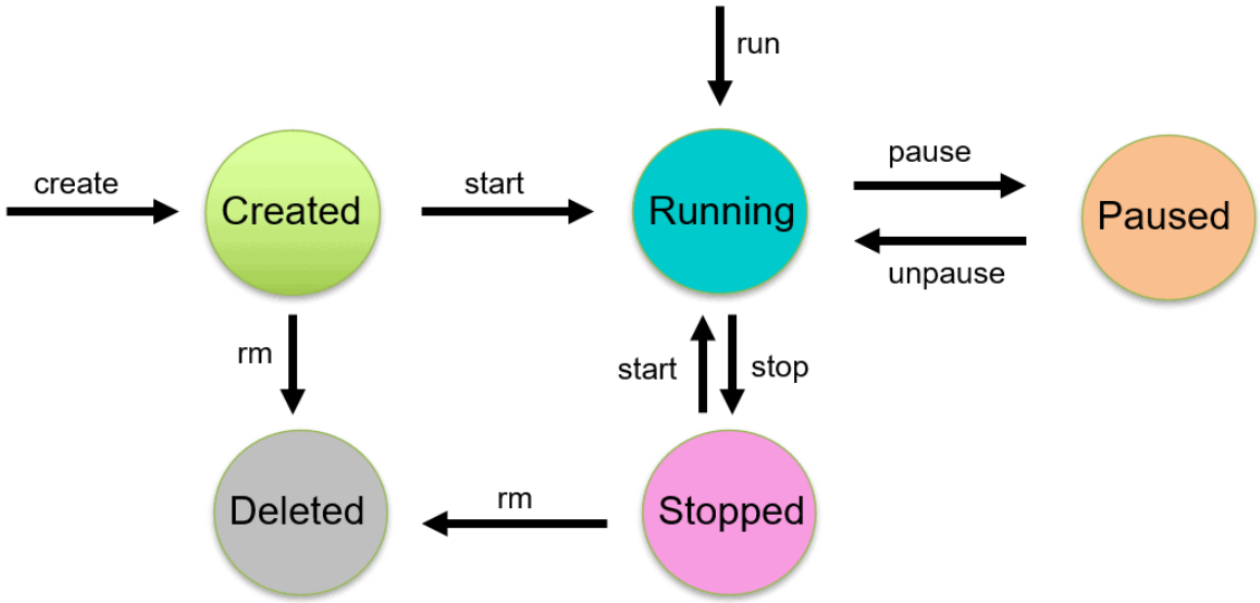
Images and Tags

- Every docker image has a tag (represents the version)
- if the tag is not provided, default tag would be `latest`
- image name `<repository>:<tag>`

```
nginx  => nginx:<tag>
       => nginx:perl

mysql => mysql:8.0-debian

jenkins => jenkins/jenkins:jdk17

        jenkins/jenkins => jenkins/jenkins:latest
```

Container lifecycle states

- Overview

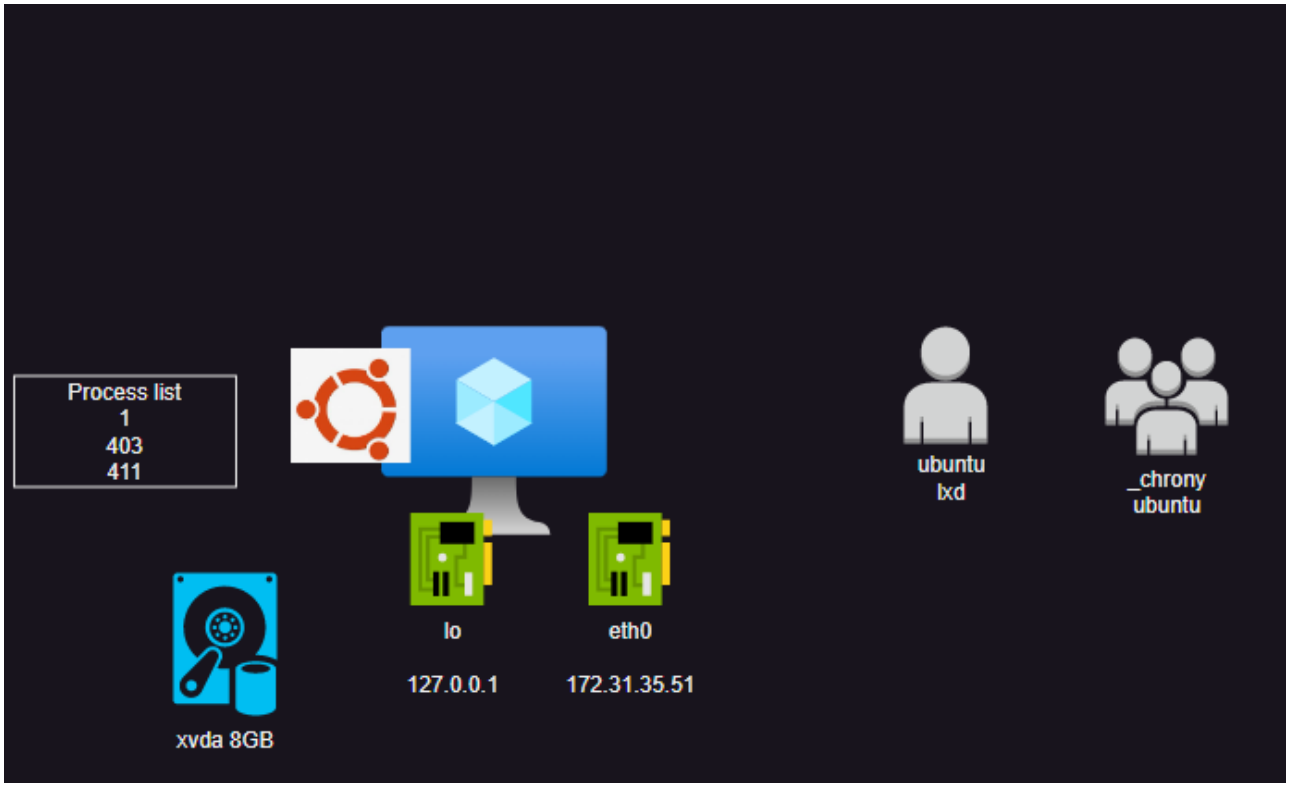


- refer classroom video for commands execute

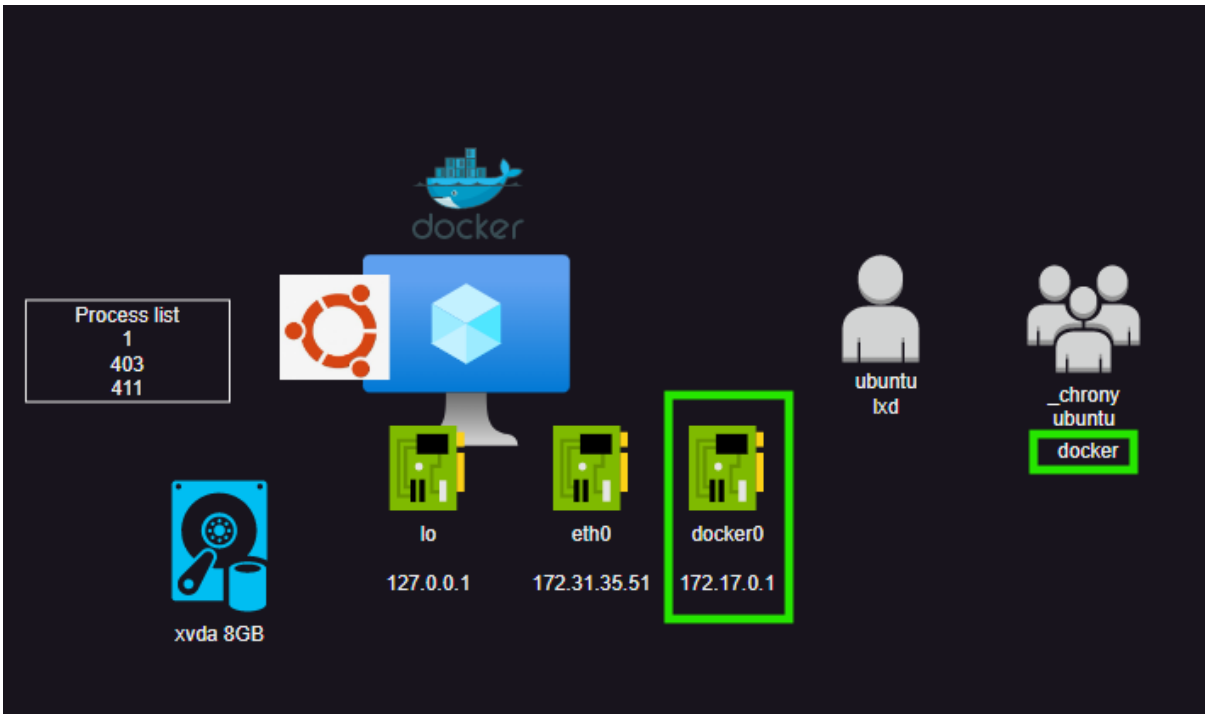
```
docker container --help
or
cheatsheet
```

Docker installation effects

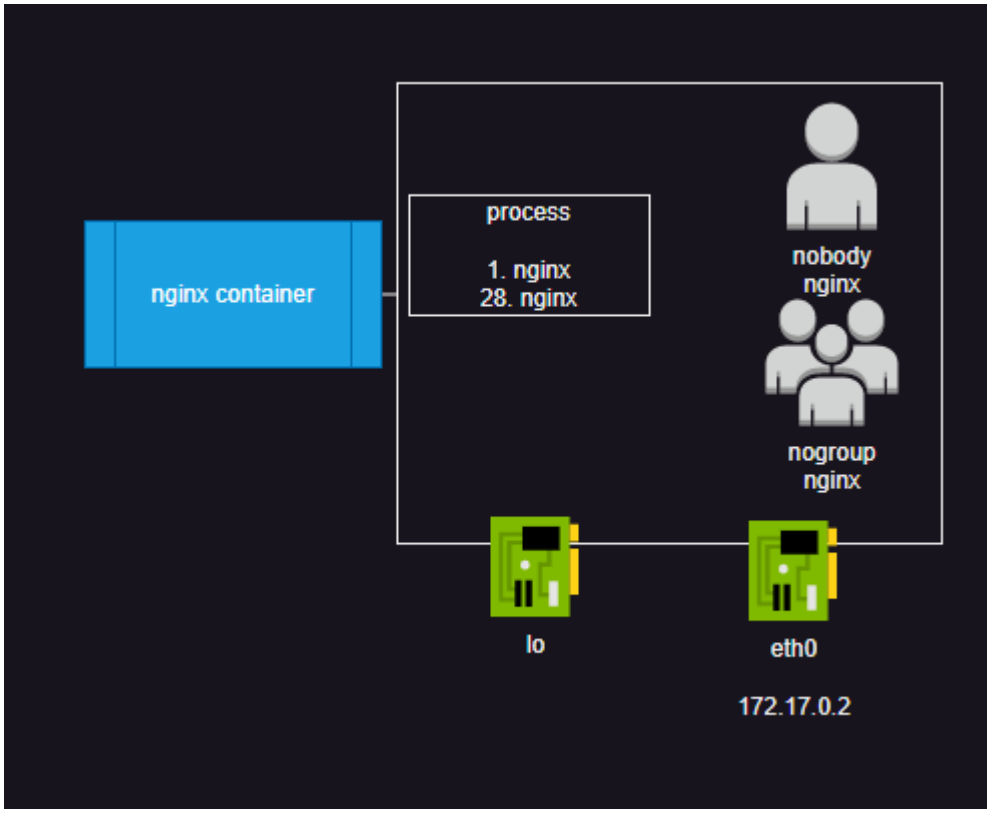
- Before installing docker on ubuntu server



- Install docker lead to creation of
  - a new network interface `docker0`
  - a new group `docker`



- Lets run a container and peek into container



- inside container, we have
    - root folder with all necessary files to run the application
    - network interfaces with ip address
    - process tree (pid 1: This is generally your application start command)
    - users and groups specific to application
- Container is an isolated area which gets its own
    - network interface
    - file system
    - users
    - groups
    - resources
        - vcpu
        - RAM
- Containers are created with the help of core linux concepts called as
    - namespaces
    - cgroups
- Container starts only one process i.e. your application and will be in running state as long as that process lasts

Exercise

- Either in docker playground or by installing docker on a linux vm in aws/azure, try executing all the states in docker container lifecycle by using the image `httpd`

- ensure `docker container ls` before changing state.