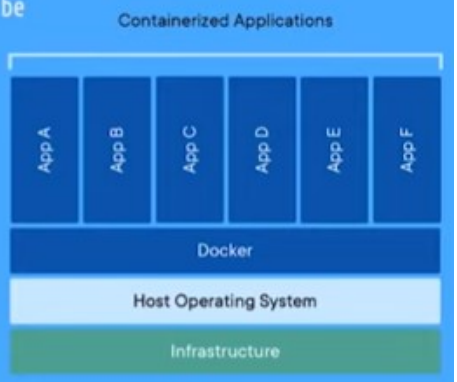
What is Docker?

Docker is an open source standalone application which works as an engine used to run containerized applications. It is installed on your operating system, preferably on Linux, but can be also installed on Windows and macOS.

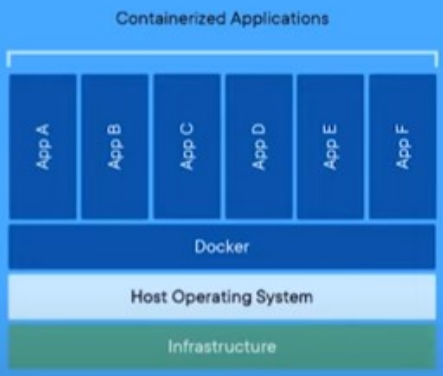
- An application running in a container is isolated from the rest of the system and from other containers, but gives the illusion of running in its own OS instance.
- Multiple Docker containers can be run on the single operating system simultaneously, you can manage those containers with Docker.
- Docker applications run in containers that can be used on any system: a laptop, on premises, or in the cloud.
- Simply we can say Docker is a container management service.



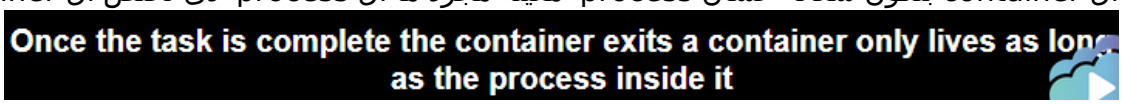
What is a Docker container?

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.

- Containerization been around for a long time, but it was introduced in a different way by Docker.
- It packages applications as images that contain everything needed to run them: code, runtime environment, libraries, and configuration.
- A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

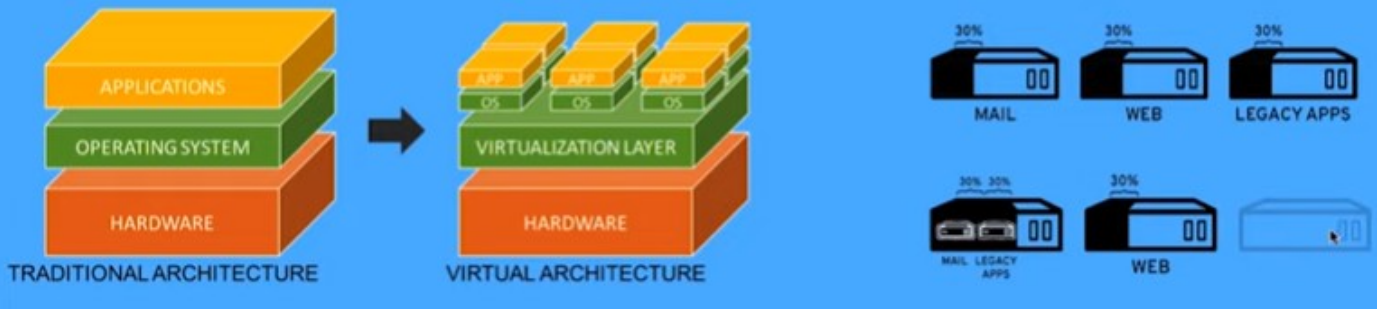


- ال container بتكون شغالة عشان process معينة مجرد ما ال process دى تخلص ال container بيتقفل



Virtualization

Virtualization is technology that lets you create useful IT services using resources that are traditionally bound to hardware. It allows you to use a physical machine's full capacity by distributing its capabilities among many users or environments.



How does virtualization work?

Software called hypervisors also known as a virtual machine monitor (VMM) separate the physical resources from the virtual environments.

Hypervisors can sit on top of an operating system (desktop or server), hypervisors take your physical resources (Processor, RAM, Hard Disk) and divide them up so that virtual environments can use them.

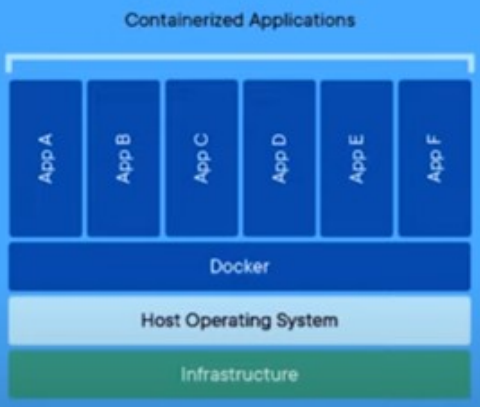
Popular Hypervisors

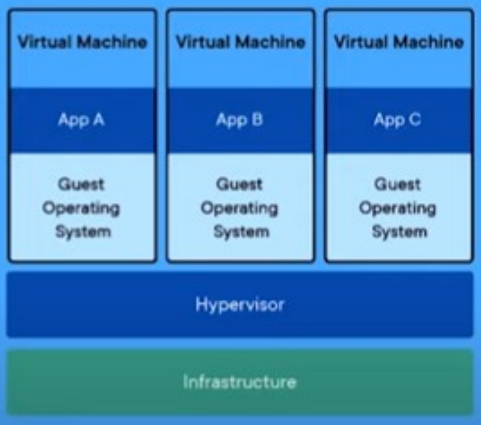
Microsoft Hyper-v

vmware vSphere

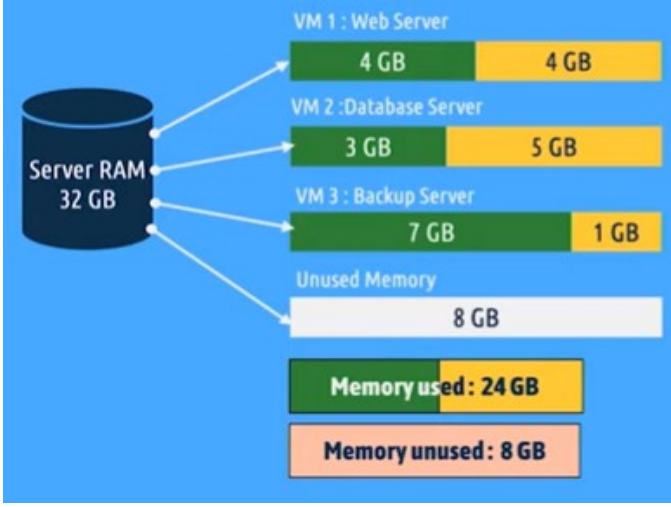
Virtualization VS Containerization

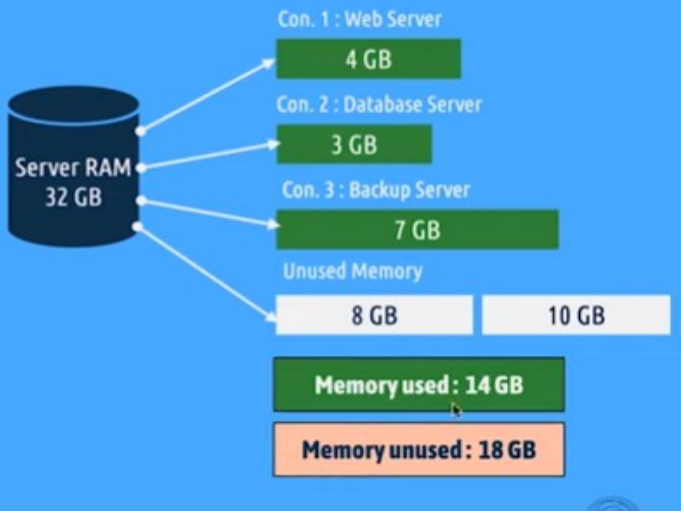
- Portability
- Lightweight
- Native Performance
- Start up in milliseconds
- Multiple containers
- Simple and Fast Deployment





Virtualization VS Containerization





Who is Docker for?

Docker is a tool that is designed to benefit both developers and system administrators, making it a part of many DevOps.

Do you have to use Docker to use containers?

No. Docker is just one set of tools that work with the container features provided by Linux and Windows. Support for containers has been part of Linux for a long time and has matured into a stable and reliable feature.

The main competitor to Docker is rkt, which is produced by a company called CoreOS .

Is Docker Free?

Docker Community Edition (CE) is free for anyone to use. This version of Docker is open source and can be used on a variety of platforms including Windows, Mac, and Linux.

Docker Architecture

Image

- An image is a read-only template with instructions for creating a Docker container. you may build, an image which is based on the Ubuntu image or SQL Server.

Container

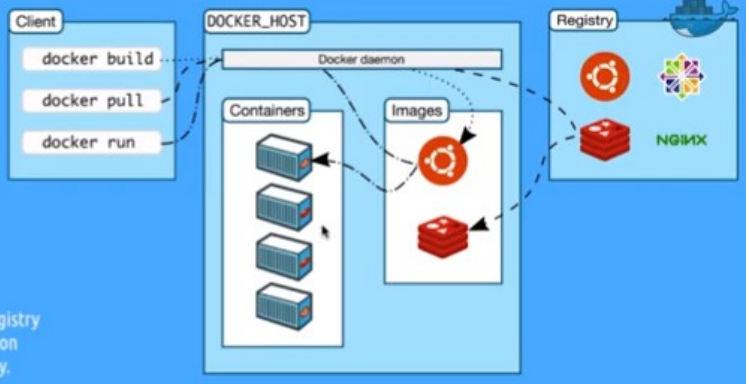
- A container is a runnable instance of an image. You can create, start, stop, move, or delete a container using the Docker API or CLI.

Registry

- A Docker registry stores Docker images. Docker Hub is a public registry that anyone can use, and Docker is configured to look for images on Docker Hub by default. You can even run your own private registry.

Client

- The Docker client is the primary way that many Docker users interact with Docker. When you use commands such as docker run, the client sends these commands to dockerd, which carries them out. The docker command uses the Docker API.



Docker daemon

- The Docker daemon listens for Docker API requests and manages Docker objects such as images, containers, networks, and volumes.

Docker Architecture

Namespaces

- Docker uses a technology called `namespaces` to provide the isolated workspace called the container. When you run a container, Docker creates a set of namespaces for that container. These namespaces provide a layer of isolation. Each aspect of a container runs in a separate namespace and its access is limited to that namespace.

Docker Tags

Docker tags are reference to docker images.

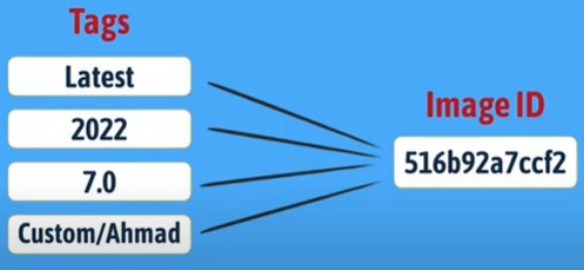


Image Layers

- بستخدمه عشان لما اجى اتعامل مع ال docker image لل refrence بتكون عبارة عن ال docker tag
  لا بستخدم ال container-id بدل ما انده عليه بال container docker tag

Docker Commands :

Install and push image:

Docker pull image-name	لو عاوز انزل image معينة من ال registry

```
 Experimental:      true
PS C:\Users\97150> docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:97a379f4f88575512824f3b352bc03cd75e239179eea0fecc38e597b2209f49a
Status: Downloaded newer image for hello-world:latest
```

Docker run image-name Docker run imagename:tag	دة بيعمل حاجتين بيروح يعمل run لل image اللى انت مديهاله لو موجودة هايعملها run عالطول لو مش موجودة هايروح ينزلها من ال registry و باعدين يعملها run لو ما حددتش تاج معين زى redis:4.0 بيروح هو بينزلى ال latest

```
PS C:\Users\97150> docker run fedora
Unable to find image 'fedora:latest' locally
latest: Pulling from library/fedora
edad61c68e67: Pull complete
Digest: sha256:40ba585f0e25c096a08c30ab2f70ef3820b8ea5a4bdd16da0edbfc0a6952fa57
Status: Downloaded newer image for fedora:latest
```

```
PS C:\Users\97150> docker run ubuntu:17.10
```

Docker push image-name	لو عاوز ارفع حاجة ع docker hub

```
PS C:\Users\Ahmad> docker push ahmadmohey/redis
Using default tag: latest
The push refers to repository [docker.io/ahmadmohey/redis]
262de04acb7e: Mounted from library/redis
45f6df634253: Mounted from library/redis
e46136075591: Mounted from library/redis
11f991845040: Mounted from library/redis
dd1ebb1f5319: Mounted from library/redis
814bff734324: Mounted from ahmadmohey/nginx
latest: digest: sha256:1bd57e1a42b99ae53412b582784d0362fa8205243ce5f289cb4f76de2907cb97 size: 1574
```

- لو عندى اكتر من image with different tags هايعرف ال latest الاول

List images and containers :

Docker ps -a	بيرجعلى كل ال containers اللى عندى بس لو فى command عملتله run اكتر من مرة هو بيجبلى اخر واحد اتعمله run

```
PS C:\Users\97150> docker ps -all
CONTAINER ID   IMAGE         COMMAND    CREATED          STATUS                      PORTS     NAMES
73126f4ac989   hello-world   "/hello"   8 seconds ago    Exited (0) 6 seconds ago              amazing_aryabhata
```

Docker container ls -a	بيرجعلى كل ال containers اللى عندى برده بس لو فى command عملتله run اكتر من مرة هو بيجبلهوملى كلهم

```
PS C:\Users\97150> docker container ls -a
CONTAINER ID   IMAGE         COMMAND    CREATED          STATUS                      PORTS     NAMES
73126f4ac989   hello-world   "/hello"   17 seconds ago   Exited (0) 15 seconds ago             amazing_aryabhata
c4d4f9f56fd6   hello-world   "/hello"   26 minutes ago   Exited (0) 26 minutes ago             naughty_elbakyan
```

Docker images	بيجبلى كل ال images اللى انا عملتلها download

```
PS C:\Users\97150> docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
fedora        latest    b78af7a83692   2 months ago   153MB
hello-world   latest    feb5d9fea6a5   4 months ago   13.3kB
```

ال image دة اللى بنزله من على ال registry بس ال container دة ال image و هى شغالة عندى ع الجهاز

Container logs and info :

Docker inspect image-name	بيرجعلى شوية معلومات عن ال image

```
PS C:\Users\97150> docker inspect redis
[
    {
        "Id": "sha256:f1b6973564e91aecb808142499829a15798fdc783a30de902bb0c4133fee19ad",
        "RepoTags": [
            "redis:latest"
        ],
        "RepoDigests": [
            "redis@sha256:0d9c9aed1eb385336db0bc9b976b6b49774aee3d2b9c2788a0d0d9e239986cb3"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2022-01-26T22:42:40.969131359Z",
        "Container": "2552e57869499f961c051f933f396e9a108a328aa50f0527e7b709c8453e2e5d",
        "ContainerConfig": {
            "Hostname": "2552e5786949",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
```

Docker logs container-id	بيرجعلى ال logs بتاعة ال container دى

```
PS C:\Users\97150> docker logs 8da1839b2f39
1:C 18 Feb 2022 13:27:24.651 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
1:C 18 Feb 2022 13:27:24.651 # Redis version=6.2.6, bits=64, commit=00000000, modified=0, pid=1, just started
1:C 18 Feb 2022 13:27:24.651 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf
1:M 18 Feb 2022 13:27:24.652 * monotonic clock: POSIX clock_gettime
1:M 18 Feb 2022 13:27:24.654 * Running mode=standalone, port=6379.
1:M 18 Feb 2022 13:27:24.654 # Server initialized
1:M 18 Feb 2022 13:27:24.654 # WARNING overcommit_memory is set to 0! Background save may fail under low memory condition. To fix this issue add 'vm.overcommit_memory = 1' to /etc/sysctl.conf
and then reboot or run the command 'sysctl vm.overcommit_memory=1' for this to take effect.
1:M 18 Feb 2022 13:27:24.655 * Ready to accept connections
```

Docker stats container-id	بيرجعلى ال container دة واخد قد ايه من ال resources بتاعة الجهاز بتاعى

```
PS C:\Users\97150> docker stats 8da1839b2f39
CONTAINER ID   NAME          CPU %     MEM USAGE / LIMIT     MEM %     NET I/O       BLOCK I/O   PIDS
8da1839b2f39   gifted_wing   0.05%     7.246MiB / 6.101GiB   0.12%     1.3kB / 0B    0B / 0B     5
CONTAINER ID   NAME          CPU %     MEM USAGE / LIMIT     MEM %     NET I/O       BLOCK I/O   PIDS
8da1839b2f39   gifted_wing   0.33%     7.246MiB / 6.101GiB   0.12%     1.3kB / 0B    0B / 0B     5
```

Docker info	بيرجع شوية داتا عن ال docker بتاعى

```
PS C:\Users\97150> docker info
Client:
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc., v0.7.1)
  compose: Docker Compose (Docker Inc., v2.2.3)
  scan: Docker Scan (Docker Inc., v0.17.0)

Server:
 Containers: 3
  Running: 1
  Paused: 0
  Stopped: 2
 Images: 2
 Server Version: 20.10.12
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: cgroupfs
 Cgroup Version: 1
```

docker inspect --format='{{range.NetworkSettings.Networks}}{{.IPAddress}}{{end}}' container-id
دى بيرجعلى ال IP Address بتاع ال container دة

```
PS C:\Users\97150> docker inspect --format='{{range.NetworkSettings.Networks}}{{.IPAddress}}{{end}}' 8da1839b2f39
172.17.0.2
```

ممكن استخدم ال container-id او name او اول 4 ارقام من ال container-id بس يكونوا مميزين

Container and image manegment:

Docker stop container-id	عشان اوقف container

```
PS C:\Users\97150> docker stop n1
n1
```

- لو عملت force stop ل container شغال هاتلاقى ال exit code مختلف عن ال 0 لو 0 معنى كدة انه خلص ع خير

```
PS C:\WINDOWS\system32> docker run -d centos sleep 2000
e096f662f361825c967e75d06239808ba68bd59d26d6520d2a7420201b14b359
PS C:\WINDOWS\system32> docker ps
CONTAINER ID   IMAGE     COMMAND        CREATED         STATUS         PORTS     NAMES
e096f662f361   centos    "sleep 2000"   9 seconds ago   Up 6 seconds             lucid_diffie
PS C:\WINDOWS\system32> docker stop e096f662f361
e096f662f361
PS C:\WINDOWS\system32> docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
PS C:\WINDOWS\system32> docker ps -a
CONTAINER ID   IMAGE     COMMAND               CREATED             STATUS                          PORTS     NAMES
e096f662f361   centos    "sleep 2000"          43 seconds ago      Exited (137) 10 seconds ago               lucid_diffie
e2f50c1171bb   centos    "sleep 20"            5 minutes ago       Exited (0) 5 minutes ago                  youthful_austin
```

Docker start container-id or name	عشان ارجع اشغل ال container من تانى

```
PS C:\Users\97150> docker container start n1
n1
```

Docker rm container-id	دة بيشيل ال container اللى انا مديله ال id بتاعه

```
PS C:\Users\97150> docker rm c4d4f9f56fd6
c4d4f9f56fd6
```

طالما رجعلى ال container id تانى يبقى كدة تمام

```
PS C:\WINDOWS\system32> docker rm c643b 9dbc1 86f4
c643b
9dbc1
86f4
```

عشان امسح اكتر من container مع بعض

Docker image rm image-name Docker rmi image-name	بيشيل ال image اللى عندى من ال repository

```
PS C:\Users\97150> docker image rm hello-world
Untagged: hello-world:latest
Untagged: hello-world@sha256:97a379f4f88575512824f3b352bc03cd75e239179eea0fecc38e597b2209f49a
Deleted: sha256:feb5d9fea6a5e9606aa995e879d862b825965ba48de054caab5ef356dc6b3412
Deleted: sha256:e07ee1baac5fae6a26f30cabfe54a36d3402f96afda318fe0a96cec4ca393359
```

عشان امسح ال image لازم مايكونش عندى containers شغالين من نفس ال image دى فلازم امسح ال containers اللى معمولين من ال image دى عشان اقدر امسح ال image ذات نفسها

Docker container run image-name	بينزل ال image دى و بيشغل منها container

Redis دى داتا بيز بس خفيفة جدا

```
PS C:\Users\97150> docker container run redis
Unable to find image 'redis:latest' locally
latest: Pulling from library/redis
5eb5b503b376: Pull complete
6530a7ea3479: Pull complete
91f5202c6d9b: Pull complete
9f1ac212e389: Pull complete
82c311187b72: Pull complete
da84aa65ce64: Pull complete
Digest: sha256:0d9c9aed1eb385336db0bc9b976b6b49774aee3d2b9c2788a0d0d9e239986cb3
Status: Downloaded newer image for redis:latest
1:C 18 Feb 2022 13:23:36.139 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo
1:C 18 Feb 2022 13:23:36.139 # Redis version=6.2.6, bits=64, commit=00000000, modified=0, pid=1, just started
1:C 18 Feb 2022 13:23:36.139 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf
1:M 18 Feb 2022 13:23:36.140 * monotonic clock: POSIX clock_gettime
1:M 18 Feb 2022 13:23:36.142 * Running mode=standalone, port=6379.
1:M 18 Feb 2022 13:23:36.142 # Server initialized
1:M 18 Feb 2022 13:23:36.142 # WARNING overcommit_memory is set to 0! Background save may fail under low memory condition. To fix this issue add 'vm.overcommit_memory = 1' to /etc/sysctl.conf
and then reboot or run the command 'sysctl vm.overcommit_memory=1' for this to take effect.
1:M 18 Feb 2022 13:23:36.143 * Ready to accept connections
```

بس هنا ال redis خد منى ال cruiser و مش قادر اكتب و لو عملت ls هلاقي ان ال status بتاعته running حلها ايه يقى ؟

```
PS C:\Users\97150> docker container run -d redis
8da1839b2f39b72d65128c4f1864458a579975c437aaf86ceae13e3d557eb10c
```

d- معناها detached انى ال container دة بيحصله run in background

Docker run -d image-name sleep 20	لو عاوز اعمل run ل image in background d : detach-

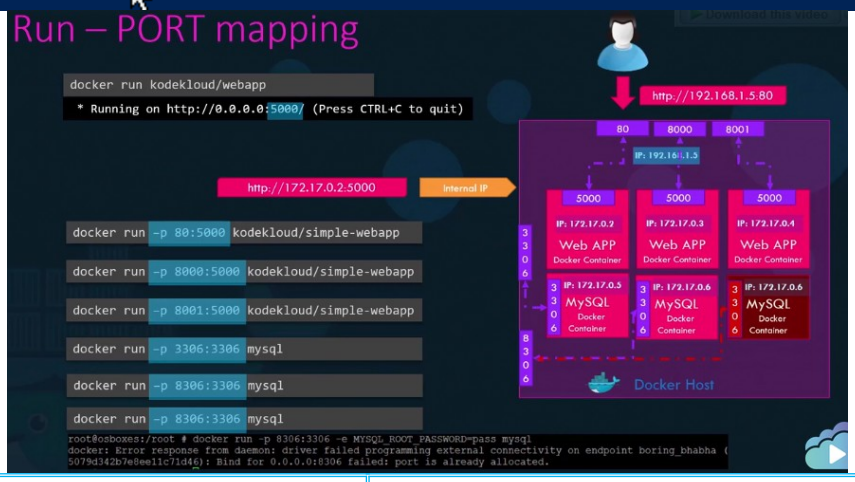
```
PS C:\WINDOWS\system32> docker run -d centos sleep 20
e2f50c1171bb1a648f4f1de06db4f53c370573f02954b0fed788063f9fefbff1
```

لو حبيت اعمله attach تانى بستخدم دة : docker attach centos

```
PS C:\Users\97150> docker attach b2d8884ca787
```

Docker container run -d -p 80:80 -n N1 nginx	p for port- انه يشتغل ع البورت دة و n for name-

```
PS C:\Users\97150> docker container run --detach --publish 80:80 --name n1 nginx
e04cebfc58585e8f6ce[illegible]c571436bf16abb880f480933d11f6788e990ef2f8fab
```



Docker exec -it container-name command Docker run -it image-name command	لو عاوز افتح ال bash ع container معين i = interactive- t = psueod terminal-

```
PS C:\WINDOWS\system32> docker exec -it 7d121a58ef1d cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
```

```
PS C:\Users\97150> docker exec -it n1 bash
root@e04cebfc5858:/# service nginx -V
service ver. 1.60
root@e04cebfc5858:/# service nginx status
nginx is running.
root@e04cebfc5858:/# service nginx exit
Usage: /etc/init.d/nginx {start|stop|status|restart|reload|force-reload|upgrade|configtest|check-reload}
root@e04cebfc5858:/# service nginx stop
```

```
PS C:\WINDOWS\system32> docker run -it centos bash
[root@3b7de2d08fd6 /]# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
```

Docker tag SOURCE_IMAGE[:TAG] TARGET_IMAGE[:TAG]	دة عشان اعمل صورة طبق الاصل من ال image بس باسم مختلف

```
PS C:\Users\97150> docker tag redis naggar/redis
PS C:\Users\97150> docker images
REPOSITORY      TAG       IMAGE ID       CREATED        SIZE
naggar/redis    latest    f1b6973564e9   3 weeks ago    113MB
redis           latest    f1b6973564e9   3 weeks ago    113MB
nginx           latest    c316d5a335a5   3 weeks ago    142MB
fedora          latest    b78af7a83692   2 months ago   153MB
```

- ممكن اغير ال tag بتاعه كمان من خلال انى اباصيله tag فى ال command

```
PS C:\Users\97150> docker tag redis naggar/redis:test
PS C:\Users\97150> docker images
REPOSITORY      TAG       IMAGE ID       CREATED        SIZE
naggar/redis    latest    f1b6973564e9   3 weeks ago    113MB
naggar/redis    test      f1b6973564e9   3 weeks ago    113MB
redis           latest    f1b6973564e9   3 weeks ago    113MB
nginx           latest    c316d5a335a5   3 weeks ago    142MB
fedora          latest    b78af7a83692   2 months ago   153MB
```

- ممكن اكتر من ال image tag يشاوروا ع نفس ال image

Docker run -e env-var=value –name name-value image-name	لو عاوز ادي قيمة ل environment variable بحط -e و باعدين اسم المتغير و باعدين القيمة بتاعته و لو عاوز ادى اسم للل container بحط –name و باعدين الاسم بتاعها

```
$ docker run -p 38282:8080 --name blue-app -e APP_COLOR=blue kodekloud/simple-webapp
```

لو انا عاوز اشوف ال environment variables اللى عندى اصلا بعمل ال command دة :

docker inspect image-id/image-name

```
"Env": [
    "APP_COLOR=pink",
    "PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
    "LANG=C.UTF-8",
    "GPG_KEY=0D96DF4D4110E5C43FBFB17F2D347EA6AA65421D",
    "PYTHON_VERSION=3.6.6",
    "PYTHON_PIP_VERSION=18.1"
],
```

هنا بيظهرلى القيمة الdefault اللى هو واخدها اصلا

Docker run –memory 100m ubuntu	هنا بحددله هاياخد ايه من الميمورى بتاعتة الجهاز

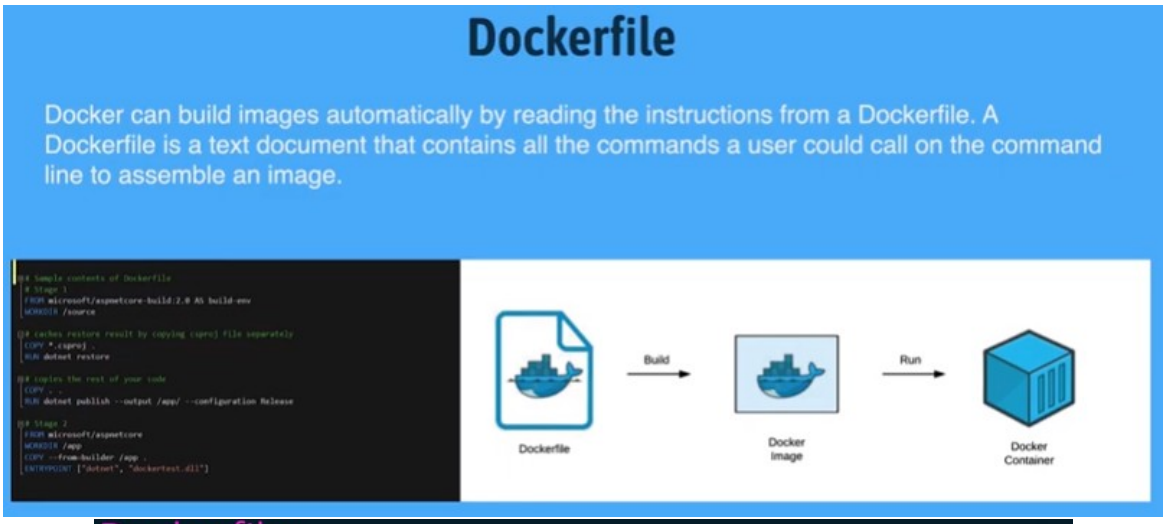
```
docker run --cpus=.5 ubuntu

docker run --memory=100m ubuntu
```

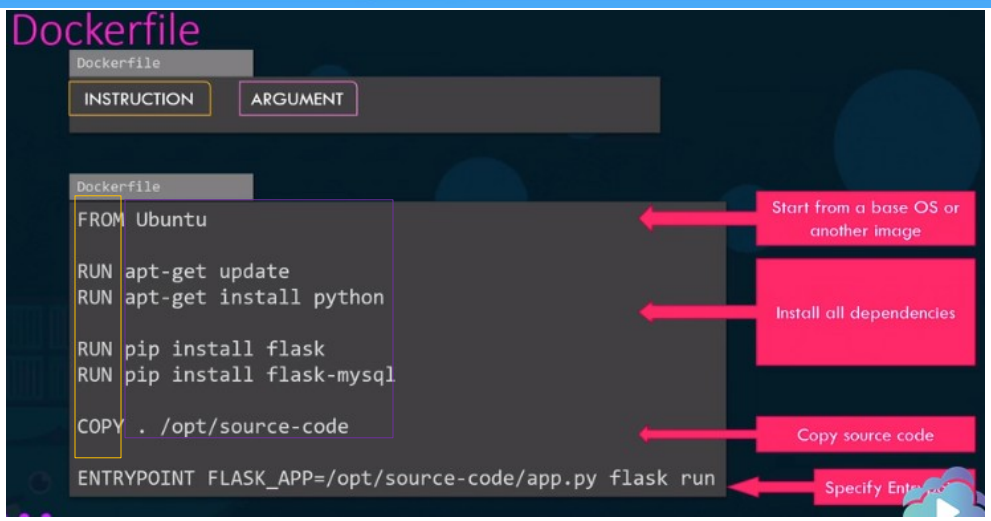
Docker commit container-id new-image-name	بستخدم ال commit عشان لو عملت تغيير ع container عندى زى انى اضيف ملف مثلا او اى حاجة و بعمل commit بعدها بيروح يعملى image جديدة من ال image اللى كان قايم منها ال container ويضيف فيها التعديلات اللى انا عملتها فى ال container

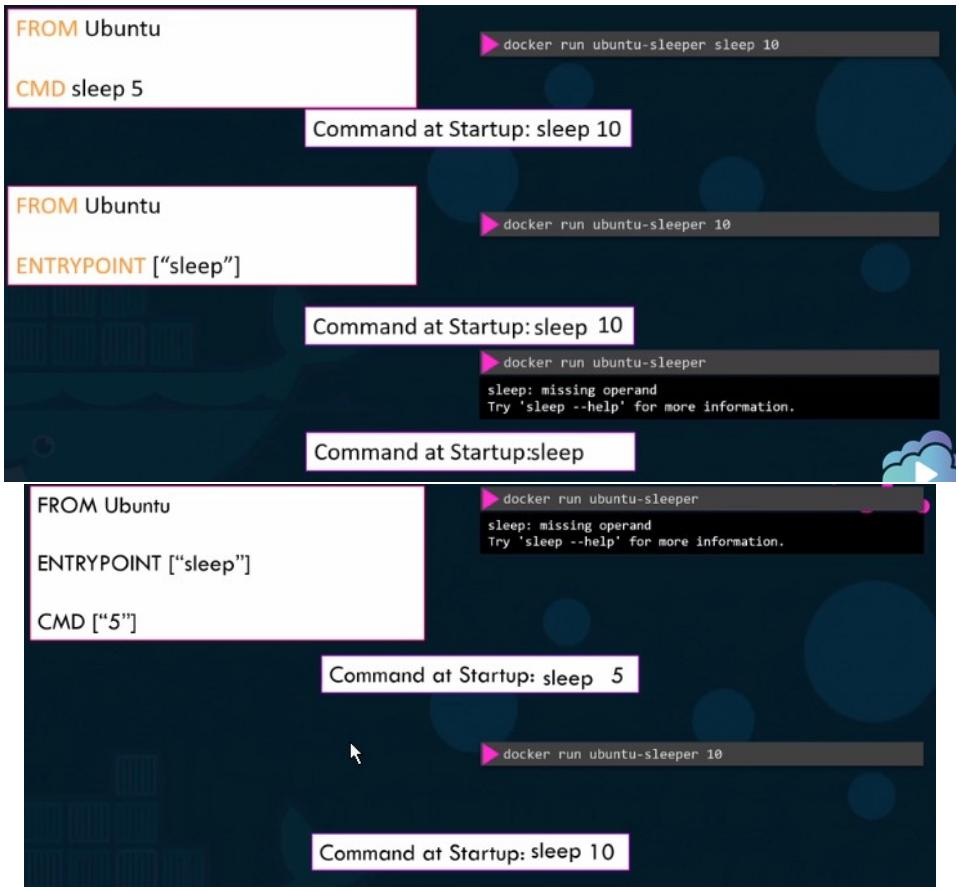
```
mostafa-alnaggar:~$ docker commit 173 ubuntu:new-ubuntu
sha256:ef6efd0dd39009a0fabe53f823a6c538d01f9a3c53383d7181b3e6aede4439e2
mostafa-alnaggar:~$ docker run -it ubuntu:new-ubuntu
root@b8b251786cb5:/# ls
bin  boot  dev  etc  file  home  lib  lib32  lib64  libx32  media  mnt  opt  proc  root  run  sbin  srv  sys  tmp  usr  var
```

انا هنا عملت image جديدة و قومت منها container ف ظهر فيه الملف اللى كنت عامله ع ال container القديم

Docker File :


Dockerfile
Docker can build images automatically by reading the instructions from a Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image.
Build
Run
Dockerfile
Docker Image
Docker Container


Dockerfile
Dockerfile
INSTRUCTION
ARGUMENT
Dockerfile
FROM Ubuntu
RUN apt-get update
RUN apt-get install python
RUN pip install flask
RUN pip install flask-mysql
COPY . /opt/source-code
ENTRYPOINT FLASK_APP=/opt/source-code/app.py flask run
Start from a base OS or another image
Install all dependencies
Copy source code



Commands:

لو عاوز اعمل image عندى اول حاجة بعمل file بكتب فيه ال details بتاعة ال image

```
FROM  alpine
CMD [ "echo","hello world custom image" ]
```

From وبكتب بعدها اسم ال OS اللى ال image هاتقوم عليه
CMD بكتب جواها ال action اللى هانفذه فى اول argument بكتب ال command و الargument التانى بكتب ال argument بتاعة ال command نفسه
لما اجى اعملها run هاتطلع بالشكل دة

```
PS V:\DockerFiles> docker run 52878d43d87e
hello world custom image
```

Docker build --tag path	عشان اعمل build لل image بتاعتى و بحددلها المكان فى ال path

```
PS V:\DockerFiles> docker build .
[+] Building 9.3s (6/6) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 91B
 => [internal] load .dockerignore
 => [internal] load metadata for docker.io/library/alpine:latest
```

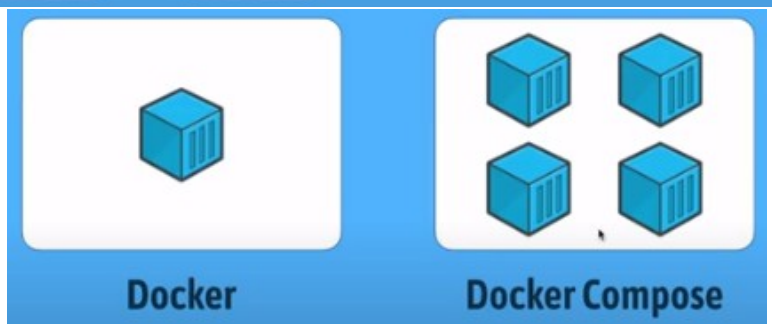
و عشان اتأكد انها اتعملت بروح اعمل image ls

```
PS V:\DockerFiles> docker image ls
REPOSITORY                  TAG       IMAGE ID       CREATED         SIZE
<none>                      <none>    52878d43d87e   4 days ago      5.57MB
```

عشان ادي لل image بتاعى اسم tag--

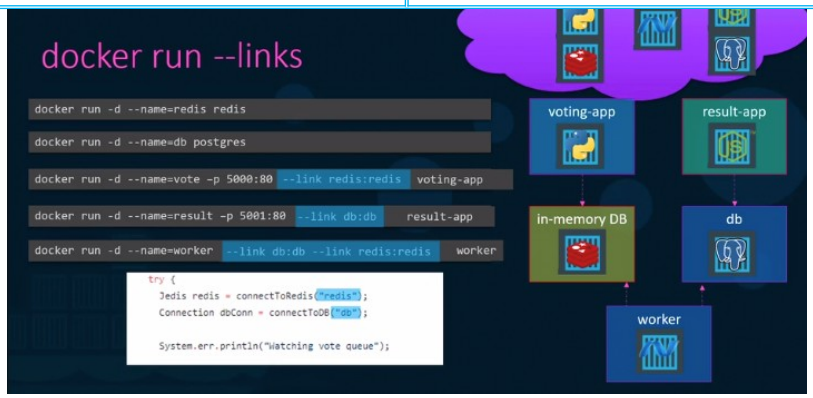
```
PS V:\DockerFiles> docker build --tag new-image .
new-image                   latest    c394f99b4577   4 days ago      5.57MB
```

Docker Compose:

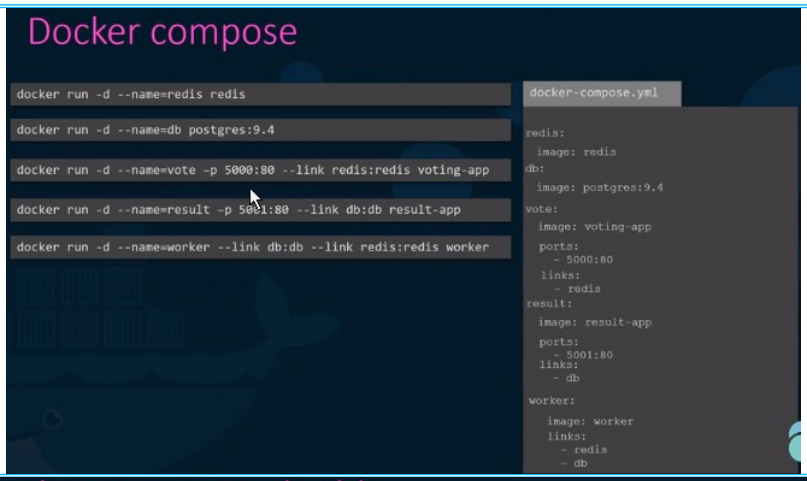


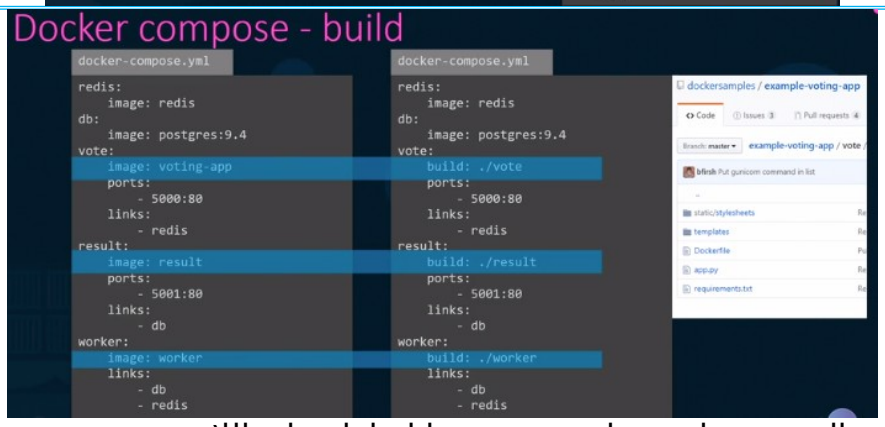
Commands:

Docker run –link db:db	بنستخدمه عشان نربط كذا container ببعض من خلال انه بديله اسم ال container اللى عاوز اعمله connect



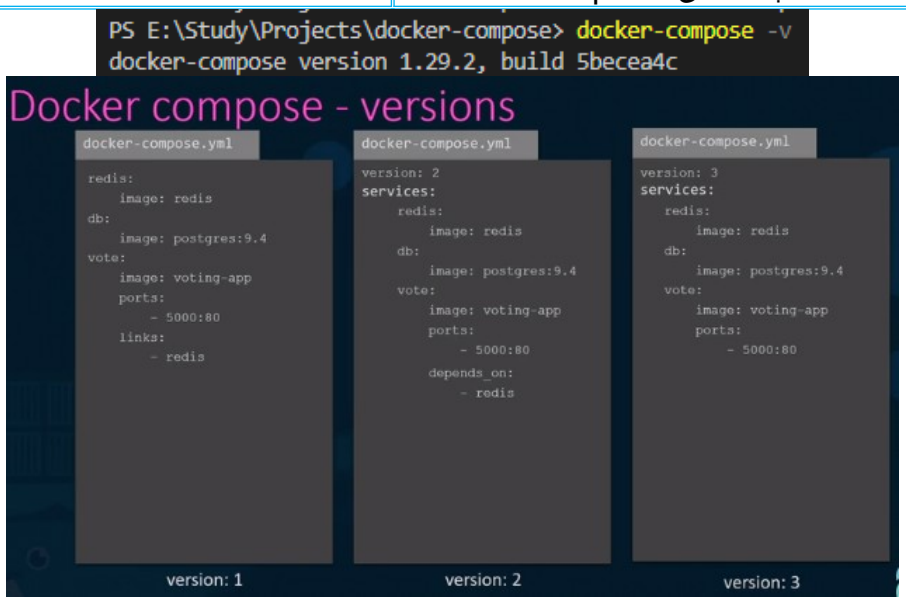
هنلاحظ ان هنا عاملين run لكل حاجة لوحدها و الموضوع رخم ف ال docker compose حل الموضوع دة من خلال اننا بنحط كل ال images اللى احنا عاوزينها فى ملف واحد بس و بيكون اسمه docker-compose.yml



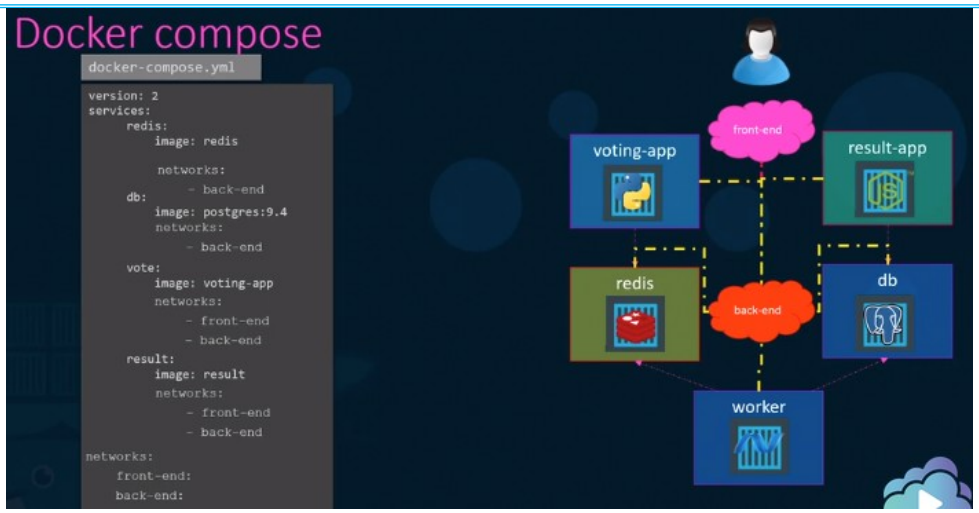


لو ال image بتاعة ال app بتاعى عبارة عن app انا عامله على اللاب عندى بروح مغير ال image ل build و اديله مكان ال app ف هو قبل ما يعمل ال compose هايروح يعمل build لل images بتاعة ال app و باعدين يعمل ال compose

عشان يرجعلى ال version بتاعة ال compose	Docker-compose -v

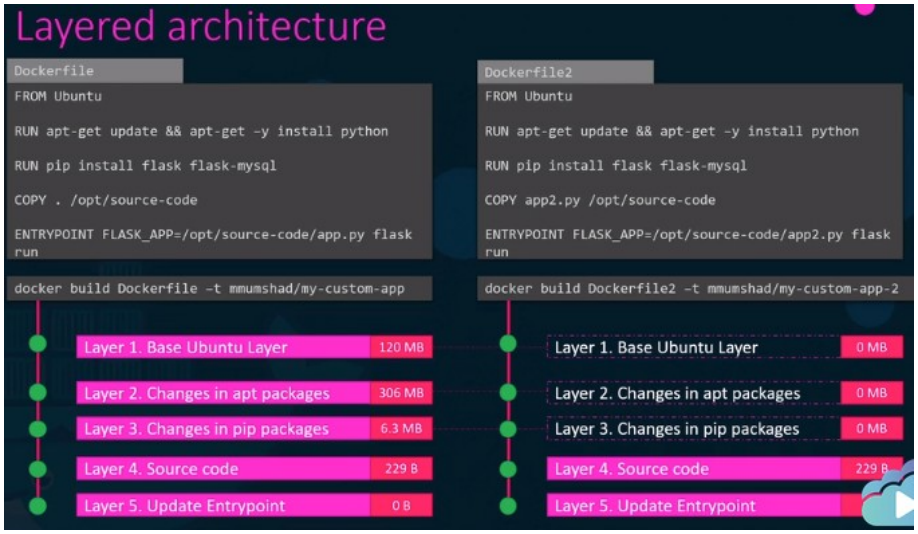


- Version 1 : ماكنتش بحط رقم ال version ف اول الملف و كان لازم بحط links ما بين ال container اللى عاوز اربطها ببعض
- Version 2 : لازم نعرف ال version و بعدها بنحط كل ال containers اللى عندى تحت ال services و مش محتاج انى احط links لانه هو بيعمل network bridge و بيوصل كل ال containers ببعضها اصلا و لو عاوز image معينة يتعملها run الاول و بعدها ابنى ع اساسها باقى ال images بحط فى الاخر depends-on و اديله اسم ال container
- version 3 : زى 2



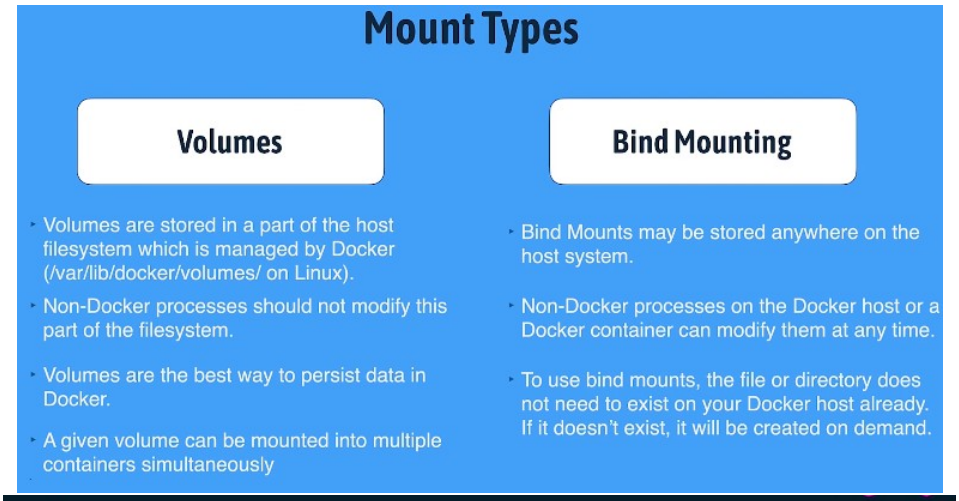
ممكن اقسم ال app لاكتر من network زى ما متوضح كدة قدامك ف ساعتها بعرف فى ملف ال compose جزء ال networks و بدى لكل container ال network اللى يشتغل عليها .

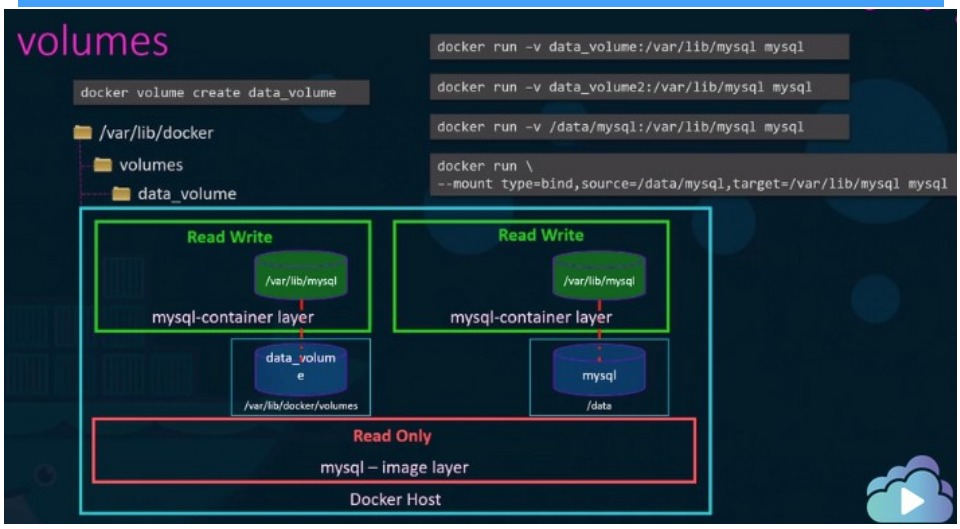
Note:



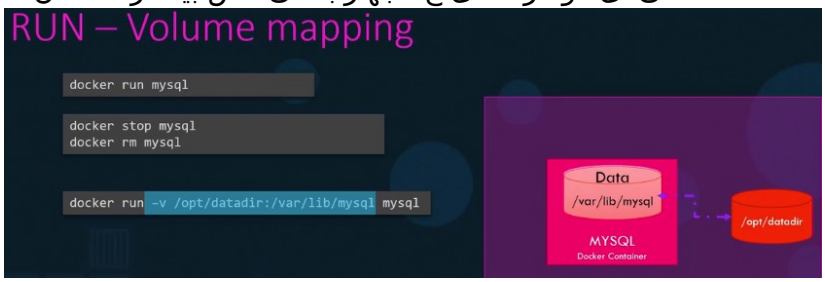
- لو انا عندى docker file 2 هايقوموا ع نفس ال os و نفس ال packages ال docker بي build ال image الاولانية عادى بس فى التانية بدل ما ينزل image , packages من تانى بياخدهم من ال image الاولى و بكدة بيقدر يعمل build لل image التانية بسهولة و بيكون حجمها اخف

Volumes:





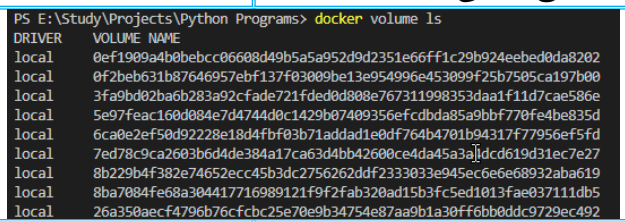
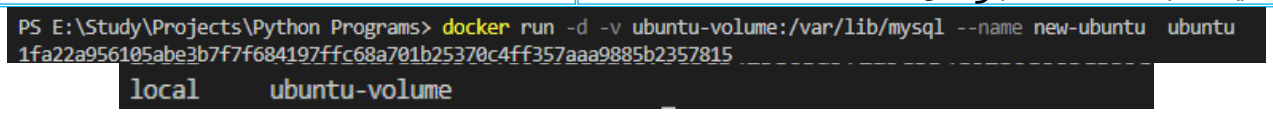
- لما اجى اعمل run image بيتعمل لل container حاجة يخزن فيها و اللى هى ال volume
- Bind mount بعمله mount على اى فولدر عندى ع الجهاز بتاعى مش بيشترط مكان معين زى ال volume



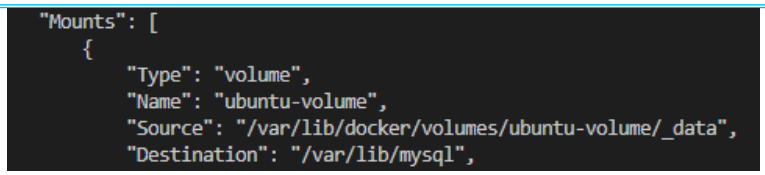
-

-

:Commands

Docker volume ls	عشان اشوف كل ال volumes اللى عندى
	
Docker run -v myvolume:/path image-name	دة كدة انا بعمل volume اسمه myvolume ل container معين ال path دة المكان اللى جوة ال container اللى لما اعمل فيه حاجة هاتتحفظ جوة ال volume
	

و لو عملنا docker inspect هايظهرلنا فى ال info اللى بترجع منه

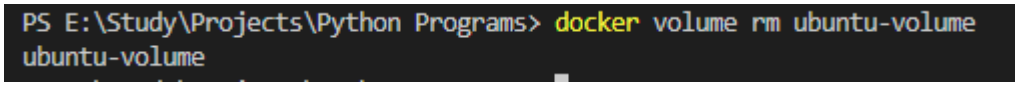


docker run -v volume:/tmp/ -it ubuntu

هنا بعمل ال volume على tmp directory

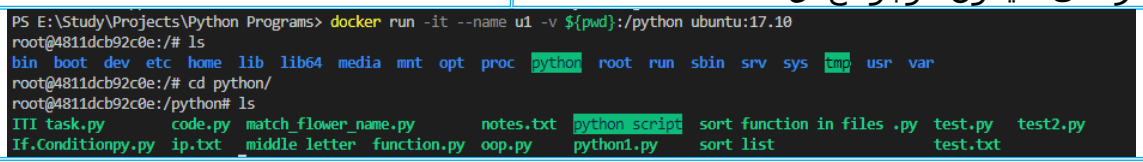
Docker volume create volume-name	عشان اعمل volume جديد

Docker volume rm volume-name

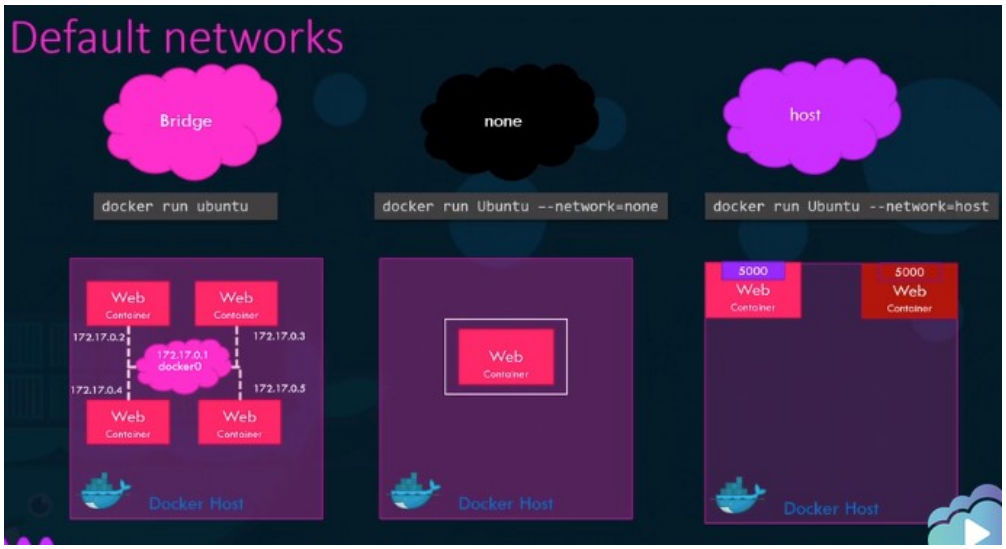


عشان امسح volume

Docker run -v path:/folder-name image-name	دة عشان اعمل bind mount بديله ال path و بعديها اسم الفولدر اللى هايكون موجود ع ال container



Docker Network:

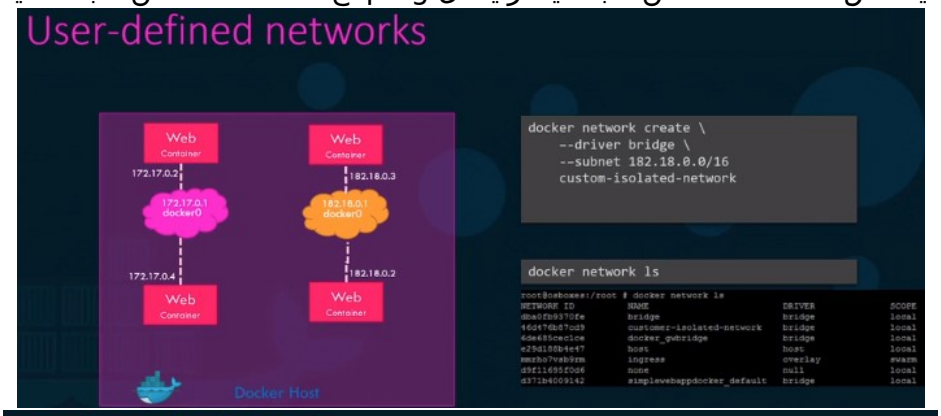


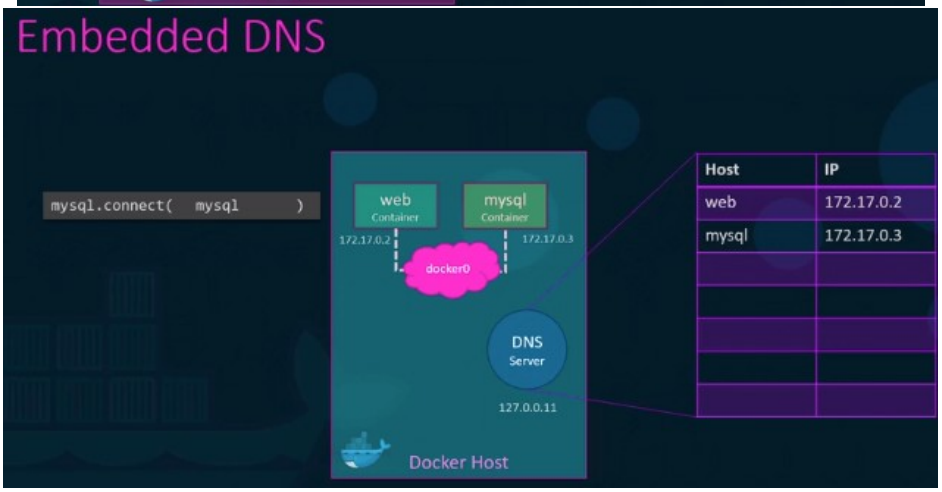
- None : بفصل ال network عن ال container اللى شغال ما بيكونش فى اى حد شايفه خاااااالص

Host : و دة بيستخدم ال network بتاعتة ال host نفسه ما بقدرش اقوم عليه اكتر من container يستخدموا نفس ال port

- Bridge : ال docker هو اللى بيعمله و بيوصل ال containers كلها ببعضها من خلال انه بيعمل شبكة داخلية توصلهم كلهم ببعض و ممكن اقسم الشبكة دى لاكتر من شبكة بس كل شبكة بتبقى منفصلة عن التانية يعنى

ماينفعش container من شبكة يقدر يعمل ping ع container من شبكة تانية





- لو عاوز اخلى container يقدر يعمل access ع container تانى بديله اسم ال container و كدة كدة بيكون فى dns table ال docker بيعمله بيحط فيه اسامى ال containers مع ال ip الخاص بكل container

:Commands

Docker network ls	عشان يعرضلى كل الشبكات اللى عندى
PS E:\Study\Projects\Python Programs> docker network ls NETWORK ID NAME DRIVER SCOPE fb2a05f833b5 bridge bridge local ad180189c095 example-voting-app_back-tier bridge local 09a558674d4e example-voting-app_front-tier bridge local e07126b27ad9 host host local e571d2631cf6 none null local	
Docker run –network network-type image-name	run container based on specific l عشان network
PS E:\Study\Projects\Python Programs> docker run --network B2 -d redis $ docker run --network=core --name alpine-2 alpine	
Docker network create –driver bridge bridge-name	دة عشان اعمل bridge جديد
PS E:\Study\Projects\Python Programs> docker network create --driver bridge B1 2261f4ff35cf28bdb492a59f99f0f07be820c188f087dd70a6a832f5a9a1993d PS E:\Study\Projects\Python Programs> docker network ls NETWORK ID NAME DRIVER SCOPE 2261f4ff35cf B1 bridge local	
عشان اعرض العملومات عنها docker network inspect B1	

```
PS E:\Study\Projects\Python Programs> docker network inspect B1
[
    {
        "Name": "B1",
        "Id": "2261f4ff35cf28bdb492a59f99f0f07be820c188f087dd70a6a832f5a9a1998d",
        "Created": "2022-04-26T21:50:40.1700535Z",
        "Scope": "local",
        "Driver": "bridge",
        "EnableIPv6": false,
        "IPAM": {
            "Driver": "default",
            "Options": {},
            "Config": [
                {
                    "Subnet": "172.20.0.0/16",
                    "Gateway": "172.20.0.1"
                }
            ]
```

Docker network create –driver bridge –subnet 172.1.1.0/24 bridge-name	عشان اديله ip معين

```
PS E:\Study\Projects\Python Programs> docker network create --driver bridge --subnet 172.33.33.0/16 B2
bcac694eb629374fe24b89997051ba337fc5dad29b09f61709379b2377220364
"Containers": {
    "968fe171c6a6083830a2439a0ff93cf7fc61e5813454b47b6f7a46d67534a584": {
        "Name": "dazzling_wozniak",
        "EndpointID": "1bece3cee2ec7e25eee97c605132a51b7dda715b2101936c59aa8f5e2ed2901f",
        "MacAddress": "02:42:ac:21:00:02",
        "IPv4Address": "172.33.0.2/16",
        "IPv6Address": ""
    }
```

Docker network disconnect bridge-name container-id	دة عشان افصل ال container من ال bridge اللى انا هاحددهاله network

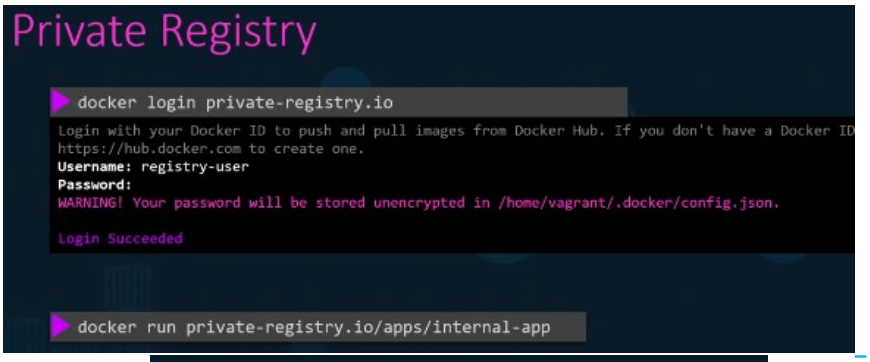
```
PS E:\Study\Projects\Python Programs> docker network disconnect B2 968
```

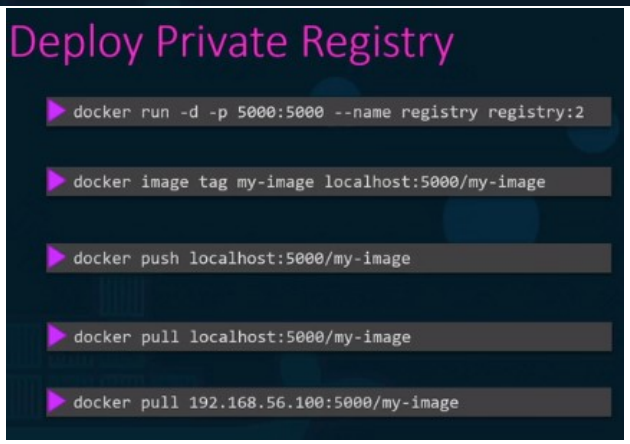
لو حبيت اعمله connect on other bridge

Docker network connect bridge-name container-id

```
PS E:\Study\Projects\Python Programs> docker network connect B1 968
```

Docker Registry:





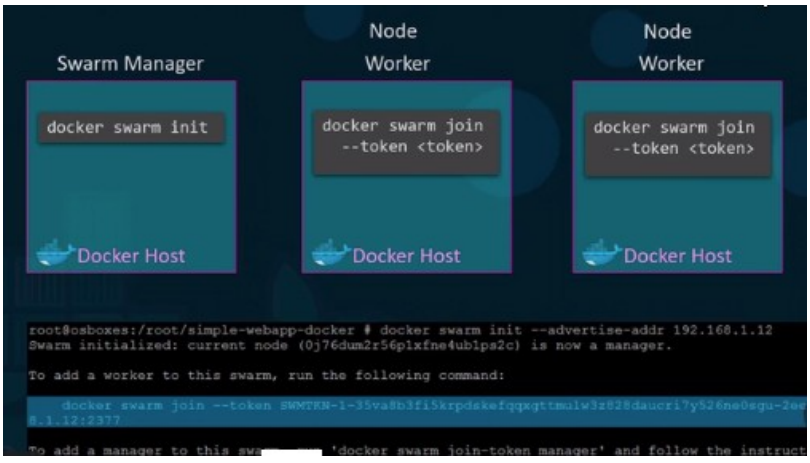
Docker Orchestration:

بتقدر تعملى mange لعدد كبير من ال containers بحيث ان لو فيه container حصله failure بتقوملى واحد تانى مكانه

و من امثلة ال docker orchestration : docker swarm – Kubernetes – Mesos

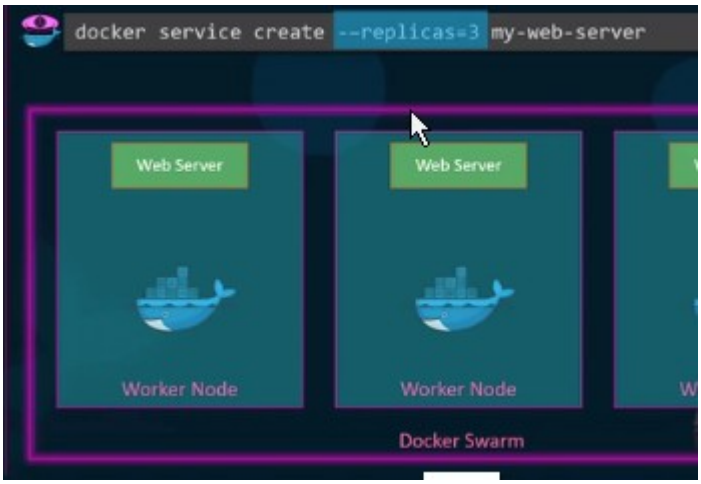
:Docker Swarm

لو عندى app بيقومه على اكتر من host بحيث انه يضمن ال high availability و كمان بيعمل load balance بينهم



بيكون عندى swarm manager ع host معين عن طريق انى بعمل الcommand دة : docker swarm init

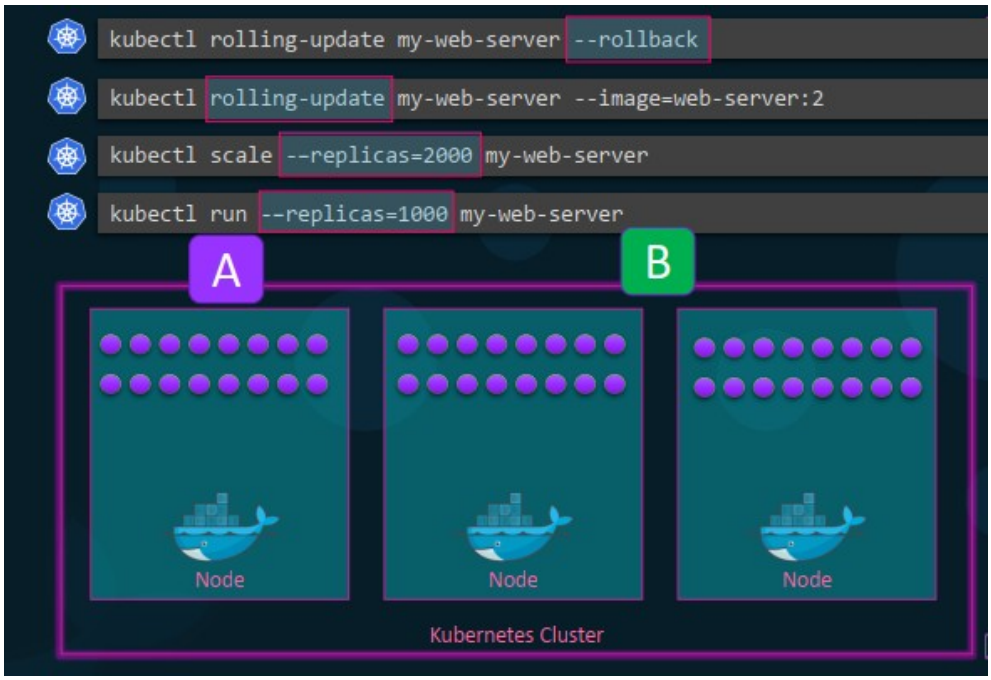
و بشغل معاه اكتر من node worker تانى عن طريق ال command دة : docker swarm <join –token <token



عشان اقوم بقى containers على اكتر من node بستخدم ال docker service command
من خلال انى برن ال command دة على ال swarm manager :
docker services create –replicas=<num of replicas> image-name

و ال docker service command دة زيه زى ال docker run command بياخد نفس ال
network , port , environment variable زى ال option

:Kubernetes



بقدر فيها manage l ال replica بتاعتى من خلال انى بحدد عاوز كام container من خلال ال command دة :
kubectl run -–replicas=1000 my-web-server

بس دة بيكون على **Kubernetes cli**

ممكن ازود عدد ال replica من خلال ال command دة :
kubectl scale -–replicas=2000 my-web-server

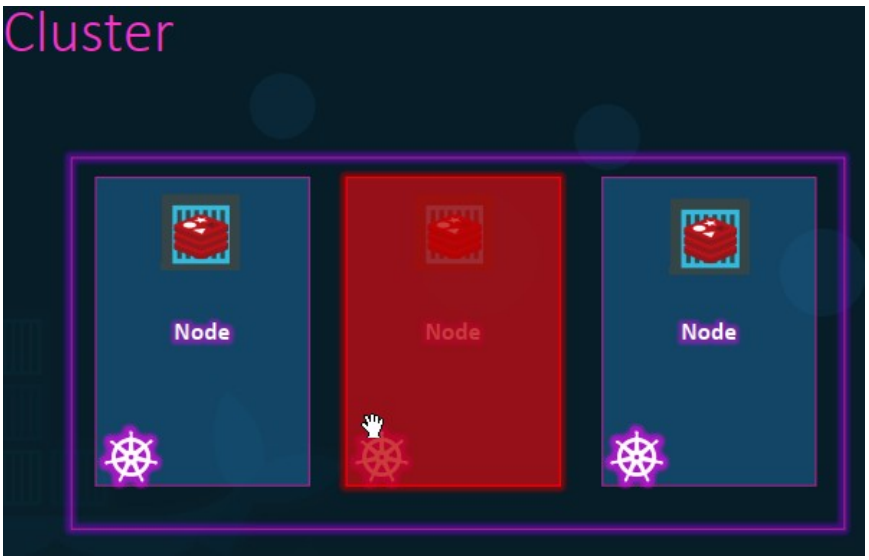
و لو عاوز اغير ال image اللى قايم منها ال containers بستخدم ال command دة :

kubectl rolling-update my-web-server --image=web-server:2

و لو حبيت ارجع لل image القديمة بعمل rollback من خلال ال command دة :

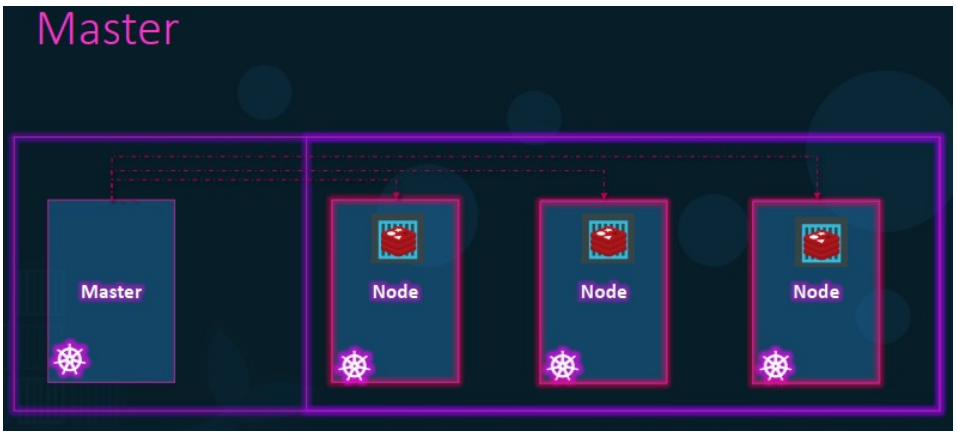
kubectl rolling-update my-web-server –rollback

ال app بيقوم على node طب لو حصل failure فى ال node دى طبيعى ان ال app هايقع هو كمان ف بيستخدم حاجة اسمها cluster



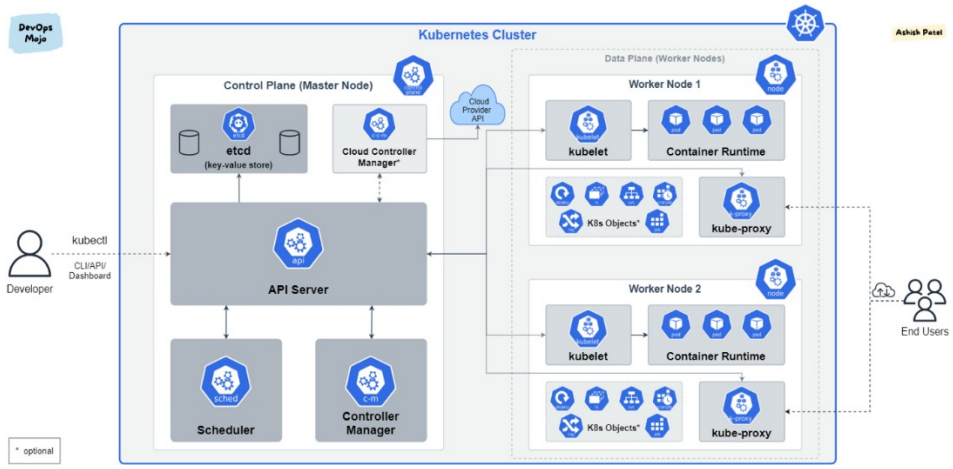
بحيث انى اقوم ال app بتاعى ع اكتر من node عشان لو فى node حصلها failure ال app نفسه ما يوقعش

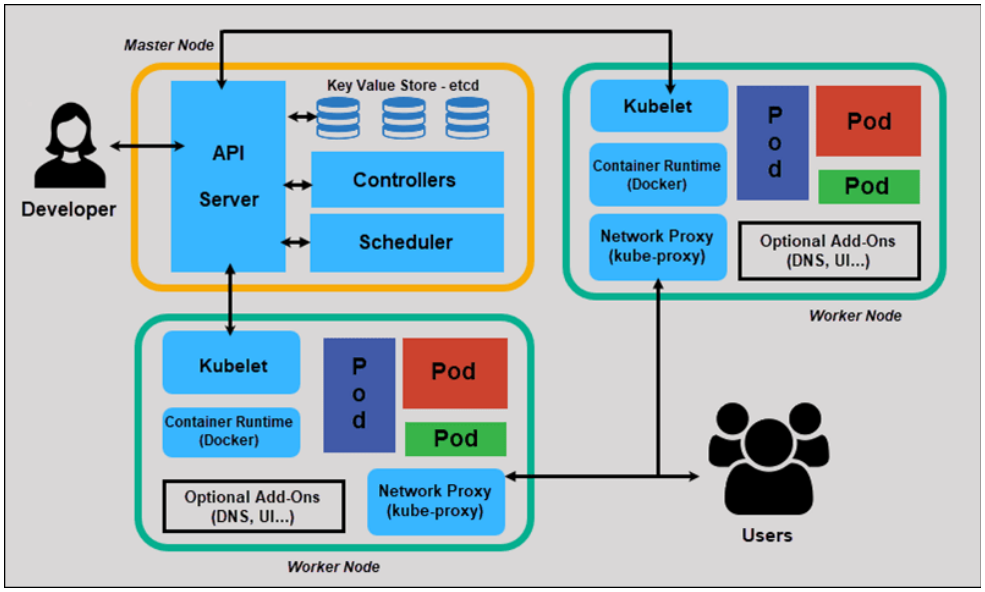
طب هو مين بيعمل manage لل nodes دى ؟

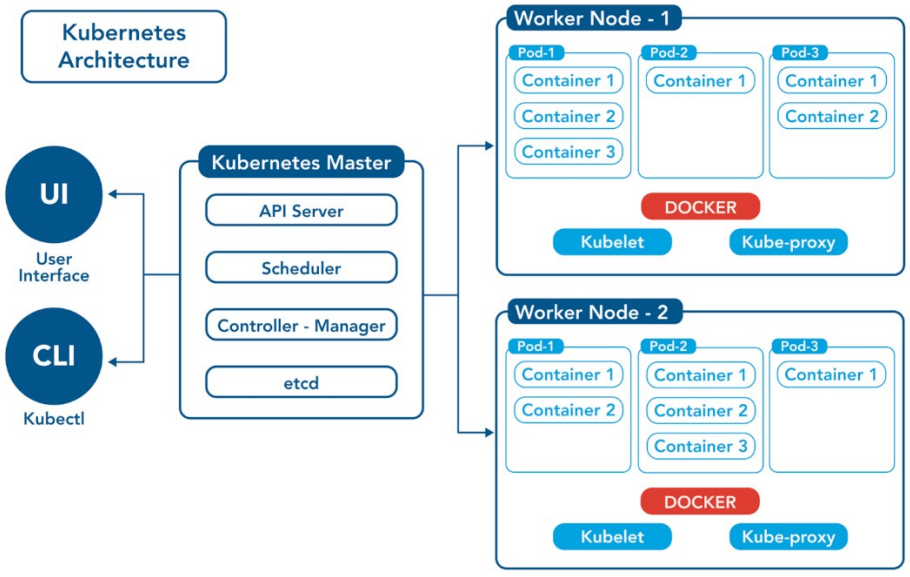


ال master هو اللى بيكون مسؤل عنهم

:Components







المكونات دى كلها عبارة عن containers شغالين جوة ال master بتاع ال Kubernetes

Etcd : key value store بيخزن الداتا الخاصة بال cluster بيحث ان ما يحصلش conflict بين ال masters

API Serve: هو اللى بيكون مسؤل عن كل ال communications بين ال master and workers او بين ال master and any third party tool هايحصلها integration with Kubernetes , دة بيكون زى ال api gateway كدة

Controller: دة عقل ال Kubernetes بيكون مسئول عن عملية cluster creation و برده عملية ال join بتاعة ال cluster لل master cluster هو اللى بياخد ال info اللى فى ال yml file و يترجمها و ينفذ اللى فيها و دة اهم مكون فيهم كلهم و بيقدر يعمل communicate مع اللى حوليه من خلال api server

Scheduler: بيكون مسئول عن توزيع ال containers على ال worker nodes اللى موجودة عندى فى ال cluster

Docker runtime: عشان الحاجات دى كلها تشتغل عندى لازم يكون عندى حاجة بتشغل ال containers دى زى ال docker

الحاجات دى كلها مكونات ال master بتاعى عشان اقدر اشغل Kubernetes و يفضل لو عندى app اشغله على worker node مش على ال master node

طب ايه المهم فى ال worker node عشان يشتغل من غير مشاكل ؟

Kubelet: دة اللى بيكون مسئول عن ال communication بين ال "master worker node and Kubernetes" و دة بيحصل من خلال ان ال kubelet يكلم ال api server اللى فى ال master nodeو ال kubelet لو حصل فى اي مشكلة ال connection كله بيفصل فبالتالى ال worker node هاتتفصل عن ال master

Proxy: و دة بيكون مسئول عن ال communication اللى بين ال containers اللى قايم على ال worker node و الانترنت