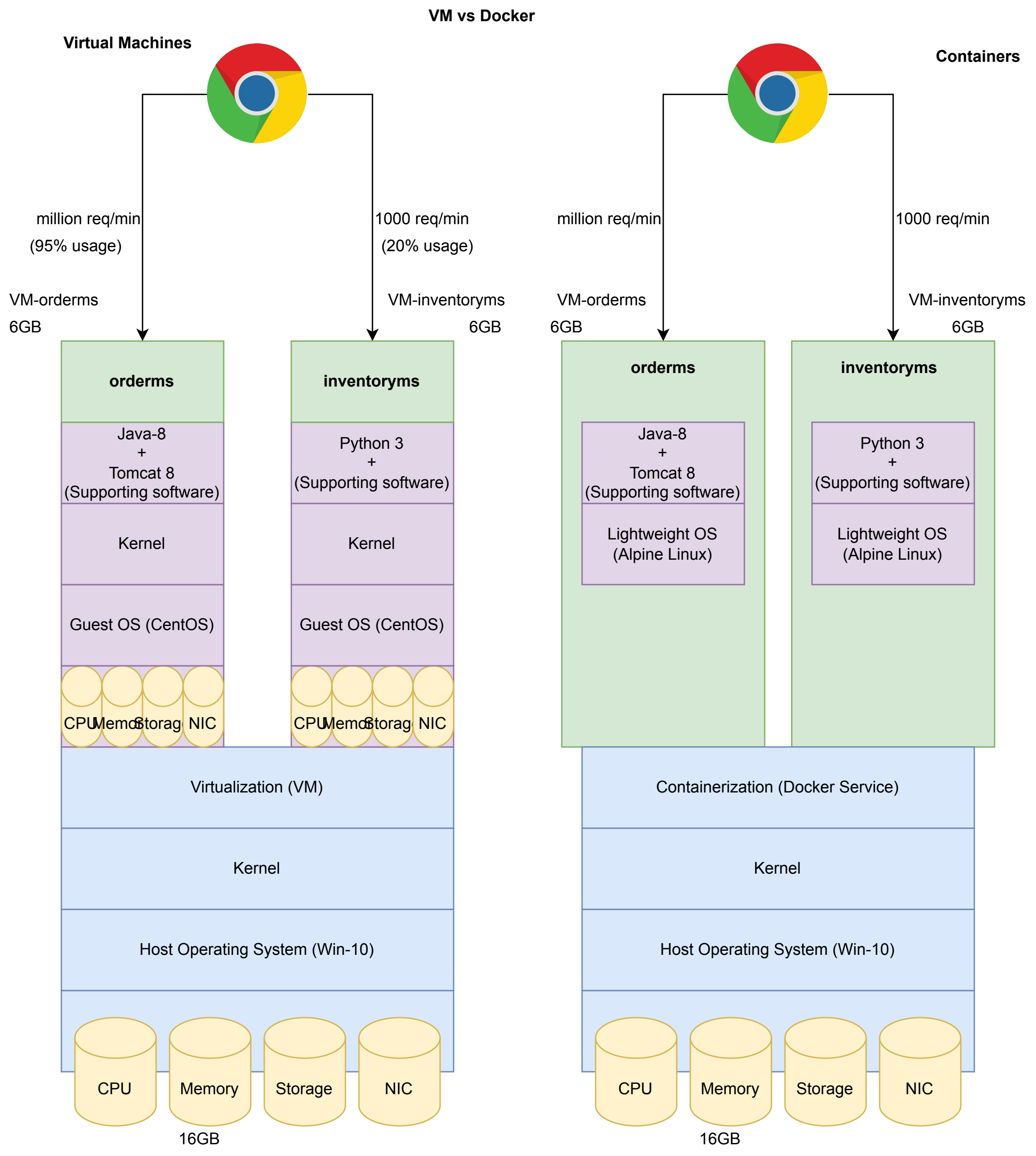
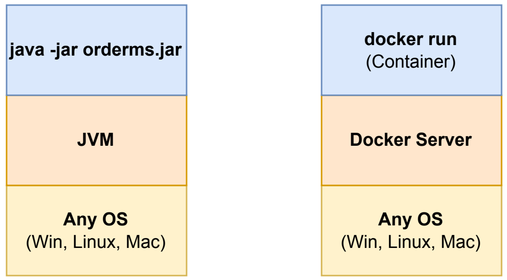


Host Machine:
orderms: http://localhost:8082/orders
inventoryms: http://localhost:8083/inventories

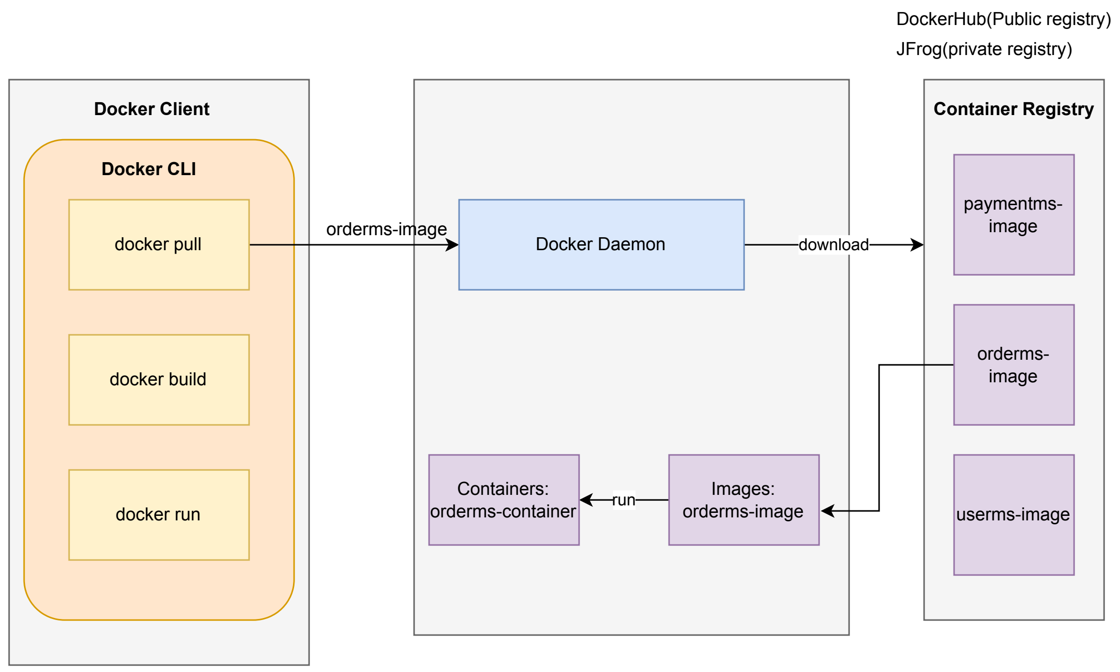
Issues with VM:

1. Resource sharing not possible
2. Software version mismatch
3. Startup time

VM: jar file -> java -jar orderms.jar
Container: [jar+Java-8+Tomcat-8+AlpineLinux] = orderms-docker-image



Docker(Client-Server)



docker-image == jar file (orderms.jar)
docker-container (docker run orderms-image) == java -jar orderms.jar

mvn install/package == docker build -t orderms:1.0
java -jar orderms.jar == docker run orderms:1.0

Docker Image

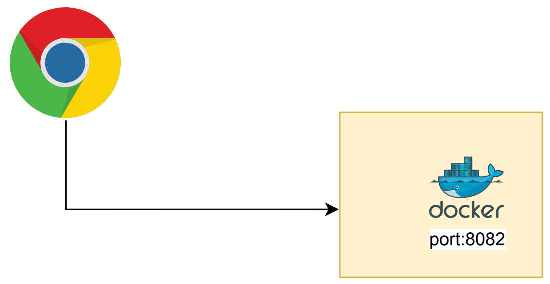
```
FROM base-image
RUN <apt-get install opnejdk-11>
COPY userms-0.0.1-SNAPSHOT.jar /app.jar
```

Dockerfile:

```
FROM openjdk:8-jdk-alpine
COPY userms.jar /app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

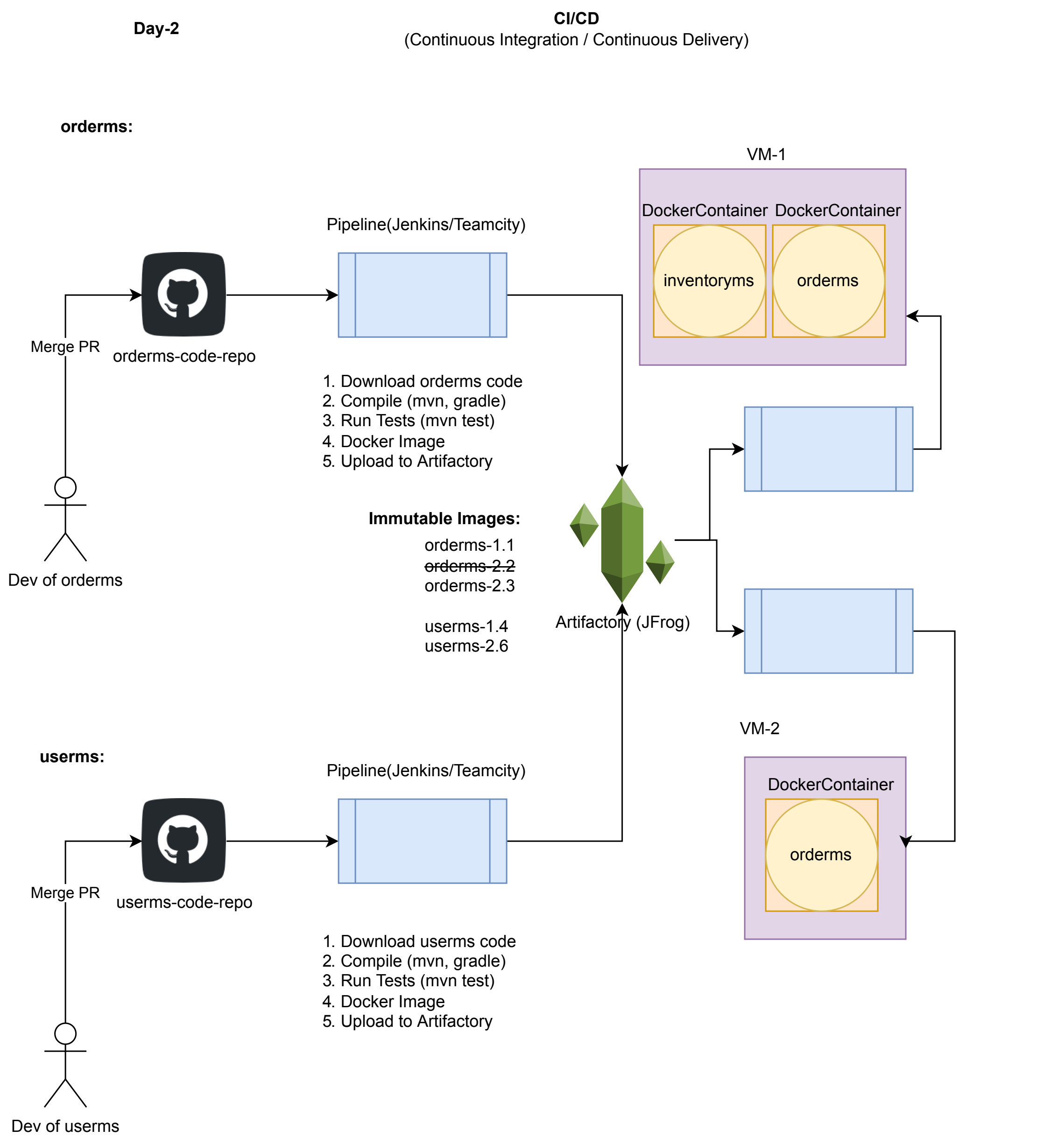
```
FROM openjdk:11-jdk-slim
COPY userms.jar /app.jar
ENTRYPOINT ["java", "-jar", "/app.jar"]
```

Port Forwarding



ubuntu - **docker**:

8081	-
8082	- 8082
8083	- 8082



1. Download orderms code
2. Compile (mvn, gradle)
3. Run Tests (mvn test)
4. Docker Image
5. Upload to Artifactory

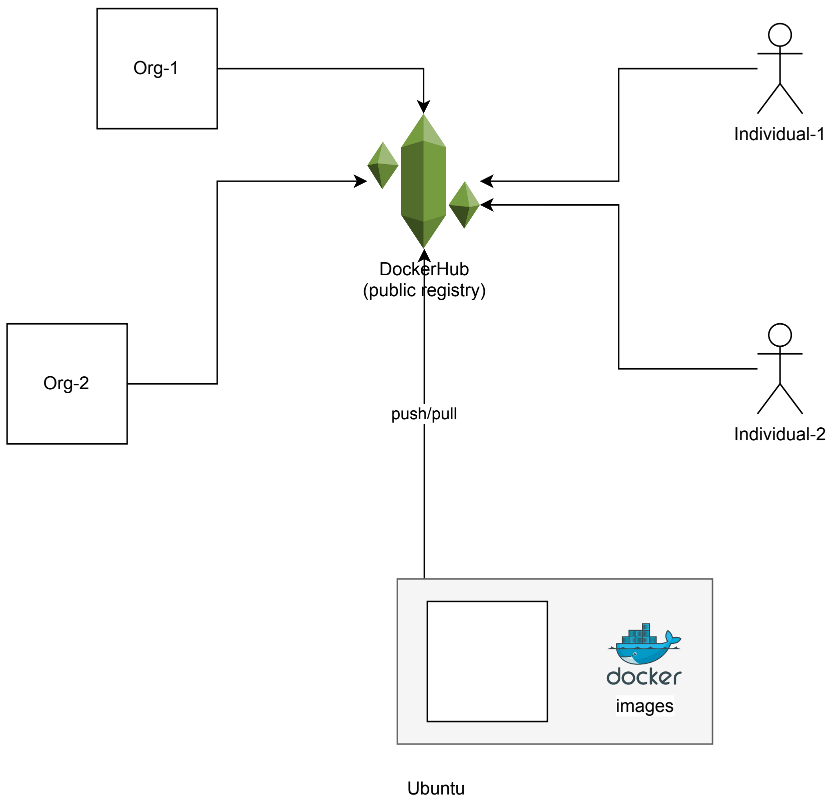
Immutable Images:

orderms-1.1
~~orderms-2.2~~
orderms-2.3

userms-1.4
userms-2.6

1. Download userms code
2. Compile (mvn, gradle)
3. Run Tests (mvn test)
4. Docker Image
5. Upload to Artifactory

DockerHub



```
REPOSITORY                 TAG            IMAGE ID       CREATED          SIZE
orderms                     1.0           cfdf1d1f9370   39 minutes ago   502MB
misbaharchitect/firstrepo   orderms-3.3   cfdf1d1f9370   22 hours ago     482MB
```

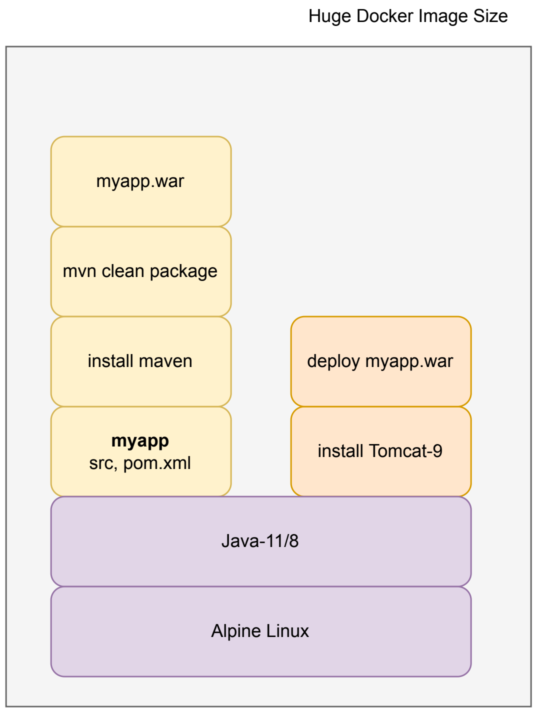
```
docker tag orderms:1.0 misbaharchitect/firstrepo:orderms-3.3
docker login
docker push misbaharchitect/firstrepo:orderms-3.3

docker pull misbaharchitect/firstrepo:orderms-3.3
```

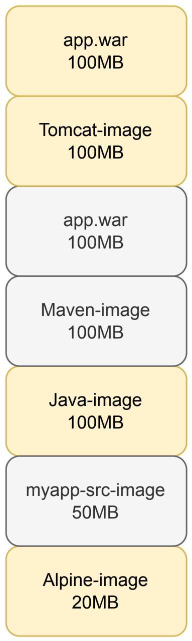
Day-3 **Multistage Build**

Normal Build:

Docker Instructions to build Image in Dockerfile:

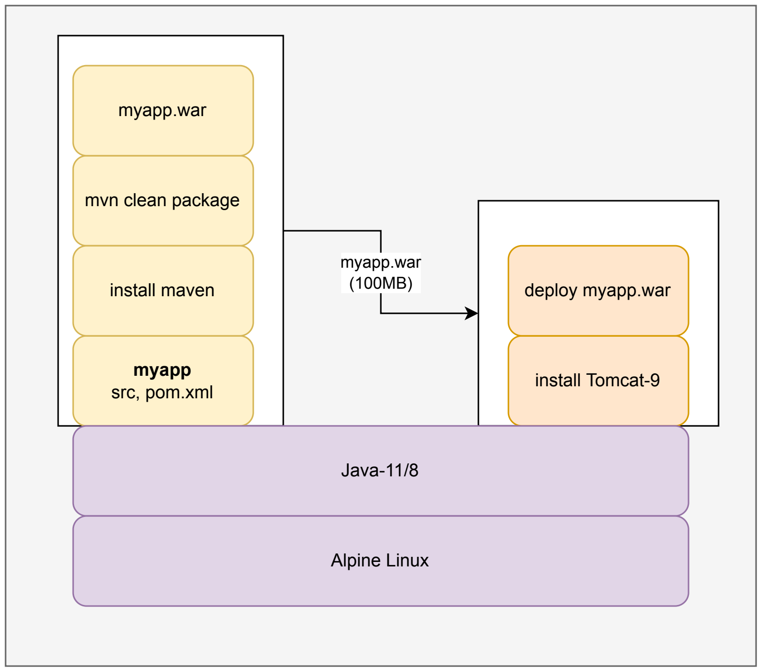
1. From alipne:latest
2. apt-get install jdk-11-slim
3. Copy myapp .
4. apt-get install maven-3.6.3
5. Run mvn clean package
6. apt-get install tomcat-9
7. Entrypont ["catalina.sh", "run", "target\myapp.war"]





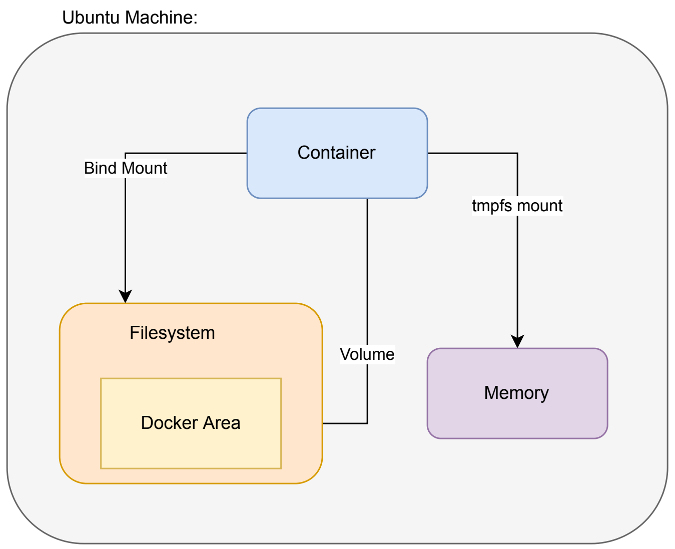
Multistage Build

Each stage has its own image. Only files relevant from the previous build is copied to the subsequent builds

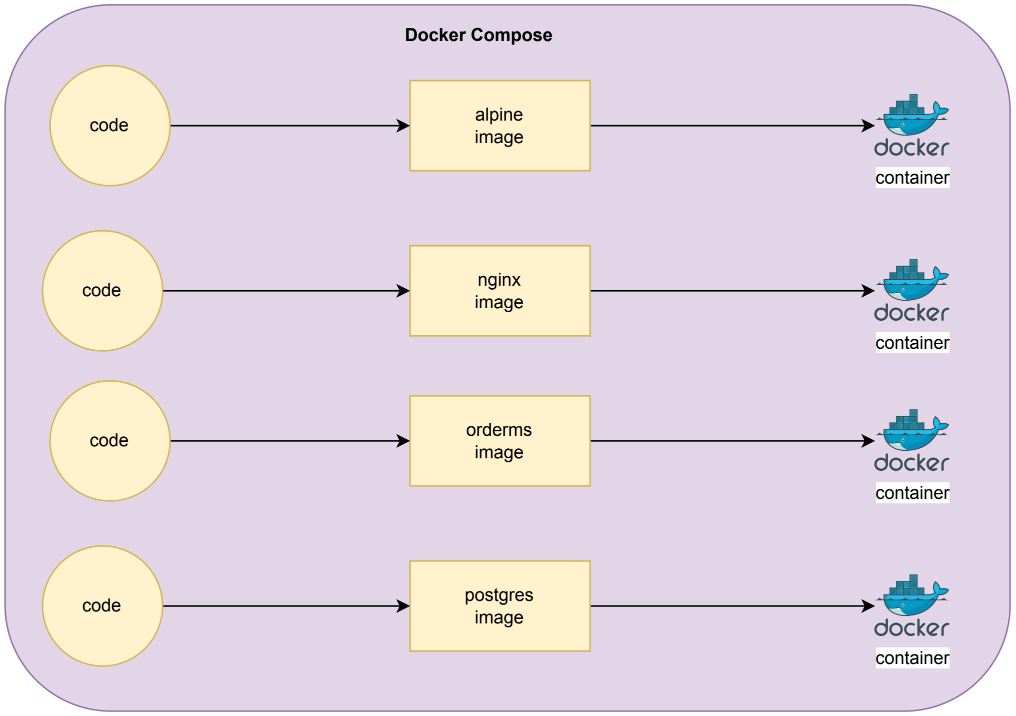


Docker Volumes

/home/ubuntu/target -> container-filesystem



Day-3 **Docker Compose**



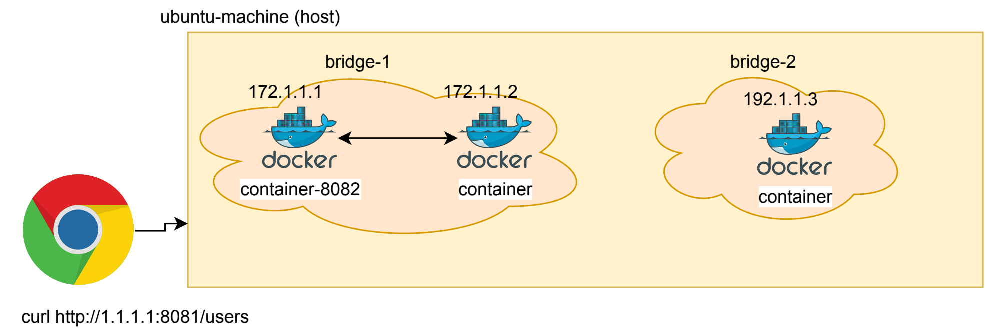
docker-compose.yml

```
docker-compose config
docker-compose config -q
docker-compose build
docker-compose up
docker-compose down
docker-compose start
docker-compose stop
docker-compose ls -a
docker-compose ps
```

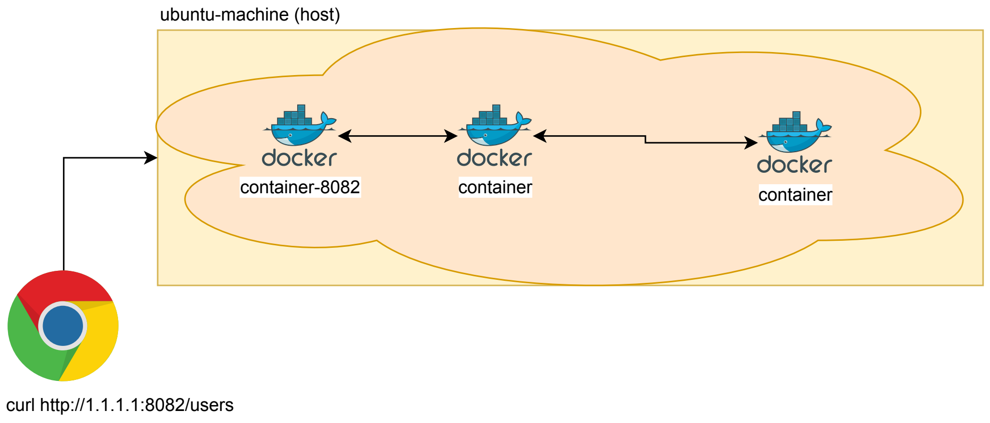
Docker Networks

1. Bridge
2. Host
3. Overlay
4. None

1. Bridge (Single House Networking): It's the detault Docker Network.
docker run -p 8081:8082 my-image
curl localhost:8081/users



2. Host Network: (No Port mapping required)
It uses host machine network/port
docker run --network host my-image
curl localhost:8082/users



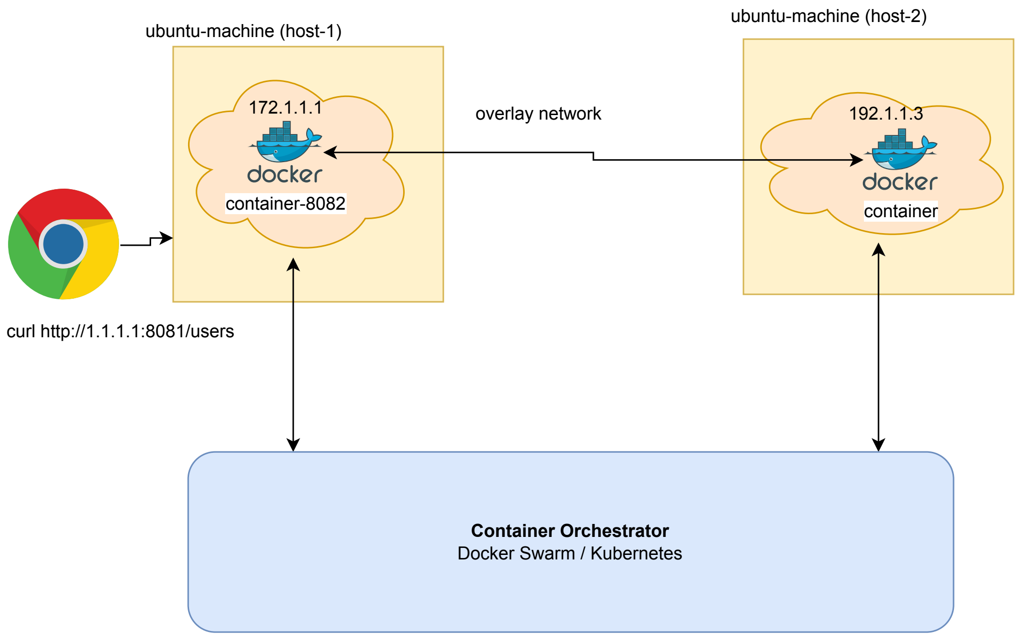
Docker Networks

1. Bridge
2. Host
3. Overlay
4. None

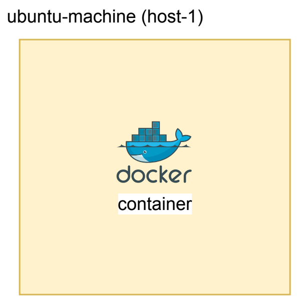
3. Overlay

```
docker network create -d overlay my-overlay-network
```

```
docker run -p 8081:8082 --network my-overlay-network my-image
curl localhost:8081/users
```



4. None Network: It disables the Docker Network

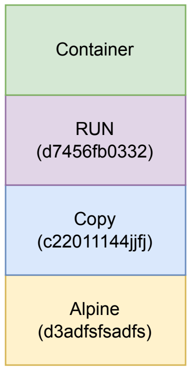


CoW
(copy-on-write)

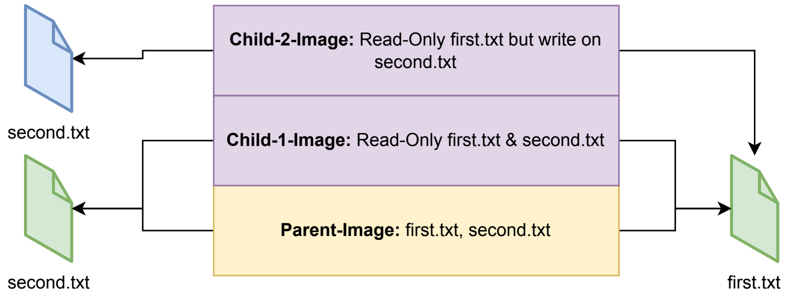
Docker Image Layering:

Dockerfile:

```
FROM alpine:latest
LABEL author=john doe
COPY src /app
RUN rm -r $HOME/cache
ENTRYPOINT ["start"]
```



Copy-On-Write


Child-2-Image: Read-Only first.txt but write on second.txt
Child-1-Image: Read-Only first.txt & second.txt
Parent-Image: first.txt, second.txt
second.txt
second.txt
first.txt