Docker:

Diagram

Description automatically generated

* Devs create a docker image, so testers don’t have to manually install the packages and dependencies, just run the image
* A container is a running version of an image
* docker –version to see the version of docker installed
* docker [container] run -p 5000:5000 in28min/hello-world-java:0.0.1.RELEASE
* ^^^To run the image as a container, public on local host 5001 the apps port 5000 and download the image from this repo with this version tag
* docker run -d -p 5000:5000 in28min/hello-world-nodejs:0.0.1.RELEASE
* ^^^To run the image in detached mode so the logs are in the background, and you have a free terminal
* docker logs 04e5(2ff9270f5810eefe1f77222852dc1461c22440d4ecd6228b5c38f09d838e)

docker logs -f [docker ID] to follow the logs live, ctrl +c to disconnect from the logs

* docker images to see all locally downloaded images
* docker container ls to see all running containers, ls -a to show all containers
* docker container stop f708b7ee1a8b to stop running container

Diagram

Description automatically generated

* docker pull mysql [latest] to just download image, not launch container
* docker search mysql, look for official OK tag for a reliable image
* docker image history in28min/hello-world-java:0.0.1.RELEASE to view layers
* docker image history 100229ba687e to use image ID instead of name
* docker image inspect 100229ba687e to see full details
* docker image remove mysql to remove image
* docker image remove in28min/hello-world-java:0.0.1.RELEASE does not work as there is a container using it
* docker container rm 3e657ae9bd16 to remove the container so you can remove the image
* docker container pause 832
* docker container unpause 832
* docker container stop 832 to stop container by closing down services it’s using, gracefully
* docker container kill 832 to stop container by force immediately
* docker container inspect ff521fa58db3 to view details about the container
* docker container prune to remove all stopped containers
* docker system df to show disk usage of docker
* docker system events to see actions happening, like live logs for the system
* docker system info
* docker system prune -a to remove all stopped containers and images not in use, networks not in use and build cache
* docker top 9009722eac4d
* docker stats 9009722eac4d shows usage stats like task managers for a container
* docker container run -p 5000:5000 -d -m 512m --cpu-quota=50000 in28min/hello-world-java:0.0.1.RELEASE to run the container with a limited amount of memory (512mb) and half cpu (5000 out of the full 10000)
* cd ~/Downloads/Udemy/DevOps/devops-master-class-master/devops-master-class-master/projects/hello-world/hello-world-python go to location where you have the docker file for the image you want to make
* docker build -t smhzaihr/hello-world-python:0.0.2.RELEASE . to build an image called that with the tag (-t) 0.0.1.RELEASE using everything form this folder (.)
* docker run -p 5000:5000 -d smhzaihr/hello-world-python:0.0.2.RELEASE
* docker history e66dc383f7a0 to see how the base image was built and how our image was built on top of it
* docker push smhzaihr/hello-world-python:0.0.2.RELEASE
* Changing the layers can make the process more efficient, having the requirements file and commands above the copy command in the dockerfile will enable caching as the requirements files doesn’t change often
* docker run -d -p 5001:5000 in28min/hello-world-nodejs:0.0.3.RELEASE ping google.com this replaces the last line “CMD…” in the docker file to “ping google.com” so the application doesn’t run but the ping command does.
* Using entry point instead of CMD prevents this unless you provide an argument like so docker run -d -p –entry point 5001:5000 in28min/hello-world-nodejs:0.0.3.RELEASE

* docker run -d -p 8000:8000 --name=currency-exchange in28min/currency-exchange:0.0.1-RELEASE. Download the image and name it currency-exchange and run it as a container on port 8000. Running on http://localhost:8000/currency-exchange/from/USD/to/INR
* docker run -d -p 8100:8100 --name=currency-conversion in28min/currency-conversion:0.0.1-RELEASE This wont work in the browser as the microservices aren’t connected in docker
* docker network ls to view all networks
* Text

  Description automatically generated
* docker network inspect bridge to see which containers are a part of the bridge network, bridged containers cannot communicate by default
* docker run -d -p 8100:8100 --env CURRENCY\_EXCHANGE\_SERVICE\_HOST=http://currency-exchange --name=currency-conversion --link currency-exchange in28min/currency-conversion:0.0.1-RELEASE stop the currency conversion container and rerun it with the environment variable tag and the link tag. Now the link works: http://localhost:8100/currency-conversion/from/EUR/to/INR/quantity/10
* host network only works with Linux
* none network is no network for the containers
* docker network create currency-network to make a new network
* remove the containers
* docker container remove currency-exchange
* docker container remove currency-conversion
* relaunch them as a part of the new currency-network with tags
* docker run -d -p 8000:8000 --name=currency-exchange --network=currency-network in28min/currency-exchange:0.0.1-RELEASE
* docker run -d -p 8100:8100 --env CURRENCY\_EXCHANGE\_SERVICE\_HOST=http://currency-exchange --name=currency-conversion --network=currency-network in28min/currency-conversion:0.0.1-RELEASE
* cd ../../microservices/ go to the place where the docker compose YAML file is
* docker-compose up to run the docker compose file
* docker-compose up -d or to launch in a detached mode
* docker network inspect microservices\_currency-compose-network this network is created by the docker compose file
* docker-compose down to stop the services
* docker-compose config this is similar to the YAML file, to validate it
* can use the same commands with docker compose as docker container docker-compose images, docker-compose ps, docker-compose top..stop, pause, etc