YIYANG WANG

CSC309 Week 10

DOCKER WORKSHOP

HTTPS://WWW.CS.TORONTO.EDU/~KIANOOSH/COURSES/CSC309H5/

Please join the

Zoom for polls

Meeting Code:

2210147631

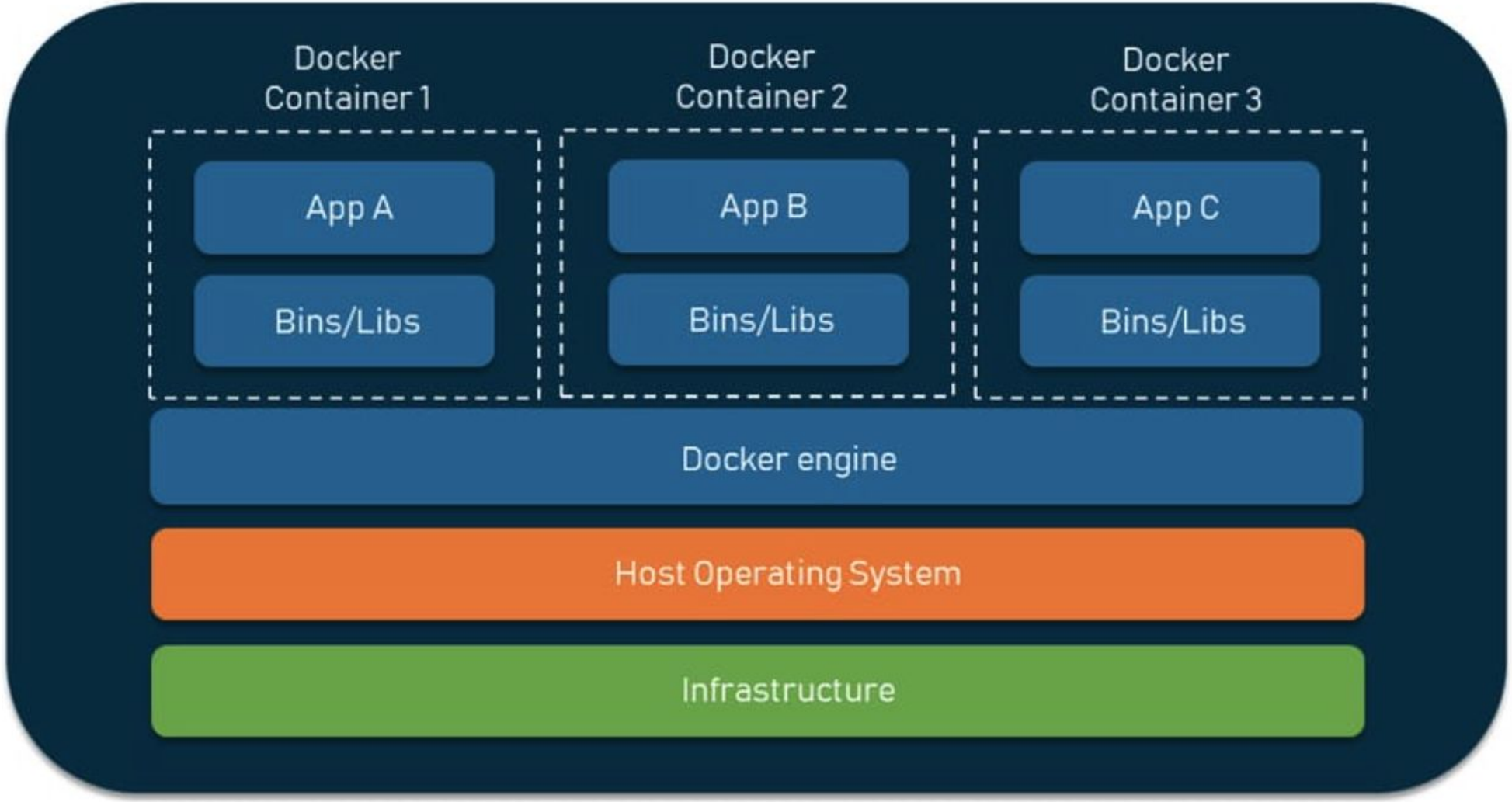
Password:

123456

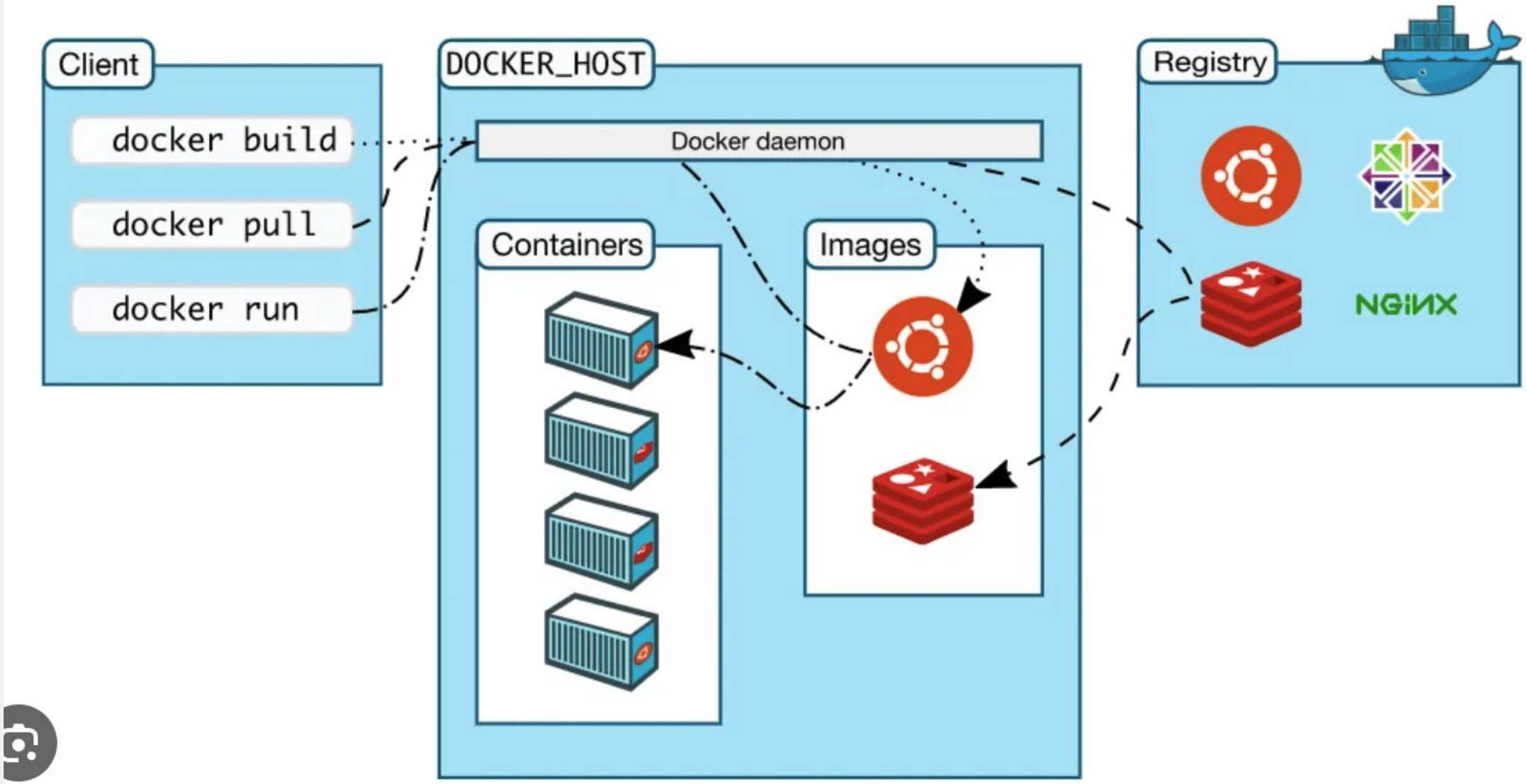
Docker

- Simply, it is a light version of virtual machine (VM). Each VM is called a container in docker.
- Similar to VM, docker’s environment is isolated from your real system.
- However, it can easily forward the real system’s file into the docker container comparing to other VM software
- Docker also can keep the environment consistent in any OS.

DOCKER CONTAINERS



altexsoft


Client
docker build
docker pull
docker run
DOCKER_HOST
Docker daemon
Containers
Images
Registry
NGINX

Docker

- Docker tools: Docker-compose
  - A tool for defining and running multi-container Docker applications.
  - It uses YAML files to configure the application's services and performs the creation and start-up process of all the containers with a single command
  - It uses “docker-compose” command client to send instruction to docker-compose

    docker-compose build build up the image for docker

    docker-compose up start up all docker containers in the YAML file with command line (use ctrl+c to exit)

    docker-compose start start up all docker containers in the YAML file in background (use docker-compose stop to exit)

Docker(-compose) - Dockerfile

- Dockerfile is an auto-script which takes care of each container’s building and running process.
- All command are running in container’s bash command line
- It formats as:
  - INSTRUCTION arguments
    - All Dockerfile should start with “FROM image:release-ver” to load the container OS image
- There are multiple environment variables you can define in the Dockerfile.

Docker(-compose) - Dockerfile

Environment Command	Explanation
ADD [--chown=<user>:<group>] <src>... <dest>	copies new files, directories or remote file URLs from <src> and adds them to the filesystem of the image at the path <dest>. (--chown option can set the ownership of the file)
COPY [--chown=<user>:<group>] <src>... <dest>	opies new files or directories from <src> and adds them to the filesystem of the container at the path <dest> (--chown option can set the ownership of the file)
ENV <key>=<value>	sets the environment variable <key> to the value <value>. This can be used in any program / executions by reading the system environment
EXPOSE <port> [<port>/<protocol>...]	informs Docker that the container listens on the specified network ports at runtime. port protocol can be specified by "/<protocol>"; the default is TCP if the protocol is not specified.
CMD ["executable","parametre"] ENTRYPOINT ["executable", "param1", "param2"]	CMD provides defaults for an executing container. There can only be one CMD instruction in a Dockerfile. An ENTRYPOINT allows you to configure a container that will run as an executable.

Docker(-compose) - Dockerfile

Environment Command	Explanation
FROM [--platform=<platform>] <image>[@<digest>] [AS <name>]	Initializes a new build stage and sets the Base Image for subsequent instructions. The image can be any valid image – it is especially easy to start by pulling an image from the Public Repositories.
USER <user>[:<group>] USER <UID>[:<GID>]	Sets the user name (or UID) and optionally the user group (or GID) to use when running the image and for any RUN, CMD and ENTRYPOINT instructions that follow it in the Dockerfile (in case you don't want to run the program by root)
VOLUME ["/data"]	Creates a mount point with the specified name and marks it as holding externally mounted volumes from native host or other containers
WORKDIR	The WORKDIR instruction sets the working directory for any RUN, CMD, ENTRYPOINT, COPY and ADD instructions that follow it in the Dockerfile
RUN <command> RUN ["executable","parametre"]	Execute any commands in a new layer on top of the current image and commit the results. The resulting committed image will be used for the next step in the Dockerfile

Computer Science
UNIVERSITY OF TORONTO

docker-compose.yml Format

- The example here is using the newest version 3.8

docker-compose.yml

version: “3.8” □ define docker-compose file version at the beginning, version needs to get quoted by double quotation.

services: □ list the service below with indentation

service_name: □ define a service

build: ./dir □ setup the service build directory, usually within the current directory's subfolder. It will automatically search the dockerfile

[context: ./dir

dockerfile: Dockerfile-alternate] □ if you have alternative dockerfile name, use this format.

args: □ arguments below with indentation

environment: □ environment variables below with indentation

volumes: □ mount real system directory to container, having the - <src>: <det> below with indentation

ports: □ port forwarding to the real system, having - “forward_port:container_port” below with indentation

Quick demo

docker-compose up -d

docker-compose ps

docker-compose stop

docker-compose start

docker exec -it db mysql -u -p

docker-compose start db

Practice - Dockerfile and docker-compose.yml

https://drive.google.com/file/d/188f_FvpBxzvHHHjg3EYGOFeOjmQm09dx/view?usp=sharing

https://github.com/yiyangwww/docker-workshop.git

- **Where are the VMs on which we can run docker**
  - mcsdocker.utm - 142.1.114.66

    mcsdocker2.utm - 142.1.114.76

    mcsdocker3.utm - 142.1.114.77
  - Log in using your utorid and password, just like for lab machines.
  - If docker (e.g., docker pull etc.) is not working for you, follow these instructions from Andrew Wang:

```
dockerd-rootless-setuptool.sh install

Then in their .bashrc, add
export PATH=/usr/bin:$PATH
export DOCKER_HOST=unix:///run/user/1000/docker.sock

Where 1000 (in the above bottom export line) is the result of
id -u
```