**Docker**

* docker ps : to get all running container, -a for all containers
* docker rmi <image id>: To remove image
* docker rm <container id>: To remove container
  + docker rm $(docker ps -a -q)
* docker create [OPTIONS] IMAGE [COMMAND] [ARG...] : To create new container, [OPTIONS]-> -it(flags), IMAGE-> image id to start container with, COMMAND-> initial command to start container with, [ARG]-> arguments needed for initial command to run
* docker rename ORIGINAL\_NAME NEW\_NAME: to rename existing container
* docker run [OPTIONS] IMAGE [COMMAND] [ARG...] : To create and start container
  + –name : to assign name to container [user/reponame:version]
  + -i: Keep STDIN open even if not attached
  + -t: Allocate a pseudo-TTY
  + –privileged : flag gives all capabilities to the container, can be used to run docker inside docker
  + -w: lets the command being executed inside directory given, here /path/to/dir/
  + -v: mounts the current working directory into the container
  + -p: binds port of the container to TCP port of the host machine
  + -d: Run container in background and print container ID
  + –rm: Remove container once image is build
  + -c : To set how much cpu should container use[1024 means 100%,512 means 50]
  + -m: To set how much memory should container use [-m 300M for 300mb]
* docker logs <Container id> : To view logs of container
* docker port <Container id >: To view port mappings
* docker top <Container id>: To view all running process inside container
* docker stats <Container id>: To view hardware resources used by container
* docker diff <Container id>: To see changes inside container File system
* docker export <Container id>: Export a container’s filesystem as a tar archive
* docker start <Container id >: To start a container
* docker stop <Container id>: To stop a running container
* docker cp [OPTIONS] CONTAINER:SRC\_PATH DEST\_PATH: To copy files from host to docker container
* docker exec [OPTIONS] CONTAINER COMMAND [ARG...]: Run a command in a running container
* docker commit <Container id> : creates image from a container, pausing it temporarily if it is running.
* docker save <Container id > > name\_of\_image.tar : To create a tarball of container
* docker load < name\_of\_image.tar : To load image from tar ball

**DockerFile**

* docker build . : To build from dockerfile
* FROM : Sets the Base Image for subsequent instructions.
* RUN: execute any commands in a new layer on top of the current image and commit the results.
* docker history <Image Id> : To view all the layers during building from dockerfile
* CMD: provide defaults for an executing container.
* EXPOSE informs Docker that the container listens on the specified network ports at runtime. NOTE: does not actually make ports accessible.
* ENV sets environment variable.
* COPY copies new files or directories to container. By default this copies as root regardless of the USER/WORKDIR settings
* WORKDIR sets the working directory.
* Use -t tag while building dockerfile[Ex docker build -t="crosbymichael/sentry" .]

**Docker-compose**

version: '2'

services:

web:

build: .

# build from Dockerfile

context: ./Path

dockerfile: Dockerfile

ports:

- "5000:5000"

volumes:

- .:/code

redis:

image: redis

* docker system prune: remove unused volumes, networks, exited containers and unused images.
* docker volume rm $(docker volume ls -q -f "dangling=true"):Remove Dangling Volumes
* docker rm $(docker ps -q -f "status=exited"): Remove exited container
* docker rmi $(docker images -q -f "dangling=true"): Remove dangling images

To build docker-compose use docker-compose up

* docker system prune: remove unused volumes, networks, exited containers and unused images.
* docker volume rm $(docker volume ls -q -f "dangling=true"):Remove Dangling Volumes
* docker rm $(docker ps -q -f "status=exited"): Remove exited container
* docker rmi $(docker images -q -f "dangling=true"): Remove dangling images