**Docker**

* Go through notes from notebook.
* Then revise below important points –
* port mapping - docker run -p <host\_port>:<container\_port> -d <image\_name>
* pass env variables - docker run -e USER\_NAME=admin -e PASSWORD=pwd --name test -d nginx
* list running containers - docker ps
* list all containers - docker ps -a
* docker stop <container\_id>
* docker start <container\_id>
* docker rm <container\_id>
* docker inspect <container\_id>
* docker exec <container\_id> -it sh
* docker logs <container\_id>
* to list all the Docker networks available on your system = docker network ls
* use build-arg to pass variable value to Dockerfile -

docker build -t <image\_name:tag> <Dockerfile\_location> --build-arg JAVA\_VERSION=18-jdk

* Dockerfile instruction differences -
* add vs copy -

copy will work only for source and dest files present on host.

add has all functionalities as copy and additional functionality to download

* arg vs env -

arg will be available only while building container and not inside the container

env will be available inside the container as well

* cmd vs entrypoint –

entrypoint is the first instruction for the container and if we change it while running the container then it will change whole instruction.

cmd is the instruction for the running container and if we change it while running the container then it will change will get appended to the existing instruction

* docker volume -

To persist the container's data, we can use volumes with -v in docker run command.

The volume can be either any directory on docker host or our created docker volume.

We can create docker volume using - "docker volume create <volume\_name>" command and list using - "docker volume ls"

* Docker compose -

Docker Compose is a tool for defining and running multi-container applications. Docker Compose deploys containers on a single host.