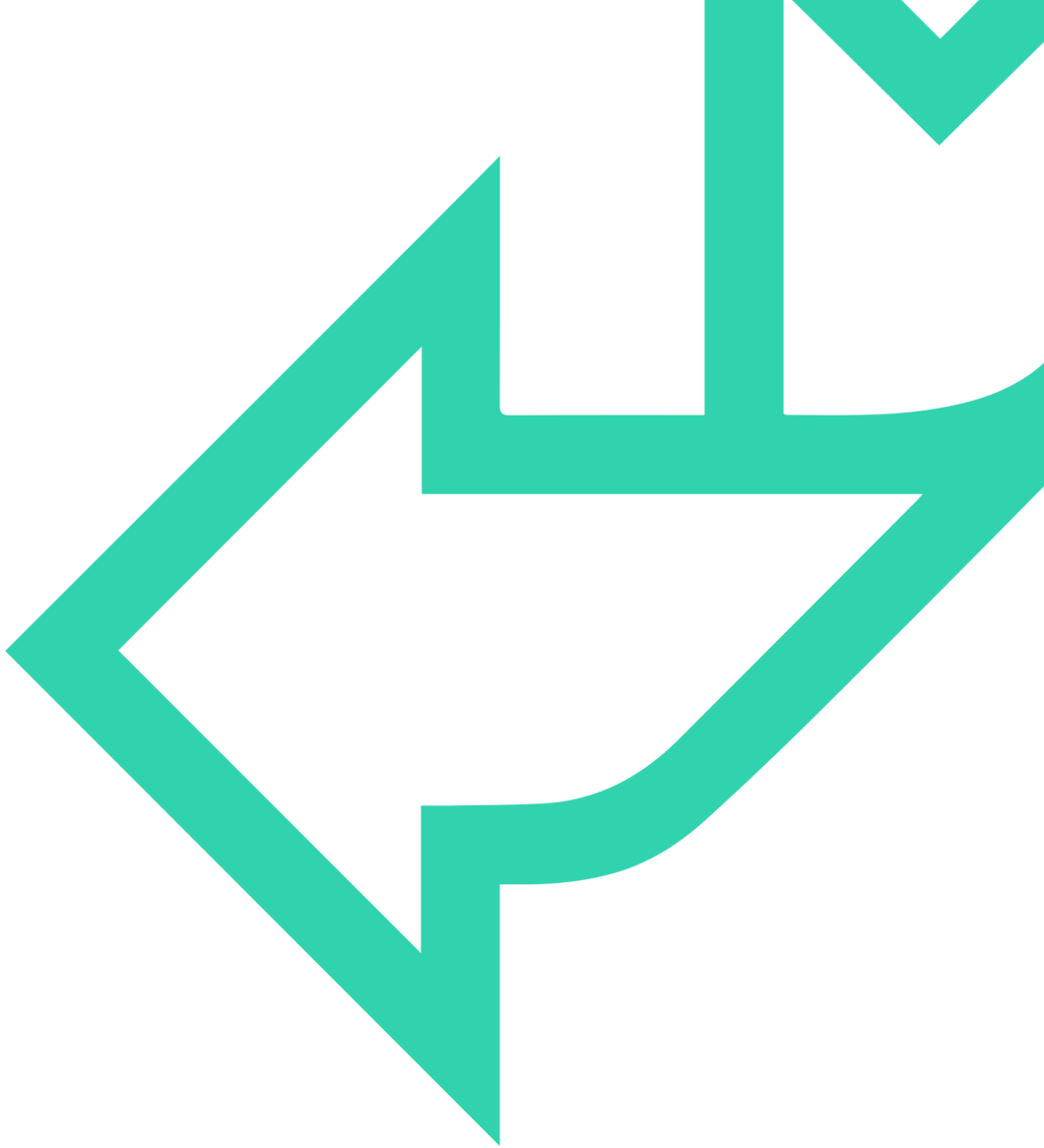
POWERING POTENTIAL

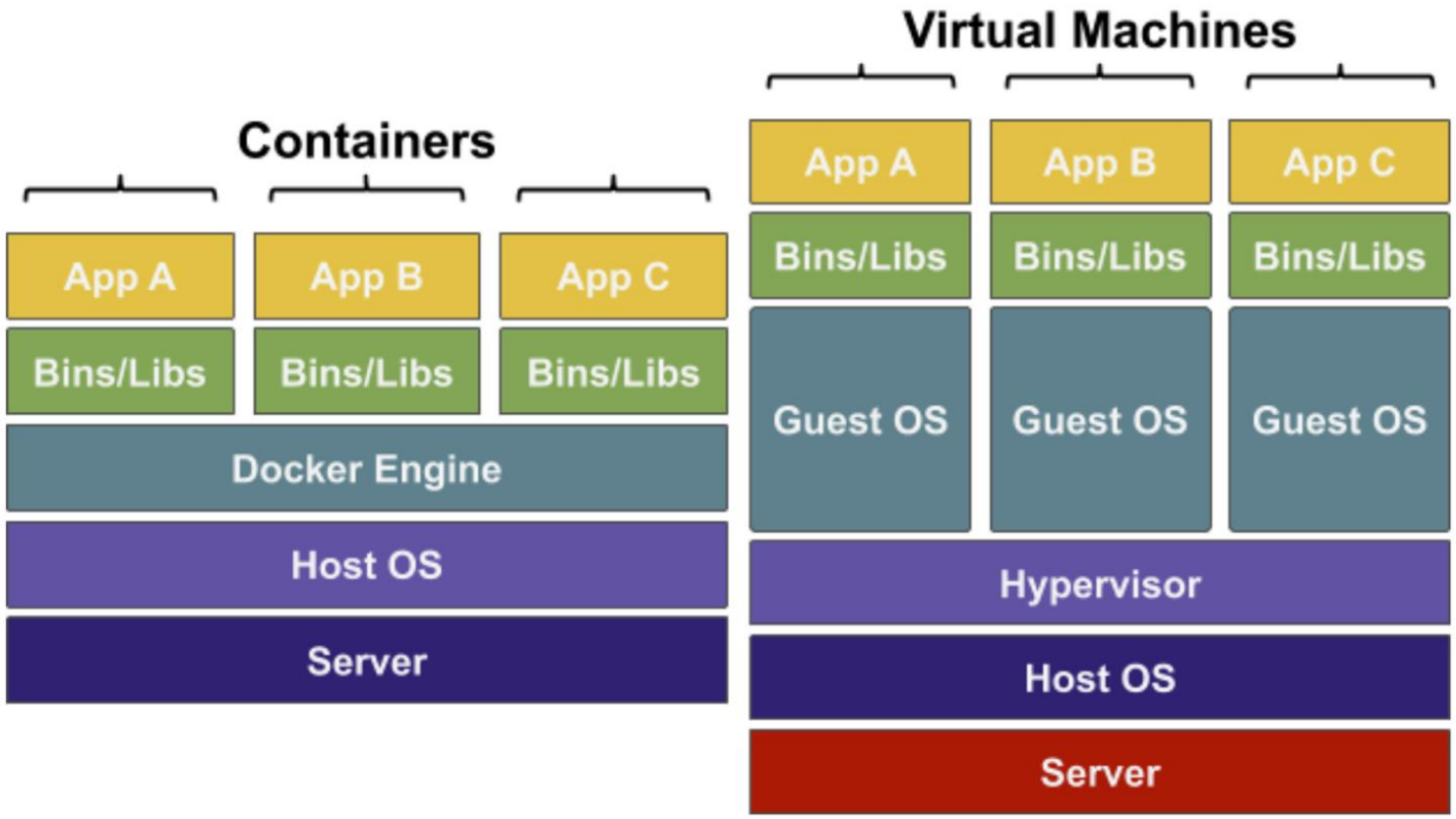


Docker Intro

Containers

- Used to transport code in a specific environment.
- Lightweight alternative to VMs
- Less computing power on unnecessary processes.

Containers vs Virtual Machines



Server – Physical hardware for the processing power

Host OS – Operating system on the server

Hypervisor – Installed in the Host OS, software to create VMs

Guest OS – Operating system used by the customer

Bins/Libs – Program files for the app to run

App – The app to run

Docker Engine – Containerisation Software on Guest OS

CONTAINERS VS VMS

Why developers use containers over VMs

Containers:

- Use the same kernel (the core of OS) as the host
- Resources are shared between containers if possible
- Extremely fast start-up time

Virtual machines:

- VMs emulate an entire computer to support an OS
- There is complete separation between machines
- Far more resources are used to isolate the hosted app

What is Docker?

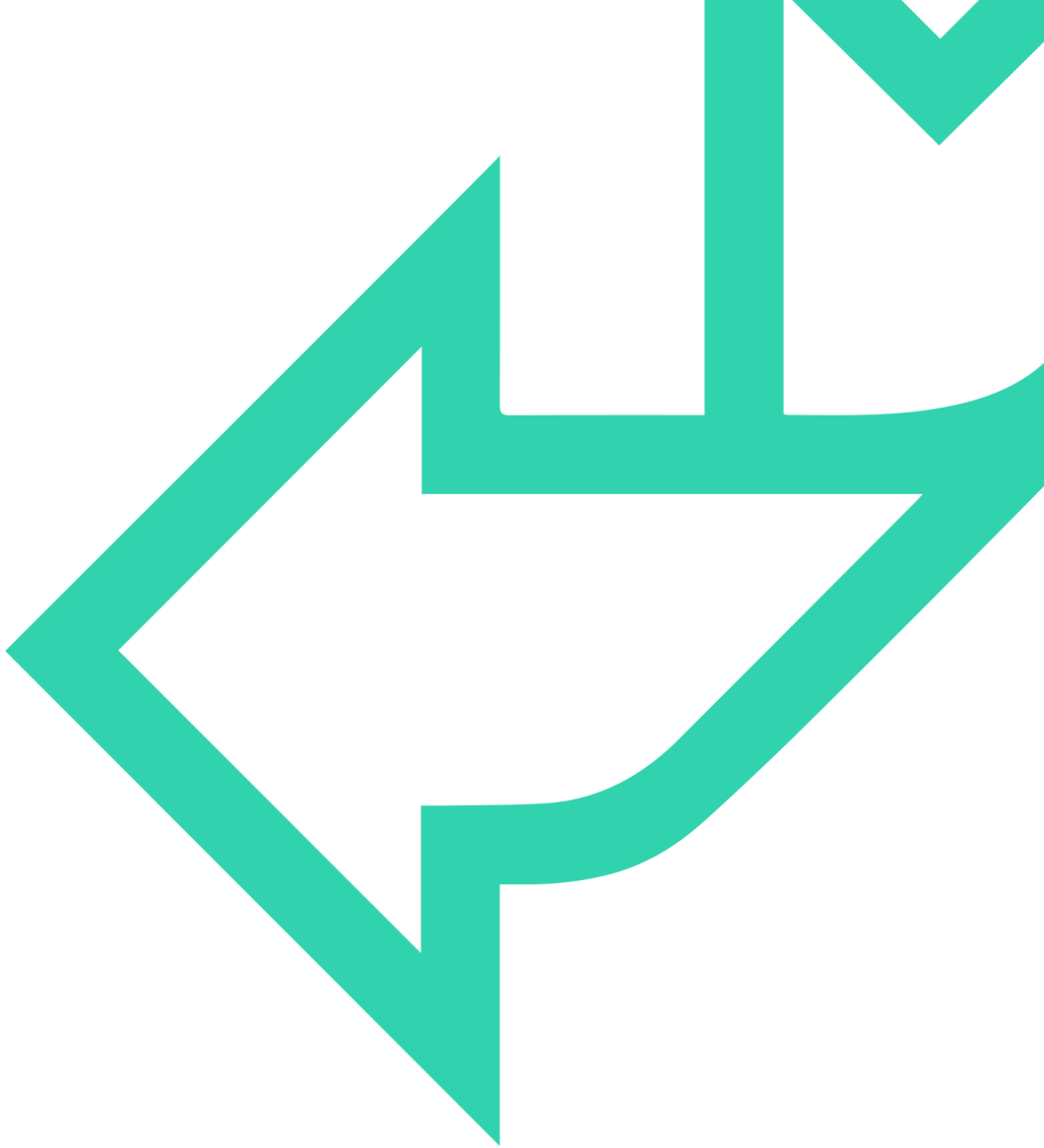
Open-source containerisation tool,
Well known in the DevOps and Sys Admin space.

Let you configure and replicate environments so you can focus on the code.

Require less computing power as you do not need a guest OS on each container.

Docker can be installed easily with a remote script hosted on **https://get.docker.com** or can be installed directly on Windows or Mac.

POWERING POTENTIAL



Docker Images

DOCKER IMAGE

- Is a read-only file
- A snapshot of the state of a system at a particular point in time
- It is like taking an image of your PC. You can then move it to another PC

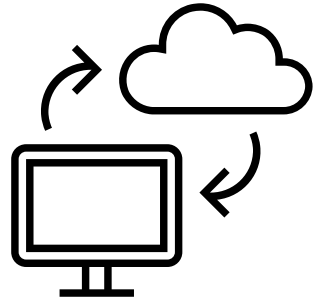
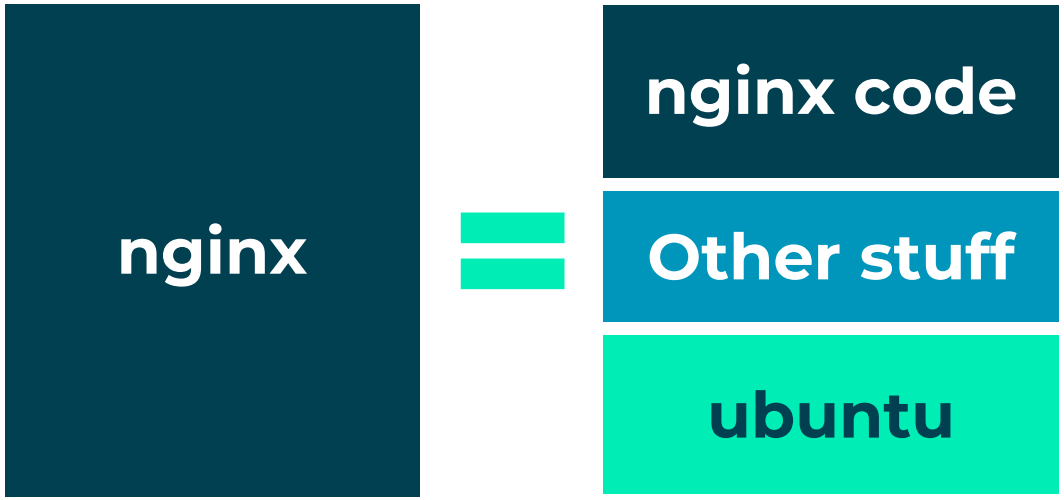


Image Layers



- Images are composed of multiple layers
- Layers are read only. To change a layer, create a new one and add it to image
- Start with a simple image and add new layers to create new images.

```
FROM alpine:3.4

RUN apk update
RUN apk add vim
RUN apk add curl
```

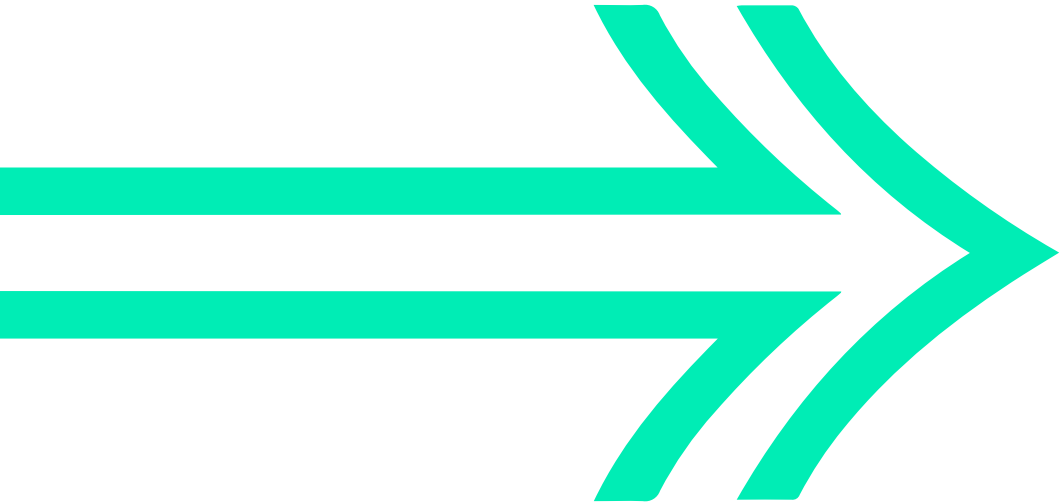
- Each *Dockerfile* instruction creates a new layer.
- Layers are cached, so changing a line in Dockerfile rebuilds only the layer affected by that change

Registry - DockerHub

- Images are typically stored in remote registries
- Can be used by other people for their CI/CD

- Docker Hub is the default image registry https://hub.docker.com/

- The Docker CLI works with Docker Hub Login with **docker login**



POWERING POTENTIAL

Docker Containers

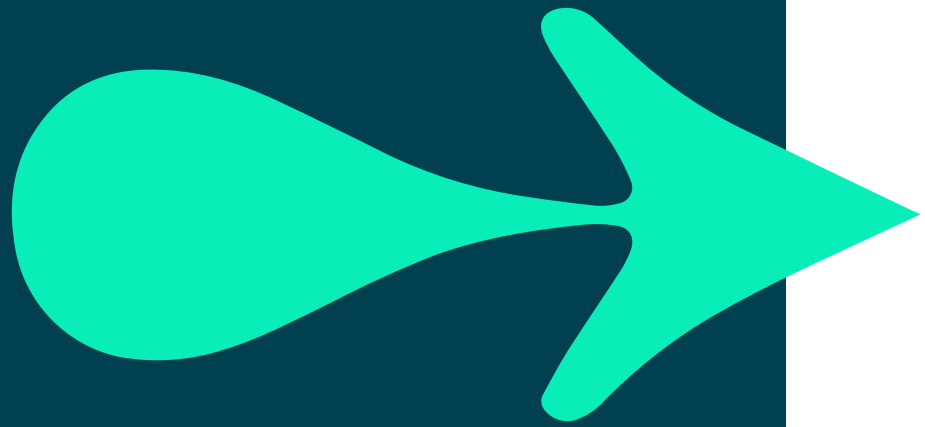
CONTAINERS

Containers are what our images run in

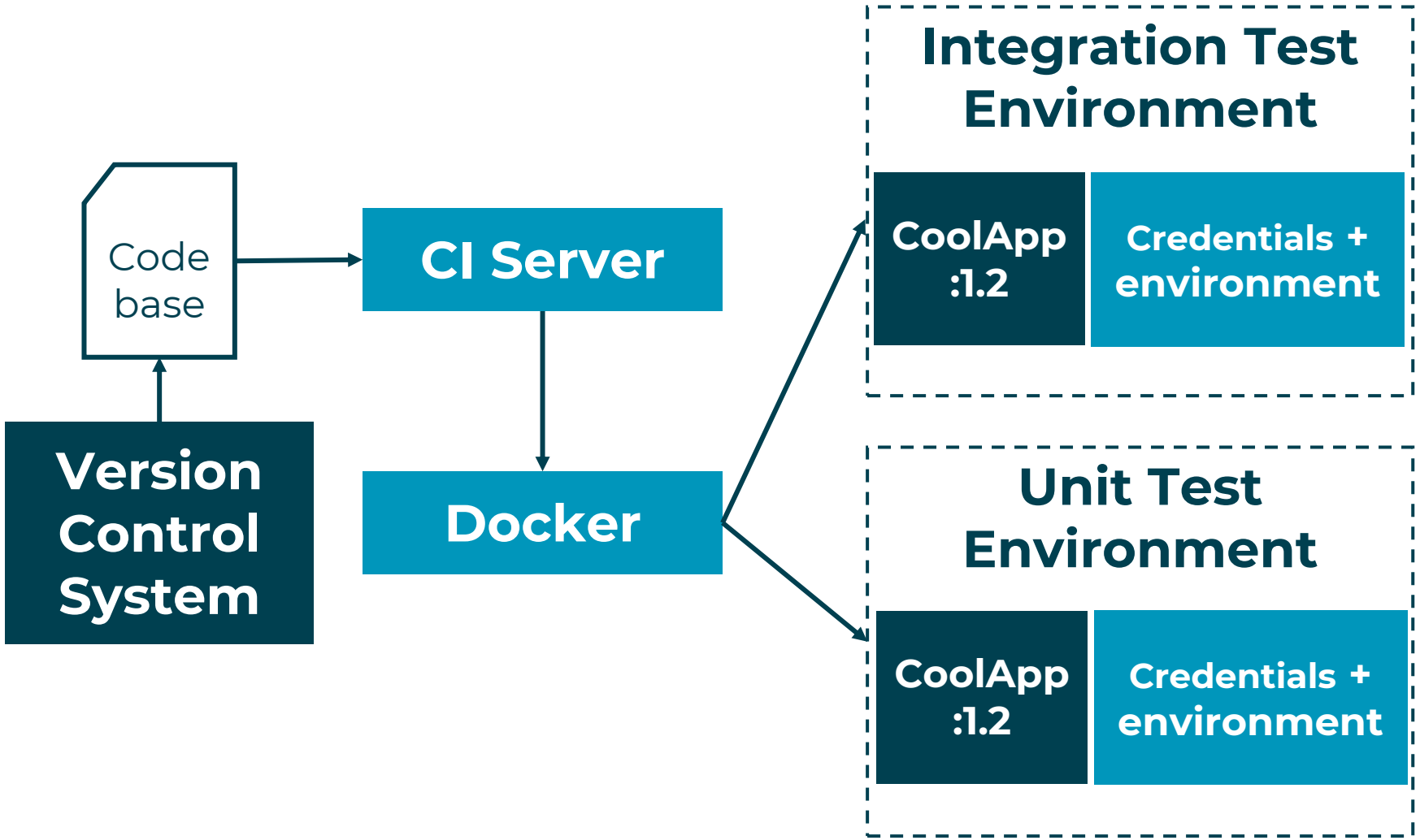
They run in the **same way across any device with Docker**.

Containers have the following info:

- Container ID
- Image
- Command
- Created
- Status
- Ports
- Names



Containers in CI



Containers allow you to simultaneously run your codebase in a repeatable environment.

Containers run the same in any machine, Can replicate part of the production environment.

Saves time and human error

A FEW CONTAINER COMMANDS

When running our application inside a container, its important to be able to interact with them individually and collectively.

There are limited commands to running and managing a number of containers. Container orchestration tools such as Kubernetes make this job easier.

Command	Function
ps ps -a	Lists all running containers -a shows all containers, running or not
logs	Returns logs of the container
exec	Connect to an individual terminal via a shell
start	Starts a stopped containers (**Run** creates and starts container)
stop	Stops a running container
rm (-f)	Deletes a container (-f forces it to be removed)
rename	Renames a container

LAB

Please do the lab

1-Docker basics.docx