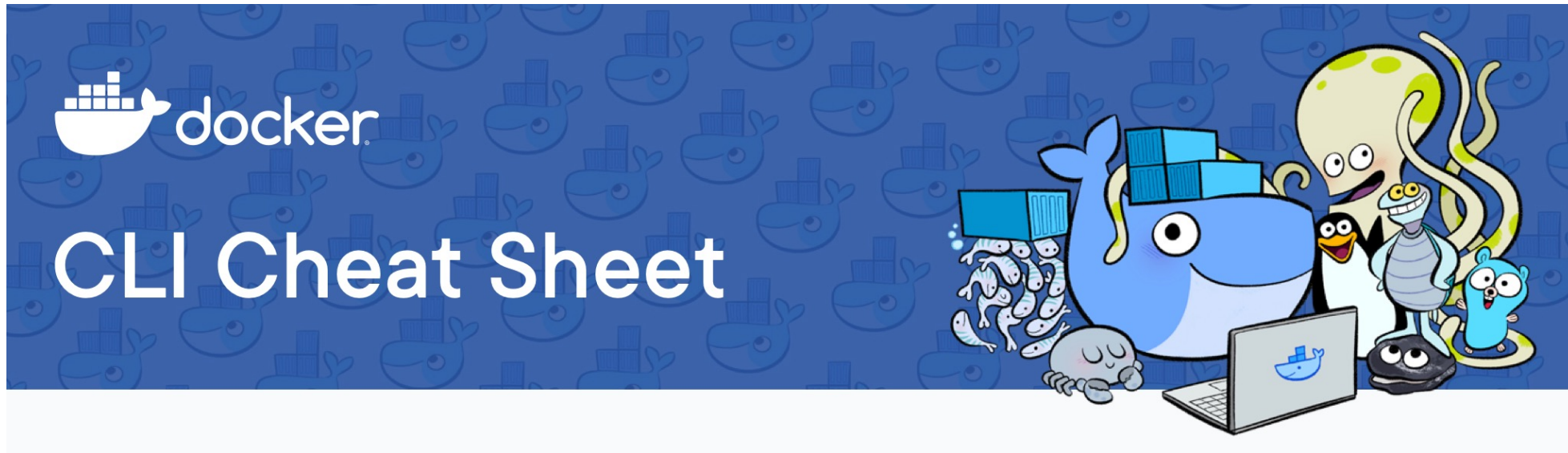


LAB-02

Deploy server in container
manually

- https://docs.docker.com/get-started/docker_cheatsheet.pdf



: create and run a new container from an image

- The command runs a command in a new container, pulling the image if needed and starting the container

```
# Run Nginx in background with the name 'nginx'  
docker run --name nginx -d nginx:alpine
```

```
# Run Busybox and remove the container afterward  
docker run --name busybox --rm busybox echo "hello world"
```

```
# Publish (map) port 80 on container to port 3000 on localhost  
docker run -d -p 3000:80 nginx:alpine
```

```
# Publish all exposed ports to random ports  
docker run -d -P nginx:alpine
```

```
# Restart the container when daemon startup
docker run -d --restart=unless-stopped nginx:alpine
sudo systemctl restart docker
docker ps
```

```
# Set environment variables in the container. Can use variables exported in host
export firstname=Jack
docker run --rm -e firstname --env lastname=Russell busybox env | grep name
lastname=Russell
firstname=Jack
```

: list containers

- alias of `docker container ls`
- `-a, --all` show all containers (default shows just running)
- `-f, --filter` filter output based on conditions provided e.g. {name, status, ancestor, publish, expose}
- `-q, --quiet` display container IDs only

```
# List all containers with the name 'busybox'  
docker ps -a --filter name=busybox
```

```
# List all containers with the image 'ubuntu' or has it as base  
docker ps --filter ancestor=ubuntu
```

```
# List all containers that expose port 80  
docker ps -f expose=80
```

```
# List all containers that publish port 80  
docker ps -f publish=80
```

Exercises

1. List all exited containers
2. Delete all exited containers in one command [hint: use '\$()']

: execute a command in a running container

```
# Run Nginx in background with the name 'nginx'  
docker run --name nginx -d nginx:alpine
```

```
# Execute shell in nginx container  
docker exec -it nginx sh
```

```
ps -ef  
exit
```

Process with PID = 1 is the container's primary process

```
# Execute a chained or a quoted command with sh  
docker exec -it nginx sh -c "echo a && echo b"
```

```
# Run Nginx container with shell  
docker run nginx:alpine sh
```

```
ps -ef  
exit
```

NB: no nginx process!

history command

: show the history of an image

○ alias of `docker image history`

```
# View image history
```

```
docker history nginx:alpine
```

IMAGE COMMENT	CREATED	CREATED BY	SIZE
02fffd439b71d	11 months ago	/bin/sh -c set -x && apkArch="\$(cat /etc...	30.8MB
<missing>	11 months ago	/bin/sh -c #(nop) ENV NJS_VERSION=0.7.11	0B
<missing>	11 months ago	/bin/sh -c #(nop) CMD ["nginx" "-g" "daemon...	0B
<missing>	11 months ago	/bin/sh -c #(nop) STOPSIGNAL SIGQUIT	0B
<missing>	11 months ago	/bin/sh -c #(nop) EXPOSE 80	0B
<missing>	11 months ago	/bin/sh -c #(nop) ENTRYPOINT ["/docker-entr...	0B
<missing>	11 months ago	/bin/sh -c #(nop) COPY file:e57eef017a414ca7...	16.4kB
<missing>	11 months ago	/bin/sh -c #(nop) COPY file:abbcbf84dc17ee44...	12.3kB
<missing>	11 months ago	/bin/sh -c #(nop) COPY file:5c18272734349488...	12.3kB
<missing>	11 months ago	/bin/sh -c #(nop) COPY file:7b307b62e82255f0...	8.19kB
<missing>	11 months ago	/bin/sh -c set -x && addgroup -g 101 -S ...	6.85MB
<missing>	11 months ago	/bin/sh -c #(nop) ENV PKG_RELEASE=1	0B
<missing>	11 months ago	/bin/sh -c #(nop) ENV NGINX_VERSION=1.23.4	0B
<missing>	11 months ago	/bin/sh -c #(nop) LABEL maintainer=NGINX Do...	0B
<missing>	11 months ago	/bin/sh -c #(nop) CMD ["/bin/sh"]	0B
<missing>	11 months ago	/bin/sh -c #(nop) ADD file:e51d4089e73ad6dee...	8.14MB

history command

```
# View busybox image history
```

```
docker history busybox
```

IMAGE	CREATED	CREATED BY	SIZE	COMMENT
650fd573e056	10 months ago	BusyBox 1.36.1 (glibc), Debian 12	4.17MB	

```
# Cannot run busybox without a command
```

```
docker run --name busybox -d --rm busybox
```

busybox has no default command!

```
docker ps -a -f name=busybox
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

```
# May run busybox with sh
```

```
# NB: need -it option
```

```
docker run --name busybox -d --rm -it busybox sh
```

```
docker ps -a -f name=busybox
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
4e09b507fe92	busybox	"sh"	6 seconds ago	Up 5 seconds		busybox

: copy files/folders between a container and the local filesystem

- container paths are relative to the container's root directory

```
docker run --name busybox -d --rm -it busybox sh
echo hello world > hello
```

```
# Copy file 'hello' to 'busybox' container at /tmp
docker cp hello busybox:tmp
```

Successfully copied 2.05kB to busybox:tmp

```
docker exec busybox cat /tmp/hello
hello world
```

```
# Copy file 'index.html' in container to host
docker cp nginx:usr/share/nginx/html/index.html .
```

Exercise 1: deploy lvmxxyyy in container

- Create and run a container to host 'lvmxxyyy'
 - Base-image: `nginx:alpine`
 - Name: `lvm[xxyyy]`
 - Restart policy: `unless-stopped`
 - port: `0.0.0.0:3000`
 - web-page: `lvmxxyyy`
 - content: `lvmxxyyy.sit.kmutt.ac.th`

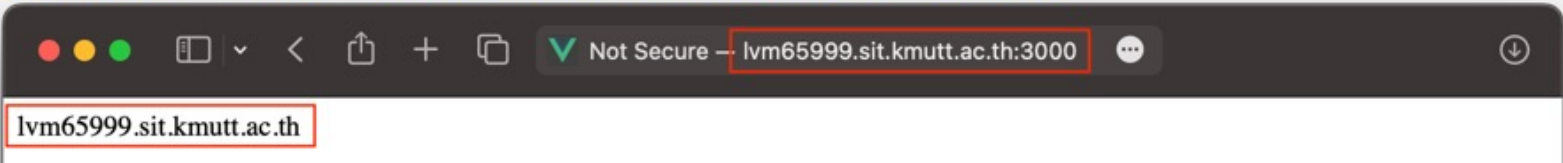
```
sysadmin@lvm65999:~$ docker ps -f name=lvm65999
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
1f8bf5269e5c	nginx:alpine	"/docker-entrypoint...."	30 minutes ago	Up 13 minutes	0.0.0.0:3000->80/tcp, :::3000->80/tcp	lvm65999

```
sysadmin@lvm65999:~$ docker inspect --format '{{.HostConfig.RestartPolicy.Name}}' lvm65999
```

unless-stopped

```
sysadmin@lvm65999:~$
```



: create a new image from a container's changes

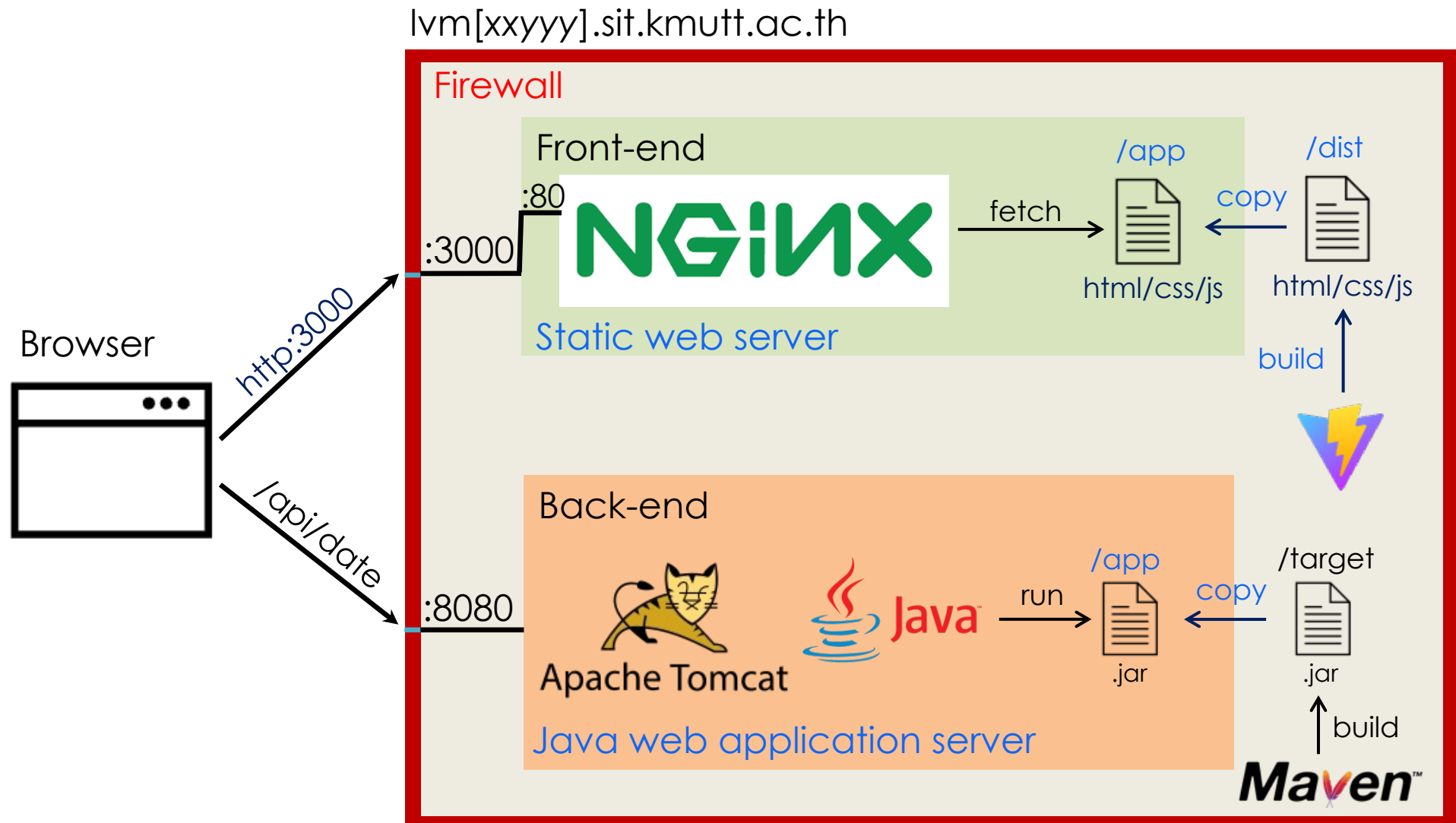
```
# commit the changes to the image 'nginx:test'  
docker commit nginx nginx:test  
  
# List all images with the image name 'nginx'  
docker images nginx  
  
# View history of nginx:test image  
docker history nginx:test
```

Exercise

1. `commit lvmxyyy` to `'lvmxyyy:bare'`

Exercise 2: fetchdate/dateapi in container

INT209 DevOps



- Create and run a container to host 'dateapi'
 - Base-image: `nginx:alpine`
 - Name: `fetchdate`
 - Restart policy: `unless-stopped`
 - port: `0.0.0.0:3000`

- Create and run a container to host 'dateapi'
 - Base-image: `openjdk:17-jdk-alpine`
 - Name: `dateapi`
 - Restart policy: `unless-stopped`
 - port: `0.0.0.0:8080`