# [Docker Notes]

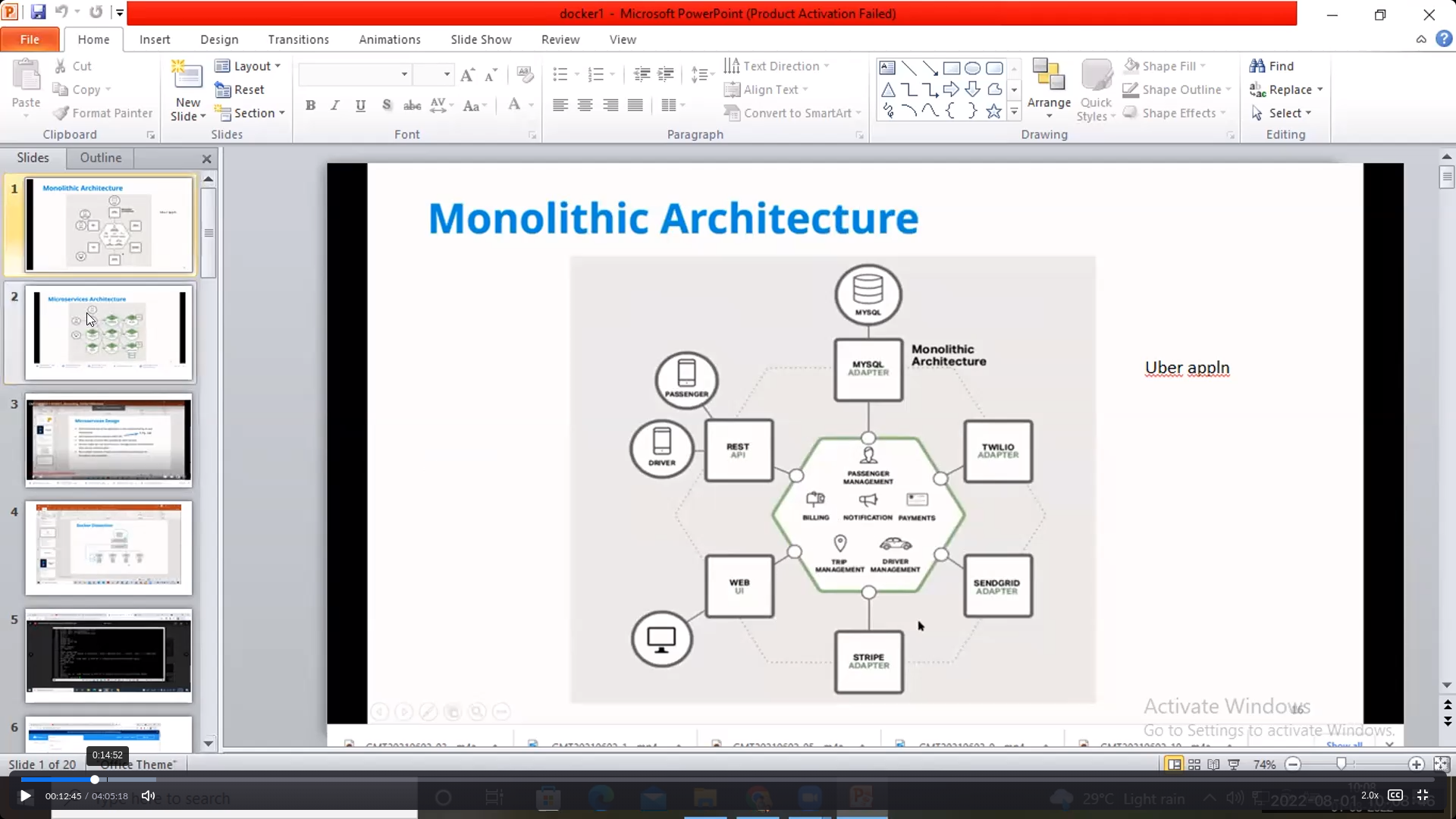
**Imp- Mrng-Introduction**

**Evng:container creation and curl**

Monolithic architecture:

One service is dependent on another (tightly coupled) changes in the code reflect all over the product

Single database

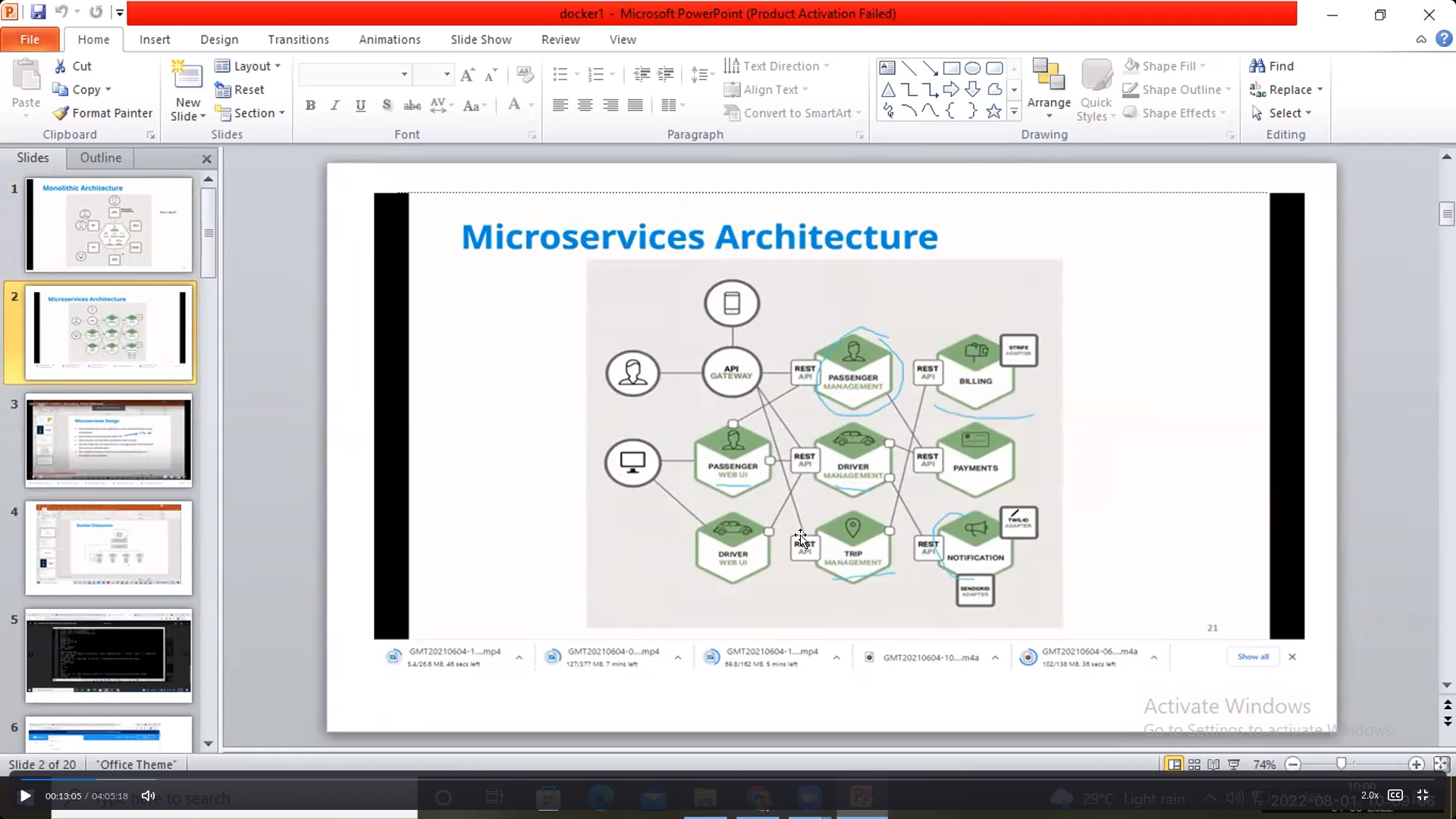


MicroService architeture:

Independent serivece(loosley coupled)

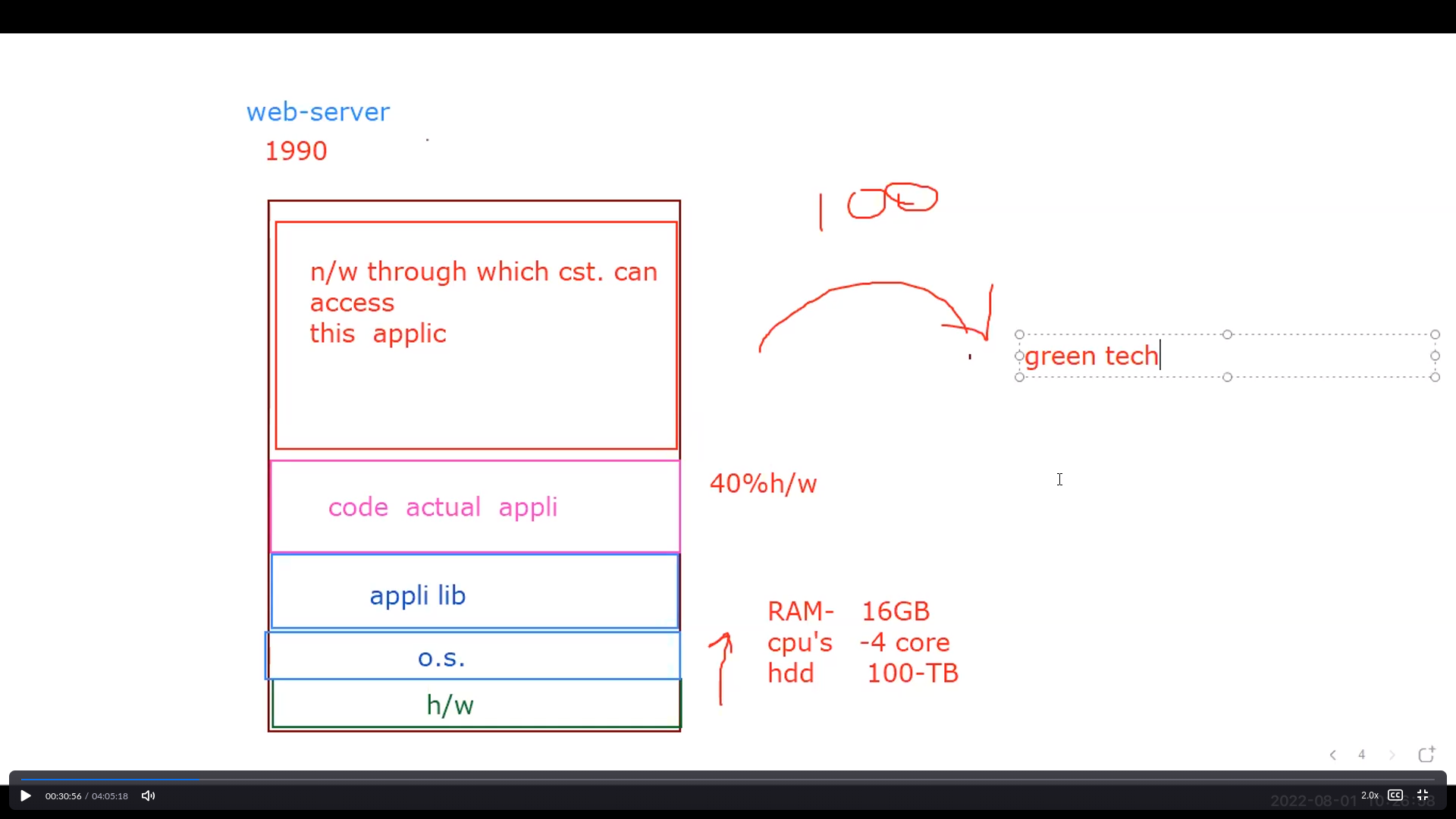
Changes in reflect to particular sevice not all over the product

Multiple databases



Old technology

One machine one webserver

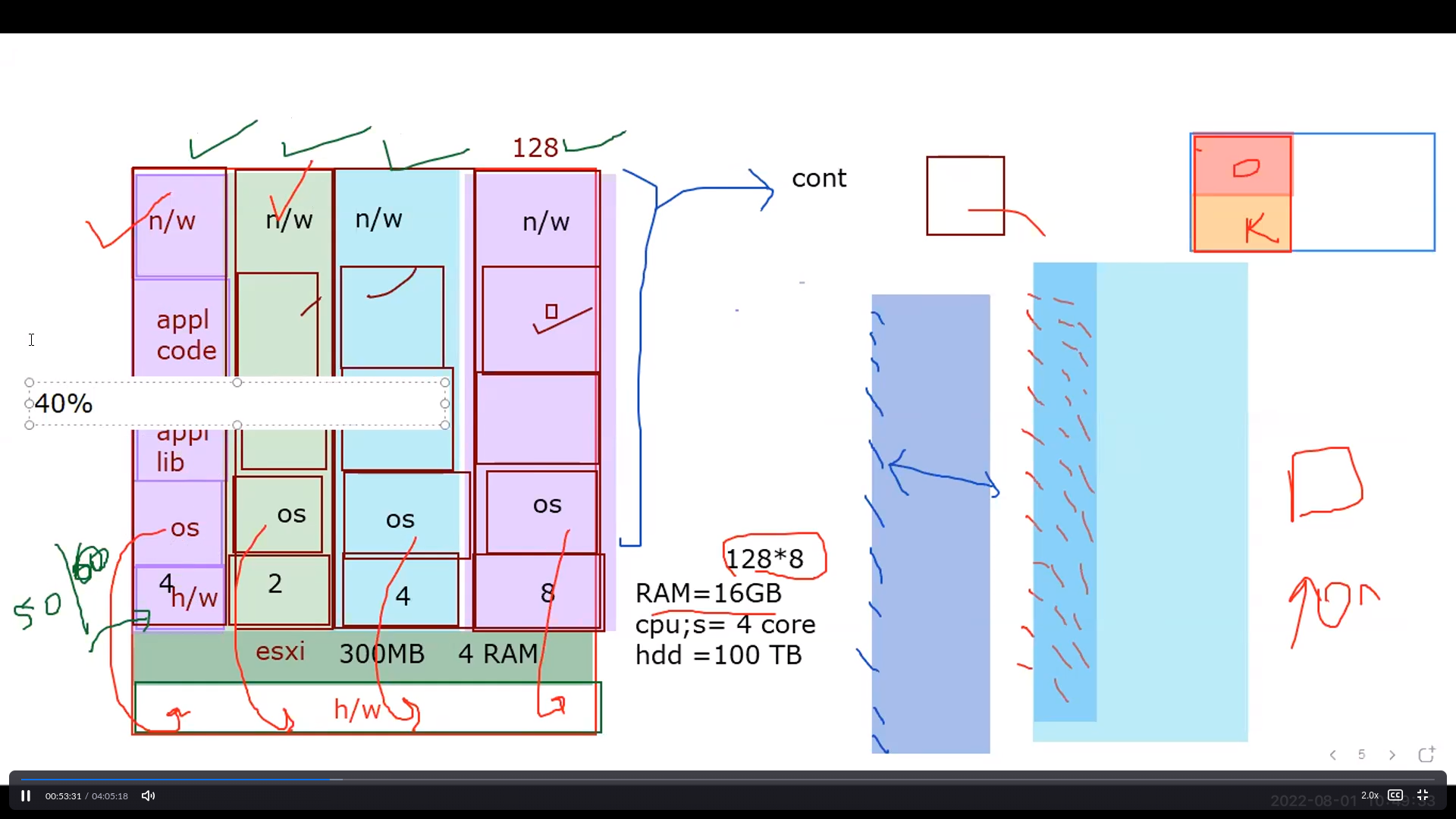


Drawbacks:

Wastage of h/w resources 40% of resources are used

Green technology:

One machine 128 servers using hypervoiser which handles the 128 servers



Drawbacks :

Need large amount of ram because whenever swithch on the machine all micro sevices need to be loaded because multiple kernel os

40% of resources might used in each service

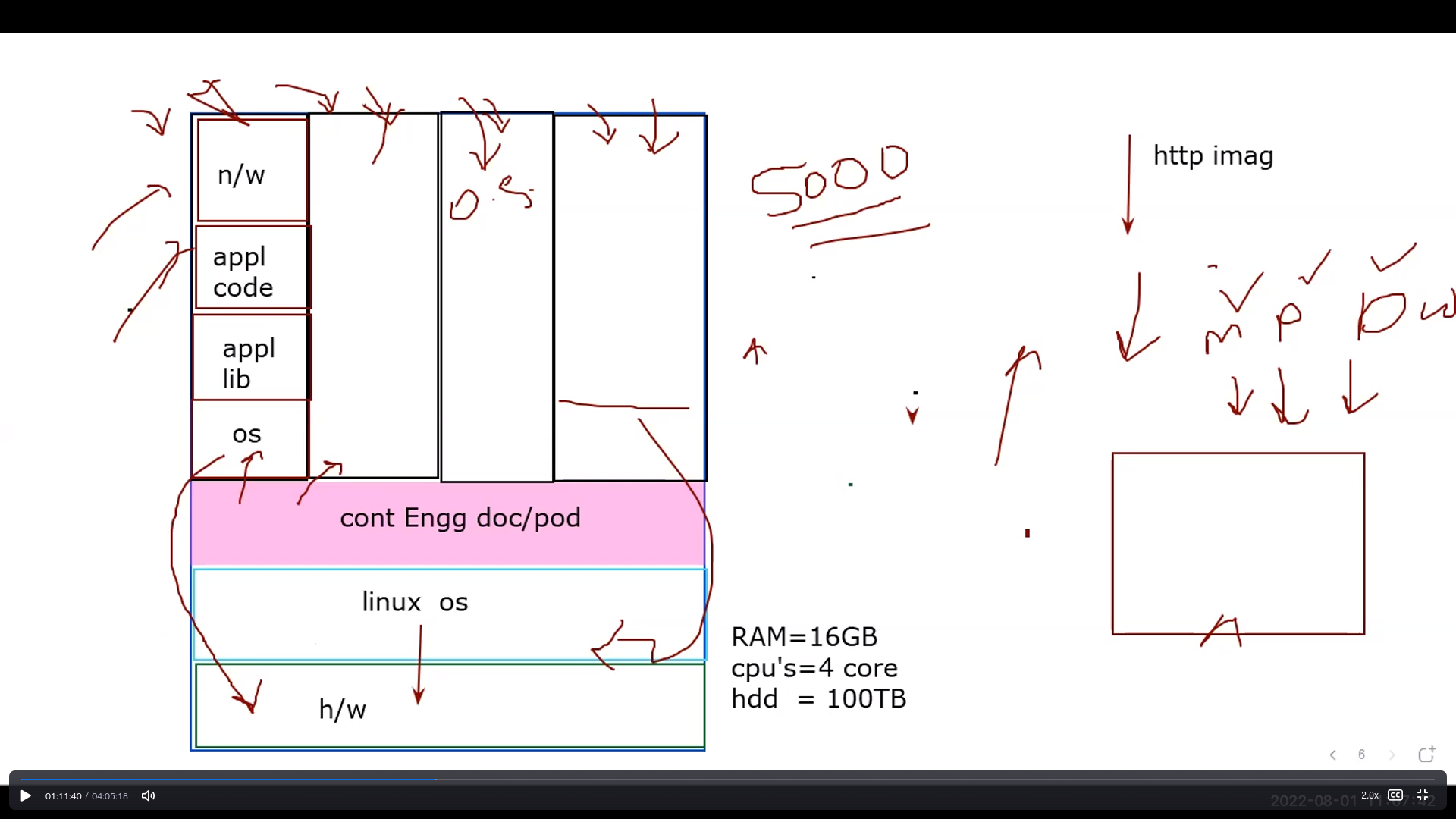
Containerization:

One machine 5000 microservies

Single kernal os so needs less ram

Resouces are allocated according to the how much service need

We pull the images So No need of installation compatibilty os container itself handles



Docker installation:

Launch aws instance and connect

Commands:

1. yum update -y
2. yum install docker
3. systemctl start docker
4. systemctl enable docker
5. systemctl status docker

Docker commands[launching containers]

docker ps –a = to check all running and not running containers

docker search httpd : to search image

docker search httpd -f stars =4 is-official=true : to search image

With filter –f

docker create --name=web1 httpd:latest : to create the container in non running mode

docker start web1 : to run the lanched container

docker rm -f $(docker ps -a -q) : to remove all container in que

1 docker inspect web1 : to check containr info

2 docker inspect web1 |grep -i ipaddr : to check ipaddress

3 docker pause web1 : to stop container temporarily

4 docker ps : to show the running containers

5 docker unpause web1 : to start container again

6 docker ps

7 docker exec -it web1 bash : to enter into container

8 docker search ubuntu : to search for images

9 docker pull ubuntu : to pull the images

10 docker images : to show the images

11 docker run -d --name ub1 ubuntu : create+start [-d detachment mode which is in lanch the container in background]

12 docker ps

13 docker rm -f ub1 : to remove the containers

14 docker run -d -it --name ub1 ubuntu : to launch the contaiers(-it is mandatory for os containers)

15 docker ps

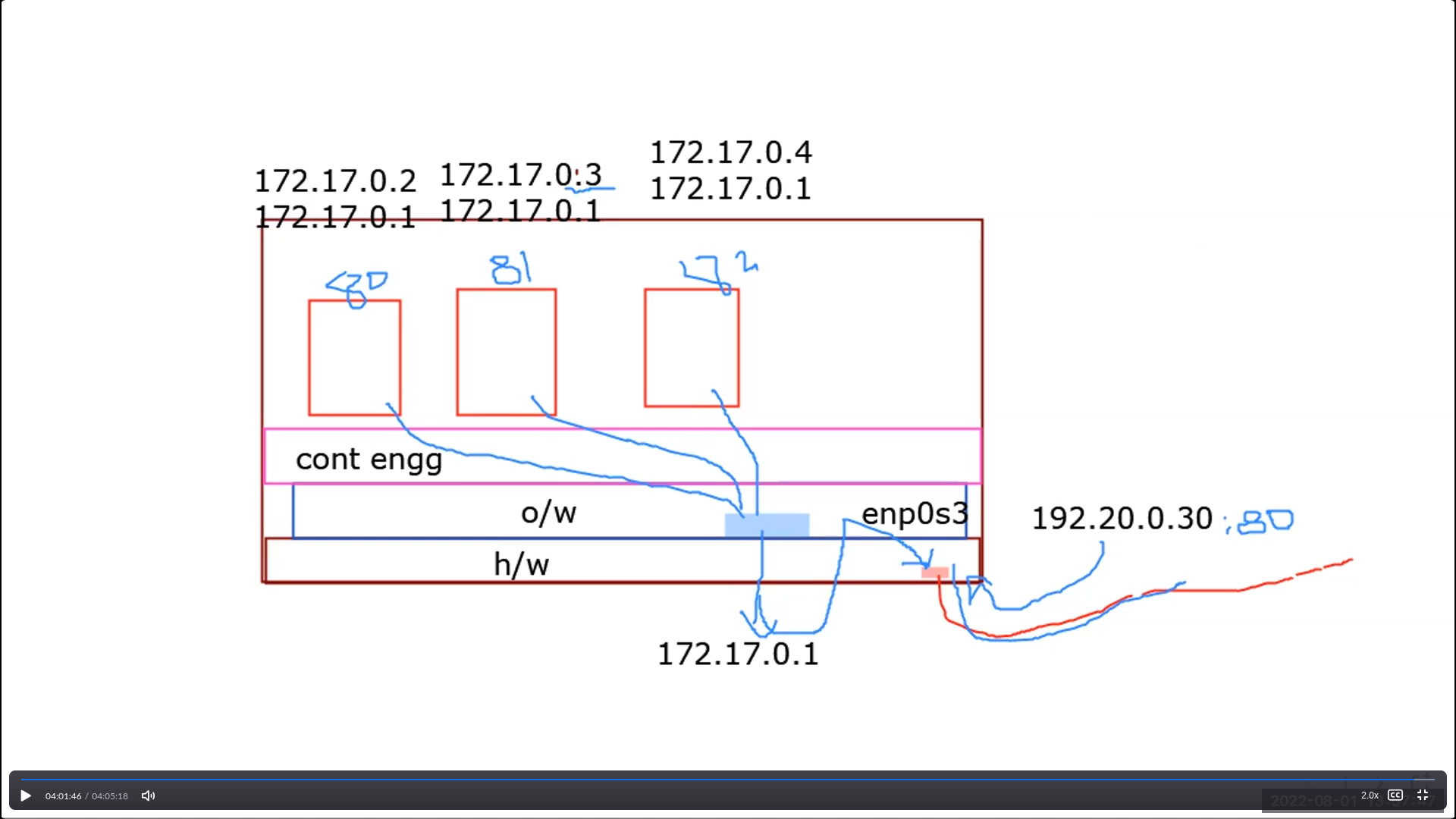
16 docker exec -it ub1 bash

17. cat /etc/os-release : to check os info

18. exit : exit from container

1 docker inspect web1 : to check containr info

When we are intall docker as container engine it wil create it own adapter along with default private ip adress which is to connected to the lan card adapter and work as a gateway for each container

* 

docker search redhat

* 36 docker pull redhat/ubi8
* 37 docker run -d -it --name r1 redhat/ubi8
* 38 docker images
* 39 docker ps
* 40 docker exec -it r1 bash

43 docker rmi -f $(docker images -q)

* 44 docker images
* 45\* docker psA
* 46 docker rm -f $(docker ps -a -q)
* 47 docker rmi -f $(docker images -q)
* 48 docker images
* 49 docker ps
* 50 docekr search httpd
* 51 docker search httpd
* 52 docker run -d --name web1 httpd: no need to pull image separately
* 53 docker images
* 54 docker ps
* 55 docker inspect web1 |grep -i ipaddr
* 56 docker exec -it web1 bash
* 58 curl 172.17.0.2

Explanation:

* 43 docker rmi -f $(docker images -q) : to remove all images
* 58 curl 172.17.0.2 : to check the browser from command line