Day 1: ›

<https://drive.google.com/file/d/1-ZdLogCkJOElMBSLfxd0hWCJlh3vupw7/view>

Graphical user interface

Description automatically generated with medium confidence

--Problem: ecommerce website, everyting needs to put in one server. This is

Monolithic Arichitecture server. If one server is down…need to fix. It might take 1 hour…or 2 days. In 2 days, we goona loose business. So monolithic architecture doesnot work.

So, Microservice based architecture, It means each component is working separately. Uber mobile app, it has 6 to 8 microservices are running. Parallelly at least 10,000 users are using uber app. So 10,000\* 8 =8,000 containers are used.

--In 2008 around, Kubernates came from Google. Google had a product name Bork. This Kubernages was owned by CNCF(cloud natives cloud free)

--GKE: Gooke kubernates engine

--EKS: Elastice kubernates service

--AKS: Azure Kubernates service

--To manage container, we need containered orchestration

Diagram

Description automatically generated

--Worker node:

Pods: abstraction on the top of docker container. One pot one docker container.

Master node:

Kube apiserver: verifies the user

Etcd:

Control manager:

Scheduler: which pod has to be deployed or taken away

Day 2:

<https://drive.google.com/file/d/1g97-YC-TBjA4CNu3jI28M_5SlB4mHbU5/view?usp=sharing>

--code of java or python need to compile and convert into machine readable language binary or artifacts.

--Docker: build, ship and run any software anywhere

---Virtualization: Cloud share resources but all applications are running independently

--Developer compiled and made artifact about the code and deploy earlier Now developer not only deploy the code but also he made containered package with code, libraries, environmen t, dependency.

--Each continers are isolated.

--Docker ensures that applications works seamlessly in any environment in dev, test or prod.

--Developer shipped all the code application along with its dependencies, libraries, configuratin files together in a package called containers.

--Container: Portable artifact, easily shared and moved around.

-- minkube start

* docker is available or not

--docker –version

--service docker status

--docker –help

--https://docs.docker.com/get-started/docker\_cheatsheet.pdf

Graphical user interface, diagram

Description automatically generated with medium confidence

--Registry: hub.docker.com

Username:

Password:

A picture containing text

Description automatically generated

--images: union of file systems, stagnant, the moment the images started, it became container.

Nginx image:

--nginx is reverse proxy of HTTPS. This reverse proxy blocks external virus. It saved websites from malware, unnecessary ciber attacks.

--hub.docker.com🡪type nginx in search

--docker images

# If you didn’t see nginx in images

--docker pull nginx:latest

--docker images

# In case we want to pull images from gcr, google cloud..we need to give path for images to pull

# Docker run with port forwarding, exposing external port. We cannot reuse the port. If we run same port ..it throws error.

-- docker run --name sample-nginx -d -p 8080:80 nginx:latest

--docker ps -a

-- docker run --name sample-nginx1 -d -p 9000:80 nginx:latest

--docker ps -a

**# show all process, what all softwares are running**

--docker ps -a

#remove docker image

--docker rm image\_id

--docker ps -a

--

#

--

Day 3:

<https://drive.google.com/file/d/1KjaFzMB8mwtgsFINpkdXquOR17qFAg8U/view?usp=sharing>

Diagram

Description automatically generated

Day4:

Day5:

Aws account creation: <https://github.com/vikas99341/AWS-Account-Creation>

Windows, java 8…..other user mac with java 11.