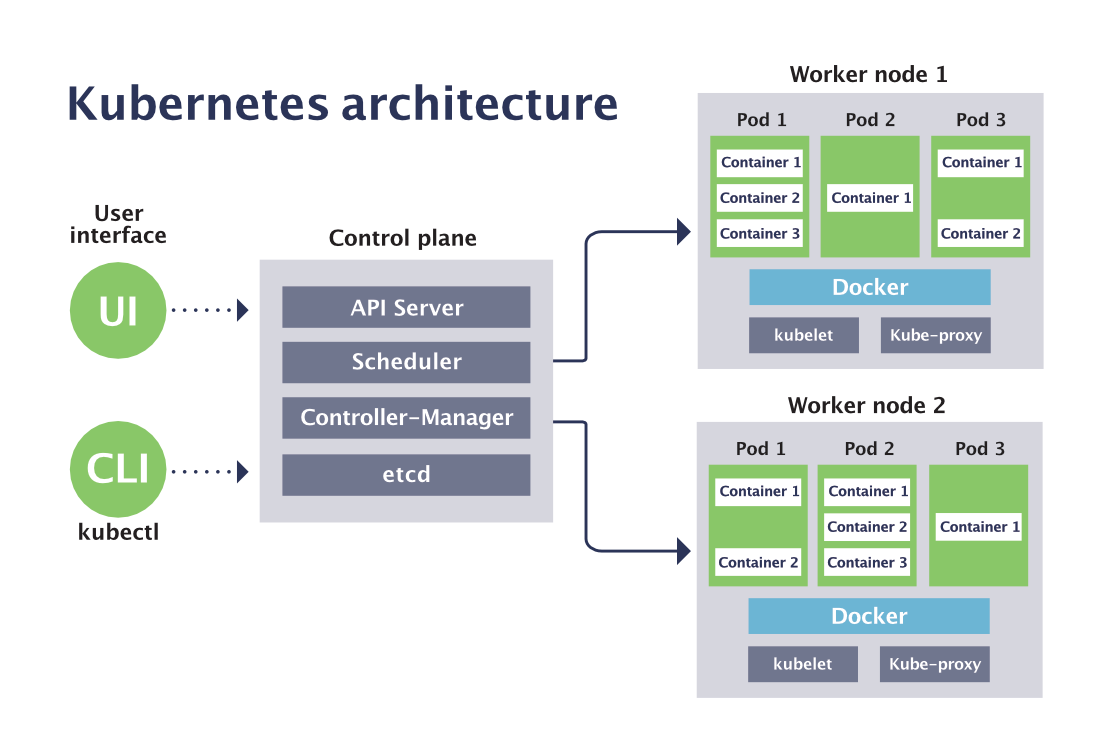
**Notes By R V K ROHINI PRIYA**

**KUBERNETES:**

Kubernetes is used for deploying, maintaining and scaling applications. It is one of the highly advanced technologies.

**Docker and Kubernetes Workflow:**

****

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Project:**

1. First we create an application and then compile with maven.
2. Then we convert the application into container based image.
3. Later we store it in the dockerhub by pushing it.
4. Finally we create a manifest file and run it.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Creating instance and login:**

Create instance from Amazon AWS Console

To log-in to ubuntu system

ssh -i keypair.pem ubuntu@<public-id>

To be a root user

sudo -i

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**To install Kubernetes:**

Be a root user first

sudo -i

Then do following commands

curl -sfL https://get.k3s.io | sh -

To get all the nodes available use this

kubectl get nodes

It shows the pods located in the default namespaces

kubectl get pod --all-namespaces

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**To create a pod:**

This is command is used to create a pod named my-first-pod

kubectl run my-first-pod --image stacksimplify/kubenginx:1.0.0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**To deploy a pod:**

To show the list of available pods

kubectl get pod

kubectl get pod my-first-pod

To get a brief description of a particular pod

kubectl describe pod my-first-pod

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Docker install:**

Now we install the docker into VM using

sudo apt-get update

sudo apt-get install ca-certificates curl gnupg lsb-release

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]

https://download.docker.com/linux/ubuntu $(lsb\_release -cs)

stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io

Finaly to know if docker is installed perfectly or not

It should show docker version

docker --version

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Install Java and Maven:**

Since we are building a maven based application I need to install java and maven

To install java first check by

javac

If not available then do following command

It installs any default version(for me jdk 11)

apt install default-jdk

To check java installation successful or not and its version

java --version

To install maven

apt install maven

To check maven

mvn --version

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Cloning from git:**

Now cloning a project from git into our instance to perform some operations:

git clone https://github.com/gopal1409/springboot-chat-app.git

To show the list of available projects at this point

ls

To go to a particular directory we use

cd <directory-name>

here we use

cd springboot-chat-app/

At this point ls shows available jar files

ls

mvn clean package

ls

To go to target directory

cd target

At this point ls shows available snapshots

ls

To go one step back from the directory

cd ..

To see what is available in a particular file we use

cat dockerfile

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Converting jar file into a container based image:**

To build (give a particular docker user name)

docker build -t rprvk/chatapp .

To see all available images

docker images

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Docker login and pushing images :**

(Before doing create account in hub.docker.com)

To login to docker

docker login

(It now asks for user name of dockerhub and the password. Password will be in shadow mode.

It is not visible while entering. So just give and enter. Then it shows login successful.)

To push your image to dockerhub

docker push <docker-id>/<application-name>

docker push rprvk/chatapp

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Now to run image in Kubernetes(K8s):**

First see availables pods

kubectl get pod

Run the pod

kubectl run my-chat-app --image rprvk/chatapp

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**To verify the pod is running image or not:**

kubectl describe pod my-chat-app