**PHASE 1: Run docker images of MySQL and microservices together independently**

**To work with MySQL docker image**

* To download the mysql image from docker hub repository

docker pull mysql:8.0.22

* One time activity to run the mysql with the following details:

root\_password=bjjd,

sql\_user=project-user,

sql\_password=pass379,

sql\_database=project-db

docker run --detach --env MYSQL\_ROOT\_PASSWORD=bjjd --env MYSQL\_USER=project-user --env MYSQL\_PASSWORD=pass379 --env MYSQL\_DATABASE=project-db --name mysql --publish 3306:3306 mysql:8.0.22

* Every time to run mysql

docker start mysql

* Download MySQL Workbench IDE for mysql8.0.22

https://dev.mysql.com/downloads/workbench/

**To work with microservice application: project-mgmt-service**

* To build the projectdownload the mysql image from docker hub repository

mvn clean install -DskipTestCase=true

* To build the docker image

docker image build -t project-mgmt-service:0.0.1-RELEASE .

* [Optional if requires to push the image in Docker]To tag and push the docker image into docker hub repository

docker tag project-mgmt-service:0.0.1-RELEASE rajivbansal2981/project-mgmt-service:0.0.1-RELEASE

docker push rajivbansal2981/project-mgmt-service:0.0.1-RELEASE

* To run the docker image of project-mgmt-service where RDS\_HOSTNAME is mysql (name of the mysql docker container)

docker run -p 5379:5379 --link=mysql --env RDS\_HOSTNAME=mysql project-mgmt-service:0.0.1-RELEASE

**PHASE 2: Using docker-compose to run docker images of MySQL and microservices in one go**

It would be diffucult for every time to remember these two long commands for running mysql docker and application docker.

1. To run the mysql docker:

docker run --detach --env MYSQL\_ROOT\_PASSWORD=bjjd --env MYSQL\_USER=project-user --env MYSQL\_PASSWORD=pass379 --env MYSQL\_DATABASE=project-db --name mysql --publish 3306:3306 mysql:8.0.22

2. To the application

docker run -p 5379:5379 --link=mysql --env RDS\_HOSTNAME=mysql project-mgmt-service:0.0.1-RELEASE

Docker-compose installation is required to run both docker container using one simple command

1. Download the docker-compose tool
2. Run this command to download the current stable release of Docker Compose:

sudo curl -L "https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

1. Apply executable permissions to the binary:

sudo chmod +x /usr/local/bin/docker-compose

1. If the command docker-compose fails after installation, check your path. You can also create a symbolic link to /usr/bin or any other directory in your path.

sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose

1. Test the installation.

$ docker-compose --version

Following are the steps to run both docker container using one simple command

1. Create docker-compose.yaml file in the project-mgmt-service project with the following contents:

version: '3.7'

services:

todo-web-application:

image: rajivbansal2981/project-mgmt-service:0.0.1-RELEASE

#build:

#context: .

#dockerfile: Dockerfile

ports:

- "5379:5379"

restart: always

depends\_on: # Start the depends\_on first

- mysql

environment:

RDS\_HOSTNAME: mysql

RDS\_PORT: 3306

RDS\_DB\_NAME: project-db

RDS\_USERNAME: project-user

RDS\_PASSWORD: pass379

mysql:

image: mysql:5.7

ports:

- "3306:3306"

restart: always

environment:

MYSQL\_ROOT\_PASSWORD: bjjd

MYSQL\_USER: project-user

MYSQL\_PASSWORD: pass379

MYSQL\_DATABASE: project-db

volumes:

- mysql-database-data-volume:/var/lib/mysql

# Volumes

volumes:

mysql-database-data-volume:

1. Run the following command from project-mgmt-service location.

cd /home/rajiv/git/BJJD-Kubernetes/project-mgmt-service

rajiv@rajiv-VirtualBox:~/git/BJJD-Kubernetes/project-mgmt-service$ docker-compose up

1. It will start the mysql and then run the application and after that we can use the application

**PHASE 3: Using minikube, a local kubernetes focusing on making it easy to** **push and run docker images of MySQL and microservices in Kubernetes cluster**

minikube is local Kubernetes, focusing on making it easy to learn and develop for Kubernetes.

All you need is Docker (or similarly compatible) container or a Virtual Machine environment, and Kubernetes is a single command away: minikube start

curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

sudo install minikube-linux-amd64 /usr/local/bin/minikube

To install kubeadm and kubectl

sudo apt-get update && sudo apt-get install -y apt-transport-https curl

curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list

deb https://apt.kubernetes.io/ kubernetes-xenial main

EOF

sudo apt-get update

sudo apt-get install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl

Following are the steps to pull and deploy the images of mysql and aplication docker into kubernetes.

1. minikube start --extra-config=kubeadm.ignore-preflight-errors=NumCPU --force --cpus 1
2. minikube stop
3. kubectl apply -f project-mgmt-service-deployment.yaml
4. kubectl apply -f mysql-deployment.yaml

#### Logging into the MySQL pod

You can get the MySQL pod and use kubectl exec command to login to the Pod.

$ kubectl get pods

NAME READY STATUS RESTARTS AGE

polling-app-mysql-6b94bc9d9f-td6l4 1/1 Running 0 4m23s

$ kubectl exec -it polling-app-mysql-6b94bc9d9f-td6l4 -- /bin/bash

root@polling-app-mysql-6b94bc9d9f-td6l4:/#

1. kubectl delete all -l app=project-mgmt-service
2. kubectl get events --sort-by=.metadata.creationTimestamp