1. docker run -p 5000:5000 in28min/todo-rest-api-h2:1.0.0.RELEASE -Runs a docker container {hostport}:{containerPort}
2. docker run -p 5000:5000 -d in28min/todo-rest-api-h2:1.0.0.RELEASE -Runs a docker container {hostport}:{containerPort} , -d runs in detached mode
3. docker container ls -to see the list of all the containers running
4. docker logs {containerid} – to see logs of a container
5. docker logs -f {containerid} – start tailing the logs
6. docker images – to see all the local images
7. docker container ls -a - to see the list of all the containers (running and stopped)
8. docker container stop {containerid} – to stop a running container

**STEPS TO RUN AND DEPLOY SPRING BOOT APP ON DOCKER CONTAINER**

1. create a docker container for mysql.

run -d -p 2012:3306 --name mysql-docker-container -e MYSQL\_ROOT\_PASSWORD=root123 -e MYSQL\_DATABASE=spring\_app\_db -e MYSQL\_USER=app\_user -e MYSQL\_PASSWORD=test123 mysql:latest

1. create a Dockerfile in root of the source directory. **Dockerfile** (instructions in the Dockerfile) will be used by the**docker build** (command) when building the **Docker Image**.

**FROM** java:8

**LABEL** maintainer=“chathuranga.t@gmail.com”

**VOLUME** /tmp

**EXPOSE** 8080

**ADD** target/spring-boot-data-jpa-example-0.0.1-SNAPSHOT.jar spring-boot-data-jpa-example-0.0.1-SNAPSHOT.jar

**ENTRYPOINT** ["java","-jar","spring-boot-data-jpa-example-0.0.1-SNAPSHOT.jar"]

**FROM** – This defines the base image for the image that we are building.(The image should be built from this base image). All we said is that, for this image to run, we need the java:8 image.

**EXPOSE**–  This specified the port number on which the docker container is running. The docker host will be informed about this port when the container is booting up.

**VOLUME**

We added a VOLUME pointing to “/tmp” because that is where a Spring Boot application creates working directories for Tomcat by default. The effect is to create a temporary file on your host under “/var/lib/docker” and link it to the container under “/tmp”. This step is optional for the simple app that we wrote here, but can be necessary for other Spring Boot applications if they need to actually write in the filesystem.

**ADD**– Adding the files into the docker image being created. Normally this command will be sued to add executable jar files into the docker image.

**ENTRYPOINT** : The specified command will get executed when the container is booted up.

4.Build the docker image of the app with following command

docker build -f Dockerfile -t spring-jpa-app .

5.Run the app in a container and link it to mysql container

docker run -t --name spring-jpa-app-container --link mysql-docker-container:mysql -p 8087:8080 spring-jpa-app

6.verify that containers are linked properly by logging on to app container in bash mode

docker exec -it **spring-jpa-app-container** bash

and verify entry of linked container in etc/hosts file.