# Title: Docker usage documentation

**To make only freshsales run from the instance :**

docker-compose up -d

Here -d means detached. So that it won't exit when you press ctrl+c. (While initial setup i.e. building images, it will exit when you press ctrl+c.)

Use container logs to get the information regarding starting of containers.

Note\* : Please use **docker::deploy**recipe from opsworks for clear deployment. This process is not going to deploy new code. It will restart all the stopped containers only.

**To bootstrap database both admin console and main app via recipes (works only in stacks having docker\_v3 recipes):**

Before doing bootstrap, there is a hack to perform. please do this first.

Comment out these 3 lines by editing **/data/freshsales/config/environments/staging.rb**.

# config.after\_initialize do  
  # instrument  
# end

Stack → Run Command → Execute Recipes → **docker::db\_bootstrap** with this custom json (Advanced → Custom Chef JSON) :

{  
  "bootstrap": {  
    "apps": [  
      "freshsales",  
      "freshsales\_admin"  
    ],  
    "domain": "development",  
    "email": "madhava.sanikommu@freshworks.com"  
  }  
}

**Note\* :**Here sed -i will simply replace BOOTSTRAP variable in yml file. It means simply editing the file and changing it manually to TRUE. Run all these commands in bash.

**To bootstrap database both admin console and main app:**

sed -i 's/BOOTSTRAP=false/BOOTSTRAP=true/g' /data/freshsales/docker-compose.yml /data/freshsales\_admin/docker-compose.yml

docker stop fs-web

docker stop fs-admin-web

sed -i 's/instrument/# instrument/' /data/freshsales/config/environments/staging.rb

docker-compose -f /data/freshsales/docker-compose.yml up -d

docker-compose -f /data/freshsales\_admin/docker-compose.yml up -d

sed -i 's/BOOTSTRAP=true/BOOTSTRAP=false/g' /data/freshsales/docker-compose.yml /data/freshsales\_admin/docker-compose.yml

**Main App Bootstrap :**

sed -i 's/BOOTSTRAP=false/BOOTSTRAP=true/g' /data/freshsales/docker-compose.yml

docker stop fs-web

sed -i 's/instrument/# instrument/' /data/freshsales/config/environments/staging.rb

docker-compose -f /data/freshsales/docker-compose.yml up -d

sed -i 's/BOOTSTRAP=true/BOOTSTRAP=false/g' /data/freshsales/docker-compose.yml

**Admin Console Bootstrap :**

sed -i 's/BOOTSTRAP=false/BOOTSTRAP=true/g' /data/freshsales\_admin/docker-compose.yml

docker stop fs-admin-web

docker-compose -f /data/freshsales/docker-compose.yml up -d

sed -i 's/BOOTSTRAP=true/BOOTSTRAP=false/g' /data/freshsales\_admin/docker-compose.yml

**To list containers :**

docker ps -a

Here -a is going to list stopped containers also. You can skip passing -a also.

**To start/stop a container :**

docker start/stop <container\_id>

**To get in to the container :**

docker exec -it <container\_id> <command>

Here command can be anything : bash, rails c, bundle exec rails c etc.

**To get the logs from the container :**

docker logs --tail <num\_of\_lines> -f <container\_id>

Here,

--tail <num\_of\_lines> going to get the last num\_of lines you requested.

-f is going to follow the output of the docker container till it stopped.

**Advanced Material :**

**When you are getting fs\_read\_only not authorized to access database error :**

Run this in mysql console or in sqlpro :  (get user, pass from stack settings and use host as instance IP)

CREATE USER 'fs\_readonly\_user'@'%' IDENTIFIED BY '<Password Here>';  
GRANT SELECT ON \*.\* TO 'fs\_readonly\_user'@'%';  
GRANT EXECUTE ON \*.\* TO 'fs\_readonly\_user'@'%';  
FLUSH privileges;

**To List all the images :**

docker images

**To delete a container :**

docker rm <container\_id> <container\_id> ...

Here, container\_id(s) with spaces going to remove multiple containers at once.

**To delete an image :**

docker rmi <image\_id> <image\_id> ...

Here, image\_id(s) with spaces going to remove multiple images at once.

**To boot up a new container so that you can debug the command thats causing its failure :**

docker run -it --rm <image\_id> <command>

Here,

--rm means when you exit, remove the container created temporarily for debug purposes. if you didn't use it, it's going to be there permanently.

-it means interactive and tty based. So that it will allocate a tty for you and gives access to write and read from docker container.

**Instance setup is hanged in opsworks :**

Login to the instance as root user and perform these tasks. This will be occured because of force killing of docker daemon. Because of force killing, Docker daemon can't able to write container's state info to files. So the state.json files in containerd folder won't exist hence the hang of all the docker commands. Fix is like :

service docker stop

rm -rf /var/run/docker/libcontainerd/containerd/\*

service docker start

if [[ ! -z $(docker ps -aq) ]]; then docker rm $(docker ps -aq); fi;