



Appl Containerization and Orchestration

Experiment 5

AIM: Working with Dockerfile to Build and Push Docker Image

Submitted by-
Akhand Pratap Singh

Submitted to:
Mr. Hitesh Kumar Sharma

Steps to Complete:

1. Create following Dockerfile

```
FROM alpine
MAINTAINER SUDIPT
RUN apk update
RUN apk add nodejs
RUN mkdir /app
WORKDIR /app
```

Screenshot :

```
[alpha@Akhands-MacBook-Pro R2142201498 % pwd
/Users/alpha/Desktop/college/ACOLAB/R2142201498
[alpha@Akhands-MacBook-Pro R2142201498 % touch Dockerfile
[alpha@Akhands-MacBook-Pro R2142201498 % vi Dockerfile
```

```
~/Desktop/college/ACOLAB/R2142201498 -
```

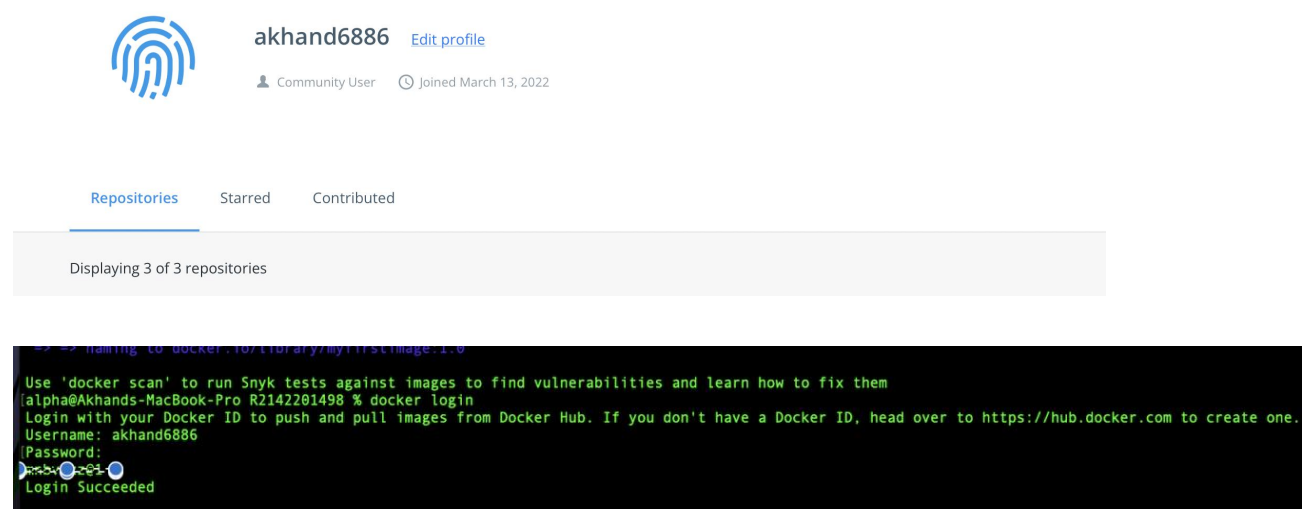
```
FROM alpine
MAINTAINER SUDIPT
RUN apk update
RUN apk add nodejs
RUN mkdir /app
WORKDIR /apps
```

2. Now we have dockerized the app, we will Build image from Dockerfile.

Command : `docker build -t myfirstimage:1.0 .`

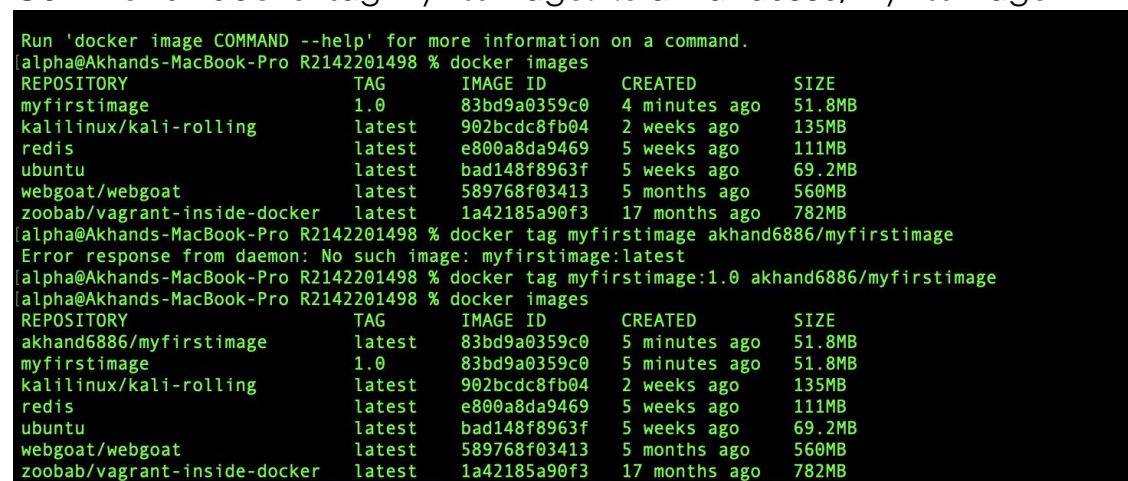
```
[alpha@Akhands-MacBook-Pro R2142201498 % vi Dockerfile
[alpha@Akhands-MacBook-Pro R2142201498 % docker build -t myfirstimage:1.0 .
[+] Building 170.4s (9/9) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 137B
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [internal] load metadata for docker.io/library/alpine:latest
[1/5] FROM docker.io/library/alpine@sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad
=> resolve docker.io/library/alpine@sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad
=> sha256:bc41182d7ef5ffc53a40b044e725193bc10142a1243f395ee852a8d9730fc2ad 1.64kB / 1.64kB
=> sha256:ed73e2bee79b3428995b16fce4221fc715a849152f364929cdccdc83db5f3d5c 528B / 528B
=> sha256:a6215f271958c760a2975a6765016044115dbae4b90f414eba3a448a6a26b4f6 1.49kB / 1.49kB
=> sha256:9b18e9b68314027565b90ff6189d65942c0f7986da80df008b8431276885218e 2.71MB / 2.71MB
=> extracting sha256:9b18e9b68314027565b90ff6189d65942c0f7986da80df008b8431276885218e
[2/5] RUN apk update
[3/5] RUN apk add nodejs
[4/5] RUN mkdir /app
[5/5] WORKDIR /apps
=> exporting to image
=> exporting layers
=> writing image sha256:83bd9a0359c072898586d8dea2de3660bfc751538d9d7cd177d0a512895f3340
=> naming to docker.io/library/myfirstimage:1.0
```

3. Create account on Dockerhub and create a repository in it.



The screenshot shows the Docker Hub profile for user 'akhand6886'. The profile includes a fingerprint icon, the username 'akhand6886', a link to 'Edit profile', and the status 'Community User' with a join date of 'March 13, 2022'. Below the profile information, there are tabs for 'Repositories', 'Starred', and 'Contributed'. The 'Repositories' tab is active, showing 'Displaying 3 of 3 repositories'. Below this, a terminal window shows the command 'docker login' being executed successfully, with the username 'akhand6886' and password masked by dots.

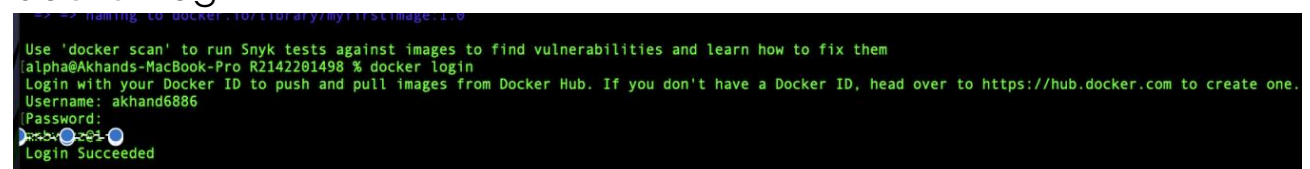
4. Tag the recently created image using following command.
Command : `docker tag myfirstimage:1.0 akhand6886/myfirstimage`



The screenshot shows a terminal window with the following output:

```
Run 'docker image COMMAND --help' for more information on a command.
alpha@Akhands-MacBook-Pro R2142201498 % docker images
REPOSITORY          TAG          IMAGE ID      CREATED       SIZE
myfirstimage         1.0          83bd9a0359c0  4 minutes ago 51.8MB
kalilinux/kali-rolling latest       902bcdc8fb04  2 weeks ago  135MB
redis                latest       e800a8da9469  5 weeks ago  111MB
ubuntu              latest       bad148f8963f  5 weeks ago  69.2MB
webgoat/webgoat      latest       589768f03413  5 months ago 560MB
zoobab/vagrant-inside-docker latest       1a42185a90f3  17 months ago 782MB
alpha@Akhands-MacBook-Pro R2142201498 % docker tag myfirstimage akhand6886/myfirstimage
Error response from daemon: No such image: myfirstimage:latest
alpha@Akhands-MacBook-Pro R2142201498 % docker tag myfirstimage:1.0 akhand6886/myfirstimage
alpha@Akhands-MacBook-Pro R2142201498 % docker images
REPOSITORY          TAG          IMAGE ID      CREATED       SIZE
akhand6886/myfirstimage latest       83bd9a0359c0  5 minutes ago 51.8MB
myfirstimage         1.0          83bd9a0359c0  5 minutes ago 51.8MB
kalilinux/kali-rolling latest       902bcdc8fb04  2 weeks ago  135MB
redis                latest       e800a8da9469  5 weeks ago  111MB
ubuntu              latest       bad148f8963f  5 weeks ago  69.2MB
webgoat/webgoat      latest       589768f03413  5 months ago 560MB
zoobab/vagrant-inside-docker latest       1a42185a90f3  17 months ago 782MB
```

5. Login to Dockerhub from console using following command.
`docker Login`



The screenshot shows a terminal window with the following output:

```
alpha@Akhands-MacBook-Pro R2142201498 % docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: akhand6886
Password:
Login Succeeded
```

6. Now push image on Dockerhub using following command.

Command : docker push akhand6886/myfirstimage

```
alpha@Akhands-MacBook-Pro R2142201498 % docker push akhand6886/myfirstimage
Using default tag: latest
The push refers to repository [docker.io/akhand6886/myfirstimage]
699c1d2fa6f5: Pushed
c65b2ffc209b: Pushed
a5d6568a5fa5: Pushed
f23f4a70b4f8: Pushed
5d3e392a13a0: Mounted from library/alpine
latest: digest: sha256:08dd008fcf2b0305c21f0cc74c450c5533e55da9617d8d80366e424b7e347abe size: 1365
alpha@Akhands-MacBook-Pro R2142201498 % docker run - akhand6886/myfirstimage
```

7. Pull and Run the container of your deployed image on docker hub.

Command : Docker run -d akhand6886/myfirstimage

```
alpha@Akhands-MacBook-Pro R2142201498 % docker run - akhand6886/myfirstimage
docker: invalid reference format.
See 'docker run --help'.
alpha@Akhands-MacBook-Pro R2142201498 % docker run -d akhand6886/myfirstimage
5cc862c3e7d6b095a847fd085dc8f040094b9c9e1916f8a8b866aa53580aa11a
alpha@Akhands-MacBook-Pro R2142201498 % docker images
```