

St. Francis Institute of Technology
Department of Computer Engineering
Five Days Student Development Programme
on
DevOps
(28th June to 2nd July 2021)

Assignment 02

Docker Task

1. Create a Docker Hub Account (<https://hub.docker.com/>)
2. Create a Dockerfile for your Mini Project or choose from a simple Java Application/ Web application or application of your choice.
3. Create an Image using that Dockerfile and Containerize your application.
(Name of the Image and Container should be (full name-roll no))
4. Push the image on Docker Hub Repository and provide the link to it in the report.
5. Take screenshots of all the steps and prepare a report for the same.

NAME: Saravana sundar Nadar

ROLL NO:72

PARTICIPANT ENROLLMENT ID:192072

CLASS: SE CMPN A

Link for my docker hub repository :

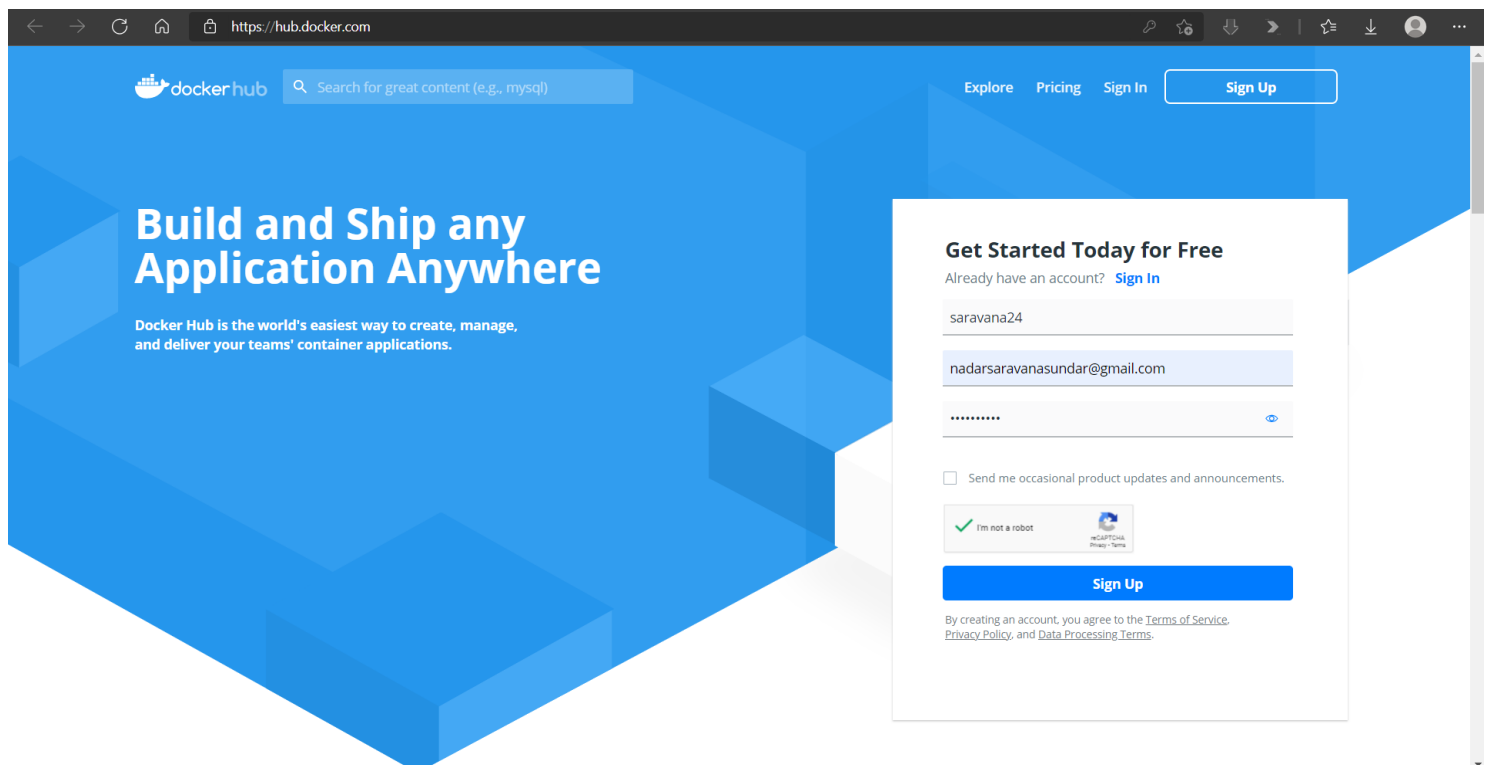
<https://hub.docker.com/repository/docker/saravana24/saravana-sundar-nadar-192072>

Docker

Steps to create Docker Hub Repository.

1. Download Docker Desktop.

Create a Docker Hub Account ([Docker Hub](https://hub.docker.com)) and Download and install Docker Desktop Application .

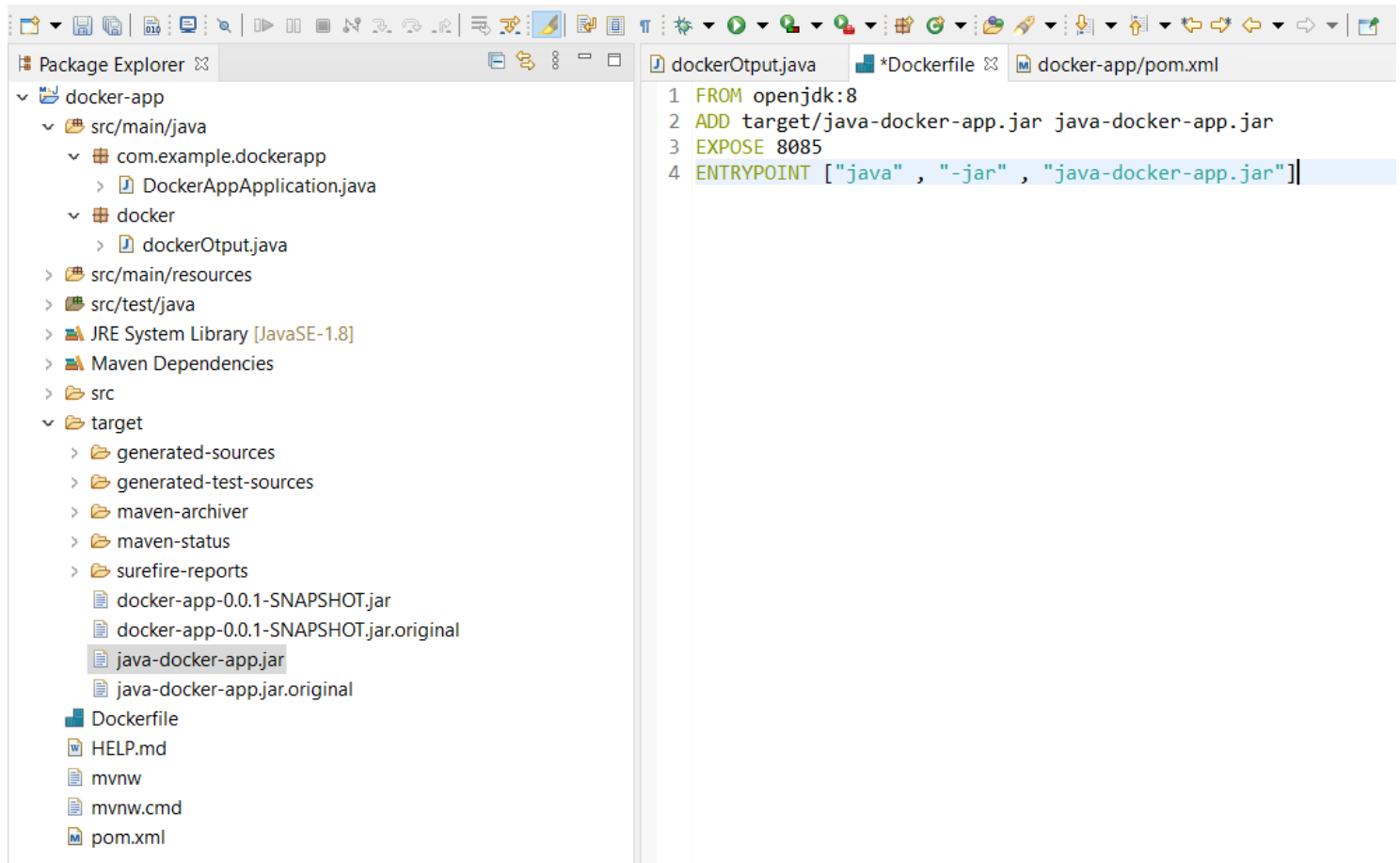


2. Create a Docker File of the Simple Java Application.

In this step, you write a Dockerfile that builds a Docker image. The image contains all the dependencies the Java application requires, including Java JDK itself.

In your project directory, create a file named Dockerfile.

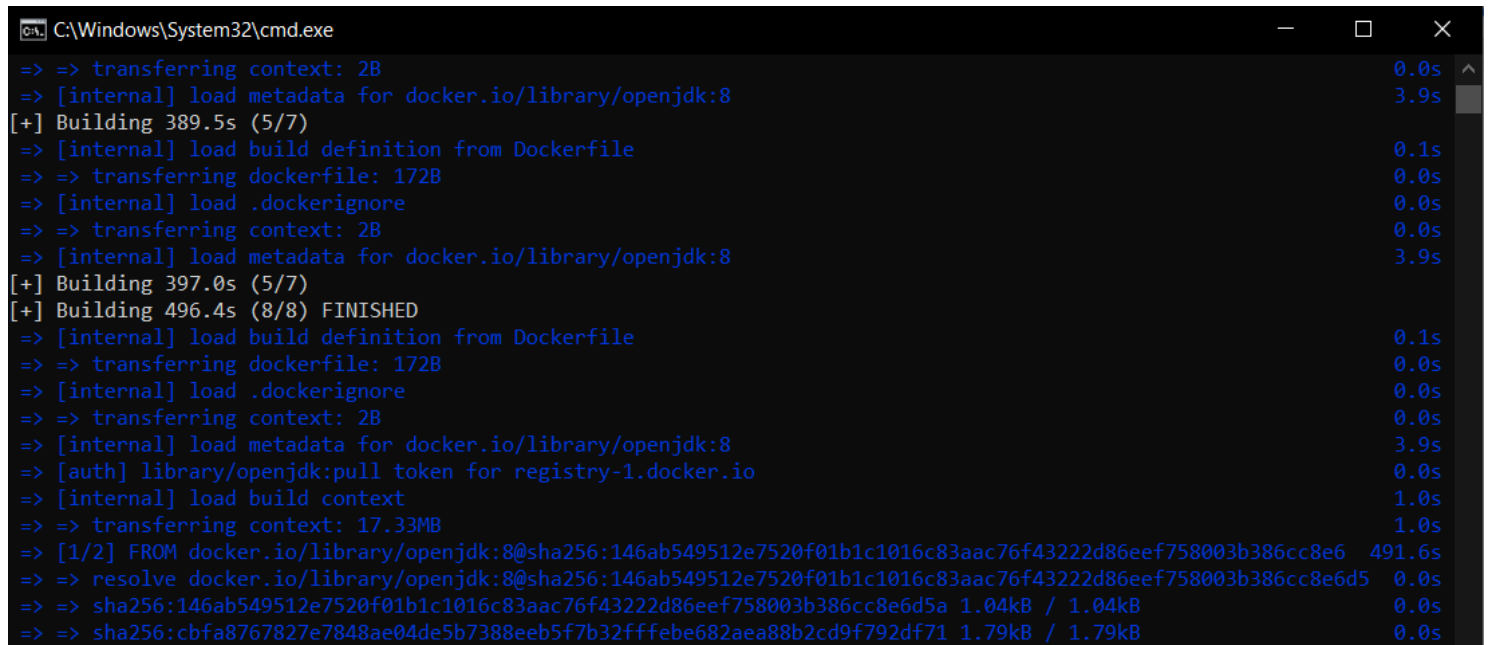
Also Clean Build Your Java application and create a jar file of your Application.



3. Create Docker image.

Create a docker image by using the command : `docker build -f Dockerfile -t saravana-sundar-192072 .`

in the terminal. It takes a while for downloading the mentioned BaseImage - "openjdk:8".



```
=> exporting to image 0.2s
=> => exporting layers 0.2s
=> => writing image sha256:bb81c35596b6fff9e9b58a61a221c177bf146bbbc2ea2b9ea16978db4fa6b6f8 0.0s
=> => naming to docker.io/library/saravana-sundar-192072 0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them

E:\DevOps Workshop\Docker\docker-app>docker image ls
REPOSITORY          TAG          IMAGE ID      CREATED        SIZE
saravana-sundar-192072  latest      bb81c35596b6  4 minutes ago  531MB
docker101tutorial    latest      8aa870ba39d3  4 hours ago   28.2MB
saravana24/docker101tutorial latest      8aa870ba39d3  4 hours ago   28.2MB
alpine/git           latest      b8f176fa3f0d  6 weeks ago   25.1MB

E:\DevOps Workshop\Docker\docker-app>
```

4. Create a container using Image.

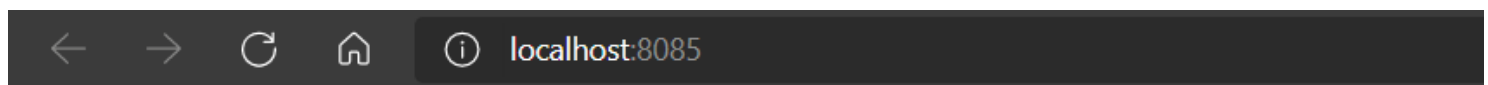
Run the Docker Image to obtain a Container for your application.

Use command : `docker run -p 8085:8085 saravana-sundar-192072`

```
E:\DevOps Workshop\Docker\docker-app>docker image ls
REPOSITORY          TAG          IMAGE ID      CREATED        SIZE
saravana-sundar-192072  latest      dee945c35da7  21 seconds ago  531MB
<none>              <none>      bb81c35596b6  14 minutes ago  531MB
docker101tutorial    latest      8aa870ba39d3  4 hours ago   28.2MB
saravana24/docker101tutorial latest      8aa870ba39d3  4 hours ago   28.2MB
alpine/git           latest      b8f176fa3f0d  6 weeks ago   25.1MB

E:\DevOps Workshop\Docker\docker-app>docker run -p 8085:8085 saravana-sundar-192072
```

5. Run Your Java Application locally in the browser .



Hello Docker - Sraravana sundar Nadar - 192072

6. Push Your Local image to Dockerhub remote repository.

Add a tag before pushing.

Use Command : `docker push saravana-sundar-192072`

```
E:\DevOps Workshop\Docker\docker-app>docker images
REPOSITORY              TAG                IMAGE ID           CREATED            SIZE
saravana-sundar-nadar-192072  latest            7ad14a1fae4d      26 minutes ago    531MB
saravana24/saravana-sundar-nadar-192072  first-push        7ad14a1fae4d      26 minutes ago    531MB
docker101tutorial         latest            8aa870ba39d3      18 hours ago      28.2MB
saravana24/docker101tutorial  latest            8aa870ba39d3      18 hours ago      28.2MB
alpine/git                 latest            b8f176fa3f0d      6 weeks ago       25.1MB

E:\DevOps Workshop\Docker\docker-app>docker push saravana24/saravana-sundar-nadar-192072:first-push
The push refers to repository [docker.io/saravana24/saravana-sundar-nadar-192072]
7fe1bdafc249: Layer already exists
aed5f5426b27: Layer already exists
c3d00b097cce: Layer already exists
79c550eb7bd2: Layer already exists
7095af798ace: Layer already exists
fe6a4fdbedc0: Layer already exists
e4d0e810d54a: Layer already exists
4e006334a6fd: Layer already exists
first-push: digest: sha256:3c6d0a6744e019e75ca229715240609fb1b1e77e7e3ae78f0aa0decec013d9d0 size: 2006

E:\DevOps Workshop\Docker\docker-app>
```

[←](#) [→](#) [↺](#) [🏠](#) <https://hub.docker.com/repository/docker/saravana24/saravana-sundar-nadar-192072> [🔖](#) [⬇](#) [▶](#) [🔍](#) [⌵](#)

Advanced Image Management
View all your images and tags in this repository, clean up unused content, recover untagged images. Available for Pro and Team accounts. [View preview](#)

saravana24 / saravana-sundar-nadar-192072
DevOps Workshop Assignment - DOCKERHUB TASK - 192072
 Last pushed: a few seconds ago

Docker commands [Public View](#)
To push a new tag to this repository,

```
docker push saravana24/saravana-sundar-nadar-192072:tagname
```

Tags and Scans **VULNERABILITY SCANNING - DISABLED** [Enable](#)
This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
first-push		7 minutes ago	a few second...

[See all](#)

Automated Builds
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.
Available on Pro and Team plans.
[Upgrade to Pro](#) [Learn more](#)

Readme
<http://localhost:8085/>

Hence, Remote Docker hub repository is Created.

Link for my docker hub repository :

<https://hub.docker.com/repository/docker/saravana24/saravana-sundar-nadar-192072>