

Virtualization and Cloud Computing, VCL
2nd Year Specialty SIQ G02, 2CS SIQ2

LAB5A Report

Containerization with Docker

Studied by:

HEDDADJI Nour El Imane
SOLTANI MERIEM

E-mails :

jn_heddadji@esi.dz
jm_soltani@esi.dz

Contents

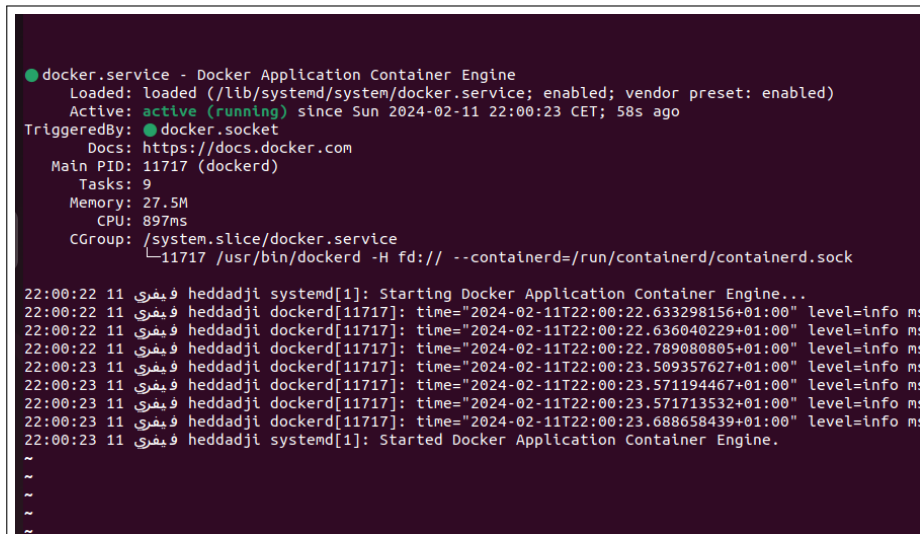
1.	Docker Installation	3
2.	Getting familiar with Docker	3
2.1.	Hello-world Image	3
2.2.	Search for an Image in the Registry	4
2.3.	Pulling an image	4
2.4.	Listing Local Docker Images	4
2.5.	Docker groups	5
3.	Managing Containers	5
4.	Custom Docker images	6
4.1.	Dockerfile Structure	6
4.2.	Building and Running the Custom Image	7
5.	Hosting a Python/Flask app on a docker	7
5.1.	Building and Running the Docker Image	9
5.2.	Hosting the Docker image to Docker Hub	10

1. Docker Installation

To proceed with this lab, we created the following:

- An Ubuntu server.
- A Docker Hub account

```
1 sudo apt update
2 sudo apt install apt-transport-https ca-certificates curl software-
3 properties-common
4 curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key
5 add -
6 sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/
7 linux/ubuntu bionic stable"
8 sudo apt install docker-ce
9 #Verify
10 sudo systemctl status docker
```



```
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2024-02-11 22:00:23 CET; 58s ago
     TriggeredBy: ● docker.socket
    Docs: https://docs.docker.com
   Main PID: 11717 (dockerd)
      Tasks: 9
     Memory: 27.5M
        CPU: 897ms
    CGroup: /system.slice/docker.service
            └─11717 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

22:00:22 11 هيدادجي systemd[1]: Starting Docker Application Container Engine...
22:00:22 11 هيدادجي dockerd[11717]: time="2024-02-11T22:00:22.633298156+01:00" level=info ms
22:00:22 11 هيدادجي dockerd[11717]: time="2024-02-11T22:00:22.636040229+01:00" level=info ms
22:00:22 11 هيدادجي dockerd[11717]: time="2024-02-11T22:00:22.789080805+01:00" level=info ms
22:00:23 11 هيدادجي dockerd[11717]: time="2024-02-11T22:00:23.509357627+01:00" level=info ms
22:00:23 11 هيدادجي dockerd[11717]: time="2024-02-11T22:00:23.571194467+01:00" level=info ms
22:00:23 11 هيدادجي dockerd[11717]: time="2024-02-11T22:00:23.571713532+01:00" level=info ms
22:00:23 11 هيدادجي dockerd[11717]: time="2024-02-11T22:00:23.688658439+01:00" level=info ms
22:00:23 11 هيدادجي systemd[1]: Started Docker Application Container Engine.
~
~
~
~
~
```

2. Getting familiar with Docker

2.1. Hello-world Image

```
1 docker container run hello-world
2
3 .
```

```
heddadj@heddadj:~$ sudo docker container run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/

heddadj@heddadj:~$
```

2.2. Search for an Image in the Registry

This command searches the Docker Hub registry for images related to "example".

It returns a list of images available on the registry that match the search term.

```
1
2 docker search example
3 .
```

2.3. Pulling an image

Pulling an image means **downloading** it to your local machine, making it available for local use.

```
1
2 docker image pull image_name
3 .
```

2.4. Listing Local Docker Images

```
1
2 docker image ls
3 .
```

```

heddadjl@heddadjl:~$ sudo docker search ns2
[sudo] password for heddadjl:
NAME                                DESCRIPTION                                STARS     OFFICIAL   AUTOMATED
ns208/guestbook-php-frontend        0
tikz/ns2-roundend                   0
tikz/ns2-skill-go                   0
tikz/ns2-discord                     0
tikz/ns2sud-web                     0
ns208/centos6-sshd                  0
tikz/ns2-web                         0
ns208/mycentos-ssh                  0
ekiourk/ns2                         Network Simulator 2.35                    3
tikz/ns2-skilll                     0
zenithal/ns2discordbot              0
massimilianomirelli/ns2_server     0
ns29/get-started                    0
serverboi/ns2                       0
ns265422/julia                      0
ns2808/cheers2019                   0
ns29/cloudserver                    0
christofferbraun/ns2-jenkins-postgresql 0
npranav10/ns2                      0
eslon/ns2_server-docker             Docker Image for Natural Selection 2 Server 0
dotrdg/ns2                          0
kersey/ns2.35                       Comes with GNUPlot                       0
jangellinav/ns2                    0
gcasella/ns2-docker                 0
ns2rk/docker101tutorial             0

```

```

heddadjl@heddadjl:~$ sudo docker image pull ekiourk/ns2
Using default tag: latest
latest: Pulling from ekiourk/ns2
Image docker.io/ekiourk/ns2:latest uses outdated schema1 manifest format. Please upgrade to a schema2 image for better
ibility. More information at https://docs.docker.com/registry/spec/deprecated-schema-v1/
a3ed95caeb02: Pull complete
2d7928839f96: Pull complete
2cbe3e3fa432: Pull complete
76a59787cc98: Pull complete
2eb92b26d2c9: Pull complete
0375c530f5ff: Pull complete
d5da3d004d16: Pull complete
cdfbbc36a11a: Pull complete
Digest: sha256:8c2764a3f35baa783af4dde92c257c6221b8f45279ccf1e84ca21c526df26ef1
Status: Downloaded newer image for ekiourk/ns2:latest
docker.io/ekiourk/ns2:latest
heddadjl@heddadjl:~$ sudo docker image ls
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    d2c94e258dcb   9 months ago   13.3kB
ekiourk/ns2    latest    afe271929563   8 years ago    743MB

```

2.5. Docker groups

To avoid using sudo each time we run a Docker command.

```

1
2 sudo usermod -aG docker your_username
3 groups
4 .

```

3. Managing Containers

```

1
2 # Start an interactive Debian container and launch a Bash shell
3 docker container run -it debian /bin/bash
4
5 # Start an interactive Debian container with automatic removal and
6 assign a custom name
7 docker container run -it --rm --name mydebian debian /bin/bash
8

```

9 .

```
heddadjl@heddadji:~$ sudo docker container run -it debian /bin/bash
[sudo] password for heddadjl:
Unable to find image 'debian:latest' locally
latest: Pulling from library/debian
6a299ae9cfd9: Pull complete
Digest: sha256:79becb70a6247d277b59c09ca340bbe0349af6aacb5afa90ec349528b53ce2c9
Status: Downloaded newer image for debian:latest
root@01db806ece14:/# exit
exit
heddadjl@heddadji:~$ sudo docker container run -it --rm --name mydebian debian /bin/bash
root@05a634549468:/# exit
exit
heddadjl@heddadji:~$
```

4. Custom Docker images

A Dockerfile is a text file that contains instructions for creating a Docker image. Instructions are written in the format: **DIRECTIVE argument**.

4.1. Dockerfile Structure

```
1
2 #Create the folder for the project
3 mkdir app
4 # Create dockerfile
5 touch Dockerfile
6 nano Dockerfile
7 # Opens the Dockerfile in the nano text editor
8 .
```

This is how a dockerfile should look like :

```
1
2 # Specify the base image
3 FROM your_base_image:tag
4
5 # Set the working directory
6 WORKDIR /path_to_working_directory
7
8 # Install dependencies
9 RUN your_package_manager install your_dependencies
10
11 # Copy application files to the container
12 COPY . /app
13
14 # Specify environment variables
```

```
15 ENV YOUR_ENV_VARIABLE=value
16
17 # Define the default command to run when the container starts
18 CMD ["your_command", "--your-option=option_value"]
19
20 .
```

4.2. Building and Running the Custom Image

Open a terminal and navigate to the directory containing the Dockerfile.

```
1 # Build the Docker image
2 docker build . -t my-custom-image:v1
3
4 # Check the list of Docker images
5 docker image ls
6
7 # Run the Docker container
8 docker container run -d my-custom-image :v1
9
10
11 .
```

5. Hosting a Python/Flask app on a docker

We will create these files :

- app.py : to store the code for our flask app
- Dockerfile

Flask file : app.py

```
1 from flask import Flask, render_template_string
2
3 app = Flask(__name__)
4
5 @app.route('/')
6 def hello_world():
7     html_content = """
8     <!DOCTYPE html>
9     <html lang="en">
10     <head>
11         <meta charset="UTF-8">
12         <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
14     <meta name="viewport" content="width=device-width, initial-  
scale=1.0">  
15     <title>Mon Premier Conteneur</title>  
16     <style>  
17         body {  
18             font-family: Arial, sans-serif;  
19             background-color: #f4f4f4;  
20             text-align: center;  
21             margin: 100px;  
22         }  
23         h1 {  
24             color: #333;  
25         }  
26     </style>  
27 </head>  
28 <body>  
29     <h1>Mon Premier Conteneur</h1>  
30     <p>Test successful!</p>  
31     <p> Done by : Heddadji and Soltani SIQ2 VCL module </p>  
32 </body>  
33 </html>  
34 """  
35     return render_template_string(html_content)  
36  
37 if __name__ == '__main__':  
38     app.run(host='0.0.0.0', port=8081)  
39  
40 .
```

Dockerfile

```
1  
2 FROM python:3.11-slim  
3  
4 WORKDIR /app  
5  
6 COPY . /app  
7  
8 RUN pip install flask  
9  
10 CMD ["flask", "run", "--host=0.0.0.0", "--port=8081"]  
11  
12 .
```



```

heddadjl@heddadjl: ~/app
heddadjl@heddadjl:~/app$ touch app.py
heddadjl@heddadjl:~/app$ touch Dockerfile
heddadjl@heddadjl:~/app$ nano app.py
heddadjl@heddadjl:~/app$ nano Dockerfile
heddadjl@heddadjl:~/app$ docker build -t flaskapp .
[+] Building 15.4s (9/9) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 159B                                              0.0s
=> [internal] load .dockerignore                                                  0.0s
=> => transferring context: 28                                                    0.0s
=> [internal] load metadata for docker.io/library/python:3.11-slim              2.2s
=> CACHED [1/4] FROM docker.io/library/python:3.11-slim@sha256:2a746e2b9         0.0s
=> [internal] load build context                                                  0.0s
=> => transferring context: 1.19kB                                               0.0s
=> [2/4] WORKDIR /app                                                            0.1s
=> [3/4] COPY . /app                                                            0.1s
=> [4/4] RUN pip install flask                                                  12.7s
=> exporting to image                                                            0.3s
=> => exporting layers                                                            0.2s
=> => writing image sha256:12ced2dabf54b883876d9e3c5aae1ce69935e583b7166       0.0s
=> => naming to docker.io/library/flaskapp                                       0.0s
heddadjl@heddadjl:~/app$

```

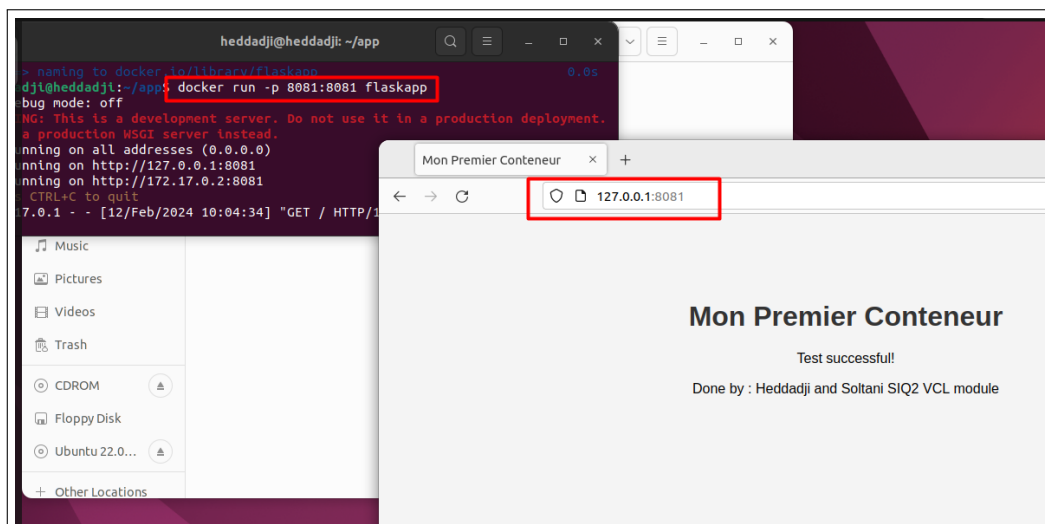
Ubuntu 22.0...
+ Other Locations
"Dockerfile" selected (122 bytes)

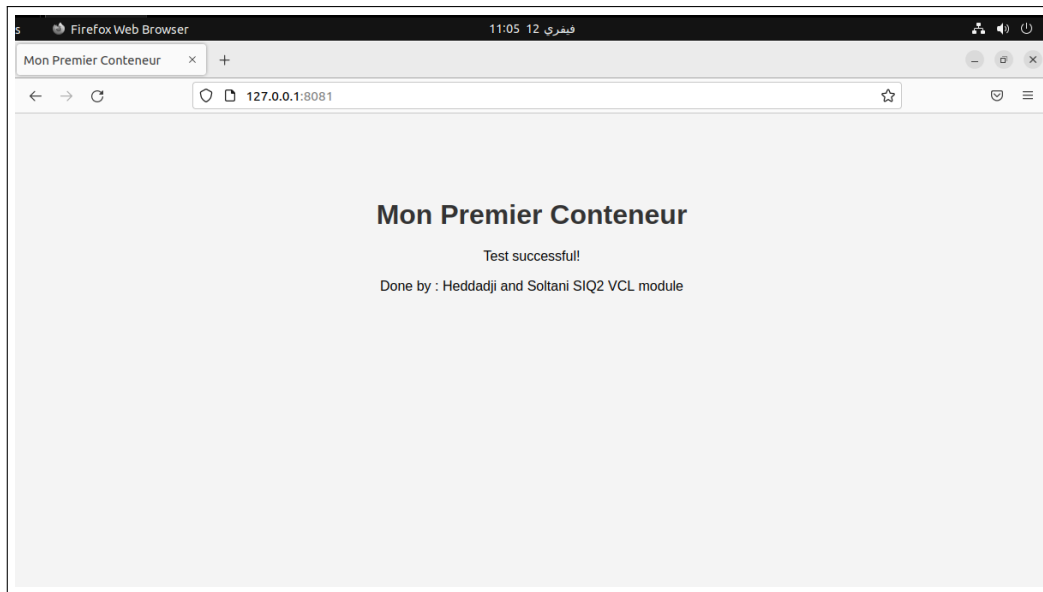
5.1. Building and Running the Docker Image

```

1 cd lab
2 // . : current dir
3 docker build -t flaskapp .
4 docker run -p 8081:8081 flaskapp
5 #Access in http://localhost:8081
6
7 .

```





5.2. Hosting the Docker image to Docker Hub

1. **Create a Docker Hub Account:** <https://hub.docker.com/>.

2. **Login to Docker Hub:**

```
1 docker login
```

Enter your Docker Hub username and password when prompted.

3. **Tag your Docker Image:** with Docker Hub username and the desired repository name.

```
1 docker tag flaskapp your-username/repository-name
```

4. **Push the Docker Image:**

```
1 docker push your-username/repository-name
```

```
heddadjl@heddadjl: ~/app
heddadjl@heddadjl:~/app$ 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://docs.docker.com/get-docker/#create-one.
Username: nourelimanehed
Password:
WARNING! Your password will be stored unencrypted in /home/heddadjl/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
heddadjl@heddadjl:~/app$ docker tag flaskapp nourelimanehed/flaskapp
heddadjl@heddadjl:~/app$ docker push nourelimanehed/flaskapp
Using default tag: latest
The push refers to repository [docker.io/nourelimanehed/flaskapp]
6f0e524afb0e: Pushed
a5bb73df1d33: Pushed
7fab17ff2e04: Pushed
26c46bc35769: Mounted from library/python
70c07101f660: Mounted from library/python
cfb2f66590e2: Mounted from library/python
da5d55102092: Mounted from library/python
fb1bd2fc5282: Mounted from library/python
latest: digest: sha256:332591cfff908891255bd945e3d38b6eb3e5370523cd22bbce6c833c68be9496 size: 1994
heddadjl@heddadjl:~/app$
```

