Assignment2-Docker

Task1

Demonstrate a minimum of 15 basic docker commands with an explanation and screenshots.

Solution:

1. **docker pull <<image_name>>** —> Pulls the image from the repository

```
admin@docker-host ~ ➜ docker pull docker/getting-started
Using default tag: latest
latest: Pulling from docker/getting-started
df9b9388f04a: Pull complete
5867cba5fcbd: Pull complete
4b639e65cb3b: Pull complete
061ed9e2b976: Pull complete
bc19f3e8eeb1: Pull complete
4071be97c256: Pull complete
79b586f1a54b: Pull complete
0c9732f525d6: Pull complete
Digest: sha256:b558be874169471bd4e65bd6eac8c303b271a7ee8553ba47481b73b2bf597aae
Status: Downloaded newer image for docker/getting-started:latest
docker.io/docker/getting-started:latest
```

2. **docker —version** —> Displays the version of the docker

```
admin@docker-host ~ ➜ docker --version
Docker version 20.10.10, build b485636
```

3. **docker run** —> Runs the image as a container

Example:

docker run -d -p 80:80 docker/getting-started

-d —> detach mode

-p 80:80 —> maps the host port 80 to container port 80

```
admin@docker-host ~ ➜ docker run -d -p 80:80 docker/getting-started
929f40533b6a4ca881ea01c576c8b5421a42630cddc9e63f37ac1d48bdc536a2
```

4. **docker images** —> Displays the current pulled images

```
admin@docker-host ~ ➜ docker images
REPOSITORY                TAG       IMAGE ID       CREATED         SIZE
docker/getting-started    latest    cb90f98fd791   5 months ago    28.8MB
```

5. **docker ps** —> To list all the running containers.

```
admin@docker-host ~ ➜ docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS          PORTS                  NAMES
929f40533b6a   docker/getting-started   "/docker-entrypoint.…"   14 minutes ago   Up 14 minutes   0.0.0.0:80->80/tcp   distracted_blackburn
```

6. **docker stop <<container_id>>** —> Stops the running container

```
admin@docker-host ~ ➜ docker stop 929f40533b6a
929f40533b6a
```

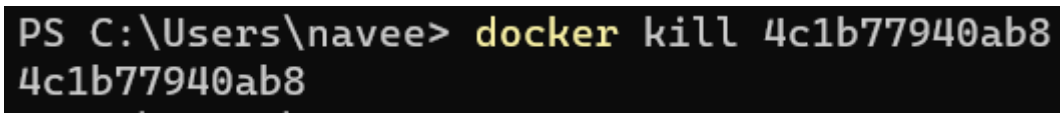
7. **docker start <<container_id>>** —> Starts the container

```
admin@docker-host ~ ➜ docker start 929f40533b6a
929f40533b6a
```

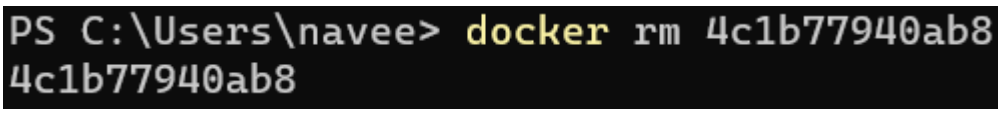
8. **docker logs <<container_id>>** —> Displays logs of the container

```
PS C:\Users\navee> docker logs 0017dbaad9a0

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/
```

8. **docker kill <<container_id>>** —> Force shutdown the container



10. **docker rm <<container_id>>** —> Removes the container from the machine



11. **docker rmi <<image_name>>** —> Removes the image from the machine

```
PS C:\Users\navee> docker rmi docker/getting-started
Untagged: docker/getting-started:latest
Untagged: docker/getting-started@sha256:b558be874169471bd4e65bd6eac8c303b271a7ee8553ba47481b73b2bf597aae
Deleted: sha256:cb90f98fd791dd49f09903cef3eb2245646b4d76b093825ea78e0f7bb8fb3403
Deleted: sha256:b6b308c7ce72e0286f9455b9f76ae6cafe55fcc6b068950414165f43bda11fd7
Deleted: sha256:711ca3e1c68e1406fd5b96a71fcf29e4838887b827bd4ee48dfc6e6a62d8fabf
Deleted: sha256:1380ce106a10fac3c312f83ddf8406d187d5c0dd567d9a2454abe6ba563114cd
Deleted: sha256:36e9639dd7f8b2549aba50c0a7d2402510ddb99d3e789515ab6646f21ef392ec
Deleted: sha256:b35646458162a8f3289c0605c02ad46c2a05ae5c977a46e11d56962b373e1e98
Deleted: sha256:e61e5c961a35926efc4df0bcd33aa988c860ba8440ae2bb713084b14b89c9806
Deleted: sha256:f60e2e50f4b58e60ef21034b9d2df92705fa8bb3870b2ca81089de8af45a2e90
Deleted: sha256:4fc242d58285699eca05db3cc7c7122a2b8e014d9481f323bd9277baacfa0628
```

12. **docker rename <<old_name>> <<new_name>>** —> Rename the container

```
PS C:\Users\navee> docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED          STATUS          PORTS                  NAMES
cc7426fc1e94   docker/getting-started   "/docker-entrypoint.…"   24 seconds ago   Up 12 seconds   0.0.0.0:80->80/tcp     busy_jemison
PS C:\Users\navee> docker rename busy_jemison new_nametray
PS C:\Users\navee> docker ps
CONTAINER ID   IMAGE                    COMMAND                  CREATED              STATUS              PORTS                  NAMES
cc7426fc1e94   docker/getting-started   "/docker-entrypoint.…"   About a minute ago   Up About a minute   0.0.0.0:80->80/tcp     new_nametray
```

13. **docker port <<container_id>>** —> Displays the port mapping of the container

```
PS C:\Users\navee> docker port cc7426fc1e94
80/tcp -> 0.0.0.0:80
```

14. **docker stats** —> Displays the usage statistics of docker

```
PS C:\Users\navee> docker stats
CONTAINER ID   NAME           CPU %     MEM USAGE / LIMIT     MEM %     NET I/O          BLOCK I/O   PIDS
cc7426fc1e94   new_nametray   0.00%     9.02MiB / 3.513GiB    0.25%     1.16kB / 0B      0B / 0B     13
CONTAINER ID   NAME           CPU %     MEM USAGE / LIMIT     MEM %     NET I/O          BLOCK I/O   PIDS
cc7426fc1e94   new_nametray   0.00%     9.02MiB / 3.513GiB    0.25%     1.16kB / 0B      0B / 0B     13
CONTAINER ID   NAME           CPU %     MEM USAGE / LIMIT     MEM %     NET I/O          BLOCK I/O   PIDS
cc7426fc1e94   new_nametray   0.00%     9.02MiB / 3.513GiB    0.25%     1.16kB / 0B      0B / 0B     13
CONTAINER ID   NAME           CPU %     MEM USAGE / LIMIT     MEM %     NET I/O          BLOCK I/O   PIDS
cc7426fc1e94   new_nametray   0.00%     9.02MiB / 3.513GiB    0.25%     1.16kB / 0B      0B / 0B     13
CONTAINER ID   NAME           CPU %     MEM USAGE / LIMIT     MEM %     NET I/O          BLOCK I/O   PIDS
cc7426fc1e94   new_nametray   0.00%     9.02MiB / 3.513GiB    0.25%     1.16kB / 0B      0B / 0B     13
CONTAINER ID   NAME           CPU %     MEM USAGE / LIMIT     MEM %     NET I/O          BLOCK I/O   PIDS
cc7426fc1e94   new_nametray   0.00%     9.02MiB / 3.513GiB    0.25%     1.16kB / 0B      0B / 0B     13
CONTAINER ID   NAME           CPU %     MEM USAGE / LIMIT     MEM %     NET I/O          BLOCK I/O   PIDS
cc7426fc1e94   new_nametray   0.00%     9.02MiB / 3.513GiB    0.25%     1.16kB / 0B      0B / 0B     13
```

15. **docker inspect <<image_name>>** —> Get the low-level information on the image in JSON format

```
PS C:\Users\navee> docker inspect docker/getting-started
[
    {
        "Id": "sha256:cb90f98fd791dd49f09903cef3eb2245646b4d76b093825ea78e0f7bb8fb3403",
        "RepoTags": [
            "docker/getting-started:latest"
        ],
        "RepoDigests": [
            "docker/getting-started@sha256:b558be874169471bd4e65bd6eac8c303b271a7ee8553ba47481b73b2bf597aae"
        ],
        "Parent": "",
        "Comment": "buildkit.dockerfile.v0",
        "Created": "2022-04-11T15:23:08.189460029Z",
        "Container": "",
        "ContainerConfig": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": null,
            "Cmd": null,
            "Image": "",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
```

11. **docker build -t <<image_name>> .** —> To build images

```
navee@LAPTOP-VKVN9NN8 MINGW64 /c/Naveen/DataScienceProjects/dockerAssignment (main)
$ docker build -t "naveenkaranth/hello-fastapi" .
[+] Building 5.3s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 31B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/python:3.8
 => [auth] library/python:pull token for registry-1.docker.io
 => [internal] load build context
 => => transferring context: 450.55kB
 => [1/4] FROM docker.io/library/python:3.8@sha256:041a09359fda61a2b58cec5201560646b66e05137ce2af1cab9ceec920279b53
 => CACHED [2/4] COPY . /app
 => CACHED [3/4] WORKDIR /app
 => CACHED [4/4] RUN pip install -r requirements.txt
 => exporting to image
 => => exporting layers
 => => writing image sha256:82c3176581e7b3ab91a35860c764eb74d252d369f936be9a73918303e5f9a1ca
 => => naming to docker.io/naveenkaranth/hello-fastapi
```

Task2

Hello, World Docker Image Run Hello World Docker Image Locally.

Solution:

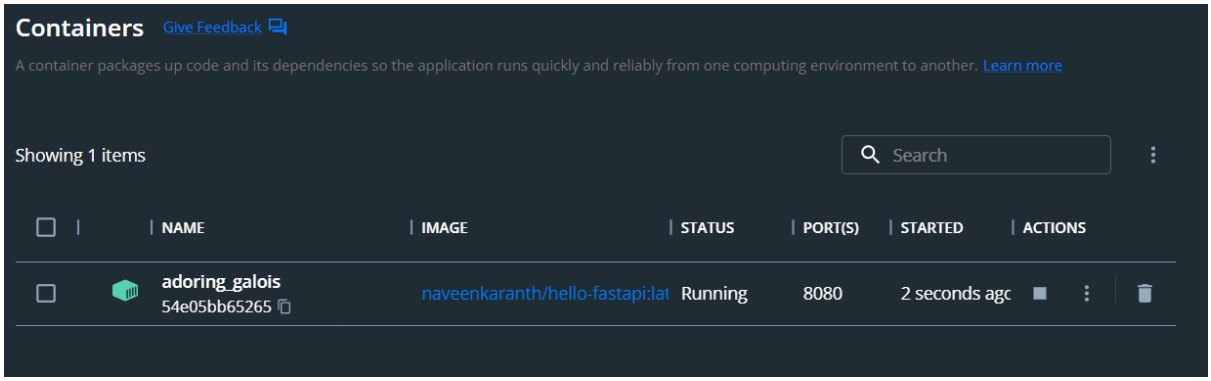
```
PS C:\Users\navee> docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:62af9efd515a25f84961b70f973a798d2eca956b1b2b026d0a4a63a3b0b6a3f2
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
PS C:\Users\navee> docker images
REPOSITORY    TAG       IMAGE ID       CREATED         SIZE
hello-world   latest    feb5d9fea6a5   12 months ago   13.3kB
PS C:\Users\navee> docker 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/
```

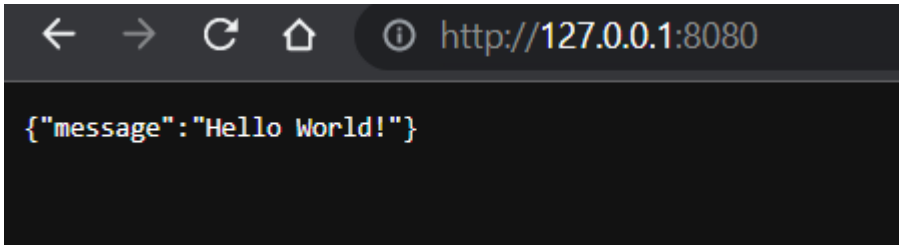
Task3

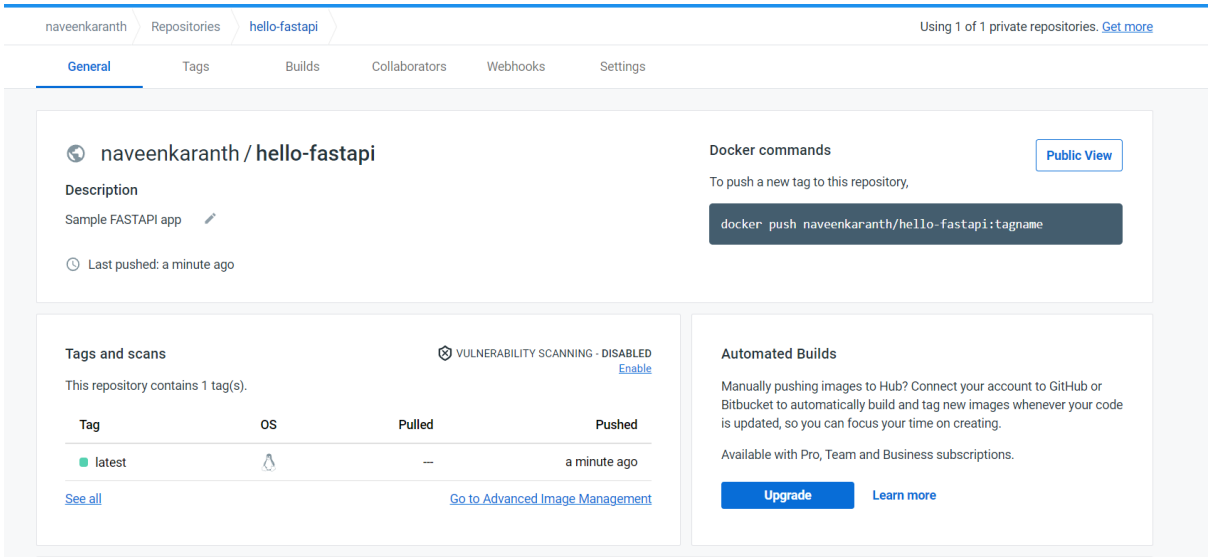
Create a hello world FastAPI application. Create a Dockerfile for your FastAPI hello world application. Build Docker image using Docker file. Run the docker image build in the previous step. Push your Docker image to Docker Hub.

Solution:

https://github.com/nav52/dockerAssignment







Task4

Automate the Assignment below task using GitHub action.

1. Build Docker Image
2. Push Docker Image to Docker hub.

Solution:

https://github.com/nav52/dockerAssignment

- Create Docker hub token and add them to GitHub repository secrets
- Create the YAML file in the .github/workflows/ to login to Docker hub, build the image on push to the main branch and push the ‘image’ to the docker hub repository.