Docker task 3 (Dockerfile customization)

1. cd /home/ubuntu & mkdir docker

```
( We change to ubuntu directory and create docker directory inside ubuntu
directory )
```

2. cd /home/ubuntu/docker & touch Dockerfile

```
( We change to docker directory and create Dockerfile inside docker
directory )
```

3. nano Dockerfile

```
( Now we gonna customize the Dockerfile which will be used as the image in
further processes using nano text editor )
```

4. cat Dockerfile

   output:

```
# Use Ubuntu as the base image
FROM ubuntu:latest

# Update the package list and install curl in a single RUN command to
reduce layers
RUN apt-get update && apt-get install -y curl && apt-get clean

# Set the working directory inside the container
WORKDIR /app

# Copy the entire current directory's contents to the container's /app
directory
COPY . /app

# Set an environment variable with a different name
ENV CONTAINER_HOSTNAME="MyDockerContainer"
```

```
# Set a long-running command
CMD ["tail", "-f", "/dev/null"]
```

This is our customized image's coding, Now lets see breifly about our customized image...

- "FROM ubuntu:latest" is used to pull latest ubuntu image from Dockerhub to this Dockerfile
- "RUN apt-get update && apt-get install -y curl && apt-get clean" is used to install new package updates , curl and "apt-get clean" is used to remove unwanted cache packages which are installed inside image after these installations to reduce image size
- "WORKDIR /app" is used to set "/app" as the working directory inside the container
- "COPY . /app" is used to copy the current directory contents inside the "app" directory of container
- "ENV CONTAINER_HOSTNAME="MyDockerContainer" is used to set environment variable as container_hostname with its value as mydockercontainer, the variable can be used inside the container at it's needed times.
- "CMD ["tail", "-f", "/dev/null"] " is a long-running command, It is used because after the container creation this command makes the container to remain in the running state by not exiting

5. sudo docker build -t custom_image:v1 .

   output :

```
[+] Building 19.1s (9/9) FINISHED
docker:default
=> [internal] load build definition from Dockerfile
0.0s
=> => transferring dockerfile: 552B
0.0s
=> [internal] load metadata for docker.io/library/ubuntu:latest
2.2s
=> [internal] load .dockerignore
0.0s
=> => transferring context: 2B
0.0s
=> [1/4] FROM
docker.io/library/ubuntu:latest@sha256:99c35190e22d294cdace2783ac55effc69d32896daaa
265f0bbedbcde4fbe3e5                                          2.4s
=> => resolve
docker.io/library/ubuntu:latest@sha256:99c35190e22d294cdace2783ac55effc69d32896daaa
265f0bbedbcde4fbe3e5                                          0.0s
=> =>      sha256:59ab366372d56772eb54e426183435e6b0642152cb449ec7ab52473af8ca6e3f
2.30kB / 2.30kB                                                        0.0s
=> => sha256:ff65ddf9395be21bfe1f320b7705e539ee44c1053034f801b1a3cbbf2d0f4056
```

```
29.75MB / 29.75MB                                                    0.5s
=> => sha256:99c35190e22d294cdace2783ac55effc69d32896daaa265f0bbedbcde4fbe3e5
6.69kB / 6.69kB                                                      0.0s
=> => sha256:5d070ad5f7fe63623cbb99b4fc0fd997f5591303d4b03ccce50f403957d0ddc4 424B
/ 424B                                                               0.0s
=> => extracting
sha256:ff65ddf9395be21bfe1f320b7705e539ee44c1053034f801b1a3cbbf2d0f4056
1.7s
=> [internal] load build context
0.0s
=> => transferring context: 603B
0.0s
=> [2/4] RUN apt-get update && apt-get install -y curl && apt-get clean
13.8s
=> [3/4] WORKDIR /app
0.1s
=> [4/4] COPY . /app
0.0s
=> exporting to image
0.4s
=> => exporting layers
0.4s
=> => writing image
sha256:c9ec4e5a2600a8cb803169fd05d0bde5523cd73d33e0795f746889913462dd29
0.0s
=> => naming to docker.io/library/custom_image:v1
0.0s



( This command is used to create an docker image in name "custom_image:v1"
by using the Dockerfile in the present directory)
```

6. docker login -u kapilanramesh

```
( This command is used to login inside the dockerhub with docker username
and password )
```

7. sudo docker tag custom_image:v1 kapilanramesh/custom_image:v1

```
( Use this command to tag your image to Dockerhub if your image isn't
tagged )
```

**--> Do this step if necessary

8. sudo docker push kapilanramesh/myimage:1

```
output :

The push refers to repository [docker.io/kapilanramesh/myimage]
62c59050b6ad: Pushed
eaea00d19660: Pushed
0345c9457665: Pushed
a46a5fb872b5: Mounted from library/ubuntu
1: digest:
sha256:b6730041aa9617577a42339529f352cc1a1b4ff02fe3b44ba3712401da39cc3b
size:
1154


( This command is used to push our customized image to the Dockerhub, And
this step has to be done after tagging our image to the dockerhub )
```

9. sudo docker run -d -it --name myimage_container1 custom_image:v1

```
output :

3b15405569139cf29f1a68e0bf5a5e51a9a30c10fe6b3a25b1ff111d579d3c2e


( This command is used to create an container with our customized image)
```