

## Experiment-6

**AIM:-Create a docker image for any application using docker file and push it to Docker Hub.**

### Step 1:-Creating Instance Ubuntu in aws

1. Login AWS(Amazon Web Services) Account
2. Lunch Instance name Docker

### Step 2:-Create Docker Hub Account and create repository in Docker Hub

### Step 3:-Install Docker and Check Status and Start Docker

1. `sudo apt update -y`
2. `sudo apt install docker.io -y`
3. `sudo systemctl status docker`(come outside use command `ctl+z`)

```
ubuntu@ip-172-31-90-47:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-02-28 03:50:40 UTC; 2min 5s ago
     TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
      Main PID: 2110 (dockerd)
         Tasks: 8
        Memory: 35.8M (peak: 38.1M)
           CPU: 253ms
        CGroup: /system.slice/docker.service
                └─2110 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Feb 28 03:50:38 ip-172-31-90-47 systemd[1]: Starting docker.service - Docker Application Container Engine...
Feb 28 03:50:38 ip-172-31-90-47 dockerd[2110]: time="2025-02-28T03:50:38.974271527Z" level=info msg="Starting up"
Feb 28 03:50:38 ip-172-31-90-47 dockerd[2110]: time="2025-02-28T03:50:38.975783980Z" level=info msg="detected 127.0.0.53 name
Feb 28 03:50:39 ip-172-31-90-47 dockerd[2110]: time="2025-02-28T03:50:39.094061737Z" level=info msg="Loading containers: star
Feb 28 03:50:39 ip-172-31-90-47 dockerd[2110]: time="2025-02-28T03:50:39.560107166Z" level=info msg="Loading containers: done
Feb 28 03:50:40 ip-172-31-90-47 dockerd[2110]: time="2025-02-28T03:50:40.287913914Z" level=info msg="Docker daemon" commit="2
Feb 28 03:50:40 ip-172-31-90-47 dockerd[2110]: time="2025-02-28T03:50:40.288020817Z" level=info msg="Daemon has completed ini
Feb 28 03:50:40 ip-172-31-90-47 dockerd[2110]: time="2025-02-28T03:50:40.338547546Z" level=info msg="API listen on /run/docke
Feb 28 03:50:40 ip-172-31-90-47 systemd[1]: Started docker.service - Docker Application Container Engine.
lines 1-21/21 (END)
```

Above status command is docker running means no problem if not run use command below to run

4. `sudo systemctl start docker`

### Step 4:- Grant Access

Why we give grant access means

A easy way to verify your Docker installation is by running the below command

`docker run hello-world`

If the output says:

```
ubuntu@ip-172-31-90-47:~$ docker run hello-world
docker: permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Head "http://%2Fvar%2Frun%2Fdocker.sock/_ping": dial unix /var/run/docker.sock: connect: permission denied.
See 'docker run --help'.
ubuntu@ip-172-31-90-47:~$
```

This can mean two things,

1. Docker daemon is not running.(start docker using “sudo systemctl start docker”)
2. Your user does not have access to run docker commands.

### Grant Access to your user to run docker commands

1. `sudo usermod -aG docker ubuntu`

In the above command ubuntu is the name of the user, you can change the username appropriately.

**NOTE:** : You need to logout and login back for the changes to be reflected.

2. Logout purpose use commands `exit` or `logout`

Again run command “`docker run hello-world`”

```
Last login: Mon Mar 3 07:15:35 2025 from 103.172.179.18
ubuntu@ip-172-31-90-15:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
e6590344b1a5: Pull complete
Digest: sha256:bfb0cc14f13f9ed1ae86abc2b9f11181dc50d779807ed3a3c5e55a6936dbdd5
Status: Downloaded newer image for hello-world:latest

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/
```

### Step 5:-Creating application and Docker file

1. `mkdir docker1`
2. `cd docker1`
3. `vim app.py`

```
print("hello world")
```

4. `cat app.py`
5. `vim Dockerfile` (below pic commands write lab record) typing command `vim` before click “i” for insert data after completion Docker file commands save before click `esc` use `:wq`

```
ubuntu@ip-172-31-90-15:~$ docker7
FROM ubuntu:latest

# Set the working directory in the image
WORKDIR /app

# Copy the files from the host file system to the image file system
COPY . /app

# Install the necessary packages
RUN apt-get update && apt-get install -y python3 python3-pip

# Set environment variables
ENV NAME World

# Run a command to start the application
CMD ["python3", "app.py"]
```

**Below image for understanding purpose**

```
ubuntu@ip-172-31-90-47:~/docker1$ vim app.py
ubuntu@ip-172-31-90-47:~/docker1$ cat app.py
print("hello world")
ubuntu@ip-172-31-90-47:~/docker1$ vim Dockerfile
ubuntu@ip-172-31-90-47:~/docker1$ cat Dockerfile
FROM ubuntu:latest

# Set the working directory in the image
WORKDIR /app

# Copy the files from the host file system to the image file system
COPY . /app

# Install the necessary packages
RUN apt-get update && apt-get install -y python3 python3-pip

# Set environment variables
ENV NAME World

# Run a command to start the application
CMD ["python3", "app.py"]

ubuntu@ip-172-31-90-47:~/docker1$
```

### Step 6:- Build and Check Docker image

Syntax:- `docker build -t dockerhub_username/repositoryname:tag .`

1. `docker build -t deekshith0607/test02:image.`
2. `docker images`

## Step 7:- Run your First Docker Container

1. `docker run -it deekshith0607/test02:image`
2. Output

# Hello World

```
ubuntu@ip-172-31-90-15:~/docker07$ docker run -it deekshith0607/test02:image
helloworld
ubuntu@ip-172-31-90-15:~/docker07$
```

## Step 8:-Docker Login

1. docker login

```
Username: deekshith0607
Password:
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

## Step 9:- Push the Image to DockerHub and share it with the world

1. docker push deekshith0607/test02:image

```
ubuntu@ip-172-31-90-15:~/docker07$ docker push deekshith0607/test02:image
The push refers to repository [docker.io/deekshith0607/test02]
ed3940815687: Pushed
705d2e1b5105: Pushed
84a24c5bbf62: Pushed
4b7c01ed0534: Mounted from library/ubuntu
image: digest: sha256:e4f9d06c41f2f3f8fa4166e4318186c8a926b1fd1b36b8b061ccd974f2d0d09c size: 1155
```

## Output:-

1. docker images (Command to Check output)

```
ubuntu@ip-172-31-90-15:~/docker07$ docker images
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
deekshith0607/test02 image       772d53dbb82e 12 minutes ago 574MB
ubuntu              latest      a04dc4851cbc 5 weeks ago   78.1MB
hello-world         latest      74cc54e27dc4 5 weeks ago   10.1kB
ubuntu@ip-172-31-90-15:~/docker07$
```



