Installation of docker:

```
vaagdevi@ubuntu:~$ sudo apt install docker.io
[sudo] password for vaagdevi:
Reading package lists... Done
Building dependency tree
Reading state information... Done
docker.io is already the newest version (20.10.21-0ubuntu1~20.04.2).
0 upgraded, 0 newly installed, 0 to remove and 58 not upgraded.
```

Docker update:

```
vaagdevi@ubuntu:~$ sudo apt update
Hit:1 http://us.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:4 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Fetched 108 kB in 2s (56.0 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
58 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

Docker is set into path or not:

```
vaagdevi@ubuntu:~$ docker

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:
      --config string      Location of client config files (default
                           "/home/vaagdevi/.docker")
  -c, --context string     Name of the context to use to connect to the
                           daemon (overrides DOCKER_HOST env var and
                           default context set with "docker context use")
  -D, --debug              Enable debug mode
  -H, --host list          Daemon socket(s) to connect to
  -l, --log-level string   Set the logging level
                           ("debug"|"info"|"warn"|"error"|"fatal")
                           (default "info")
      --tls                Use TLS; implied by --tlsverify
      --tlscacert string   Trust certs signed only by this CA (default
                           "/home/vaagdevi/.docker/ca.pem")
      --tlscert string     Path to TLS certificate file (default
                           "/home/vaagdevi/.docker/cert.pem")
      --tlskey string      Path to TLS key file (default
                           "/home/vaagdevi/.docker/key.pem")
      --tlsverify          Use TLS and verify the remote
  -v, --version            Print version information and quit

Management Commands:
  builder     Manage builds
  config      Manage Docker configs
  container   Manage containers
  context     Manage contexts
  image       Manage images
  manifest    Manage Docker image manifests and manifest lists
  network     Manage networks
  node        Manage Swarm nodes
  plugin      Manage plugins
  secret      Manage Docker secrets
  service     Manage services
```

```
  stack       Manage Docker stacks
  swarm       Manage Swarm
  system      Manage Docker
  trust       Manage trust on Docker images
  volume      Manage volumes

Commands:
  attach      Attach local standard input, output, and error streams to a running container
  build       Build an image from a Dockerfile
  commit      Create a new image from a container's changes
  cp          Copy files/folders between a container and the local filesystem
  create      Create a new container
  diff        Inspect changes to files or directories on a container's filesystem
  events      Get real time events from the server
  exec        Run a command in a running container
  export      Export a container's filesystem as a tar archive
  history     Show the history of an image
  images      List images
  import      Import the contents from a tarball to create a filesystem image
  info        Display system-wide information
  inspect     Return low-level information on Docker objects
  kill        Kill one or more running containers
  load        Load an image from a tar archive or STDIN
  login       Log in to a Docker registry
  logout      Log out from a Docker registry
  logs        Fetch the logs of a container
  pause       Pause all processes within one or more containers
  port        List port mappings or a specific mapping for the container
  ps          List containers
  pull        Pull an image or a repository from a registry
  push        Push an image or a repository to a registry
  rename      Rename a container
  restart     Restart one or more containers
  rm          Remove one or more containers
  rmi         Remove one or more images
  run         Run a command in a new container
  save        Save one or more images to a tar archive (streamed to STDOUT by default)
  search      Search the Docker Hub for images
  start       Start one or more stopped containers
  stats       Display a live stream of container(s) resource usage statistics
```

```
  stop        Stop one or more running containers
  tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
  top         Display the running processes of a container
  unpause     Unpause all processes within one or more containers
  update      Update configuration of one or more containers
  version     Show the Docker version information
  wait        Block until one or more containers stop, then print their exit codes

Run 'docker COMMAND --help' for more information on a command.

To get more help with docker, check out our guides at https://docs.docker.com/go/guides/
```

Docker version:

```
vaagdevi@ubuntu:~$ docker --version
Docker version 20.10.21, build 20.10.21-0ubuntu1~20.04.2
```

To know the status or running of docker:

```
vaagdevi@ubuntu:~$ sudo systemctl status docker
[sudo] password for vaagdevi:
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset>
     Active: active (running) since Mon 2023-05-29 01:35:25 PDT; 29min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 1175 (dockerd)
      Tasks: 8
     Memory: 47.5M
     CGroup: /system.slice/docker.service
             └─1175 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/cont>

May 29 01:35:24 ubuntu dockerd[1175]: time="2023-05-29T01:35:24.558029153-07:00>
May 29 01:35:24 ubuntu dockerd[1175]: time="2023-05-29T01:35:24.558051523-07:00>
May 29 01:35:24 ubuntu dockerd[1175]: time="2023-05-29T01:35:24.558055405-07:00>
May 29 01:35:24 ubuntu dockerd[1175]: time="2023-05-29T01:35:24.558640373-07:00>
May 29 01:35:25 ubuntu dockerd[1175]: time="2023-05-29T01:35:25.171139158-07:00>
May 29 01:35:25 ubuntu dockerd[1175]: time="2023-05-29T01:35:25.272988773-07:00>
May 29 01:35:25 ubuntu dockerd[1175]: time="2023-05-29T01:35:25.372864616-07:00>
May 29 01:35:25 ubuntu dockerd[1175]: time="2023-05-29T01:35:25.377226680-07:00>
May 29 01:35:25 ubuntu systemd[1]: Started Docker Application Container Engine.
May 29 01:35:25 ubuntu dockerd[1175]: time="2023-05-29T01:35:25.436774703-07:00>

[1]+  Stopped                 sudo systemctl status docker
```

Pulling image from Ubuntu:

```
vaagdevi@ubuntu:~$ sudo docker pull ubuntu
Using default tag: latest
12latest: Pulling from library/ubuntu
Digest: sha256:dfd64a3b4296d8c9b62aa3309984f8620b98d87e47492599ee20739e8eb54fbf
Status: Image is up to date for ubuntu:latest
docker.io/library/ubuntu:latest
```

To know docker images:

```
vaagdevi@ubuntu:~$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED             SIZE
mycustom      latest    920eb85ea2bc   About an hour ago   77.8MB
hello-world   latest    9c7a54a9a43c   3 weeks ago         13.3kB
ubuntu        latest    3b418d7b466a   4 weeks ago         77.8MB
```

To run hello-world from docker if it is not present in local repository it will take from docker hub:

```
vaagdevi@ubuntu:~$ sudo 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/
```

To execute the $sudo docker –t filename:

```
vaagdevi@ubuntu:~$ dir
Desktop  Documents  Downloads  Music  Pictures  Public  snap  Templates  Videos
vaagdevi@ubuntu:~$ cd Downloads
vaagdevi@ubuntu:~/Downloads$ mkdir demo
mkdir: cannot create directory ‘demo’: File exists
vaagdevi@ubuntu:~/Downloads$ mkdir file
vaagdevi@ubuntu:~/Downloads$ cd file
vaagdevi@ubuntu:~/Downloads/file$ sudo gedit Dockerfile

(gedit:4150): Tepl-WARNING **: 02:21:56.897: GVfs metadata is not supported. Fallback to TeplMetadataManager. Either GVfs is n
ot correctly installed or GVfs metadata are not supported on this platform. In the latter case, you should configure Tepl with
 --disable-gvfs-metadata.
vaagdevi@ubuntu:~/Downloads/file$ sudo docker build -t "harshith" .
Sending build context to Docker daemon  2.048kB
Step 1/2 : FROM Ubuntu
invalid reference format: repository name must be lowercase
vaagdevi@ubuntu:~/Downloads/file$ sudo gedit Dockerfile

(gedit:4241): Tepl-WARNING **: 02:26:29.905: GVfs metadata is not supported. Fallback to TeplMetadataManager. Either GVfs is n
ot correctly installed or GVfs metadata are not supported on this platform. In the latter case, you should configure Tepl with
 --disable-gvfs-metadata.
```

To save the file in docker:

```
vaagdevi@ubuntu:~/Downloads/file$ sudo docker build -t "harshith" .
Sending build context to Docker daemon  2.048kB
Step 1/2 : FROM ubuntu
 ---> 3b418d7b466a
Step 2/2 : RUN echo hi this is harshith from india
 ---> Running in 161e41b15010
hi this is harshith from india
Removing intermediate container 161e41b15010
 ---> 13642dcda80a
Successfully built 13642dcda80a
Successfully tagged harshith:latest
```

To know docker images:

```
vaagdevi@ubuntu:~/Downloads/file$ sudo docker images
REPOSITORY    TAG       IMAGE ID       CREATED             SIZE
harshith      latest    13642dcda80a   8 minutes ago       77.8MB
mycustom      latest    920eb85ea2bc   About an hour ago   77.8MB
hello-world   latest    9c7a54a9a43c   3 weeks ago         13.3kB
ubuntu        latest    3b418d7b466a   4 weeks ago         77.8MB
```

Docker file container id:

```
vaagdevi@ubuntu:~/Downloads/file$ sudo docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED          STATUS                          PORTS     NAMES
6712a88186b6   hello-world   "/hello"   24 minutes ago   Exited (0) 24 minutes ago                 peaceful_volhard
4b8b34222783   hello-world   "/hello"   3 hours ago      Exited (0) About an hour ago              elated_hamilton
3edd15becc0d   hello-world   "/hello"   3 hours ago      Exited (0) 3 hours ago                    wizardly_villani
```

Docker login:

```
vaagdevi@ubuntu:~/Downloads/file$ sudo docker login
Authenticating with existing credentials...
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Container Id:

```
vaagdevi@ubuntu:~/Downloads/file$ sudo docker container logs 6712a88186b6
[sudo] password for vaagdevi:

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/
```