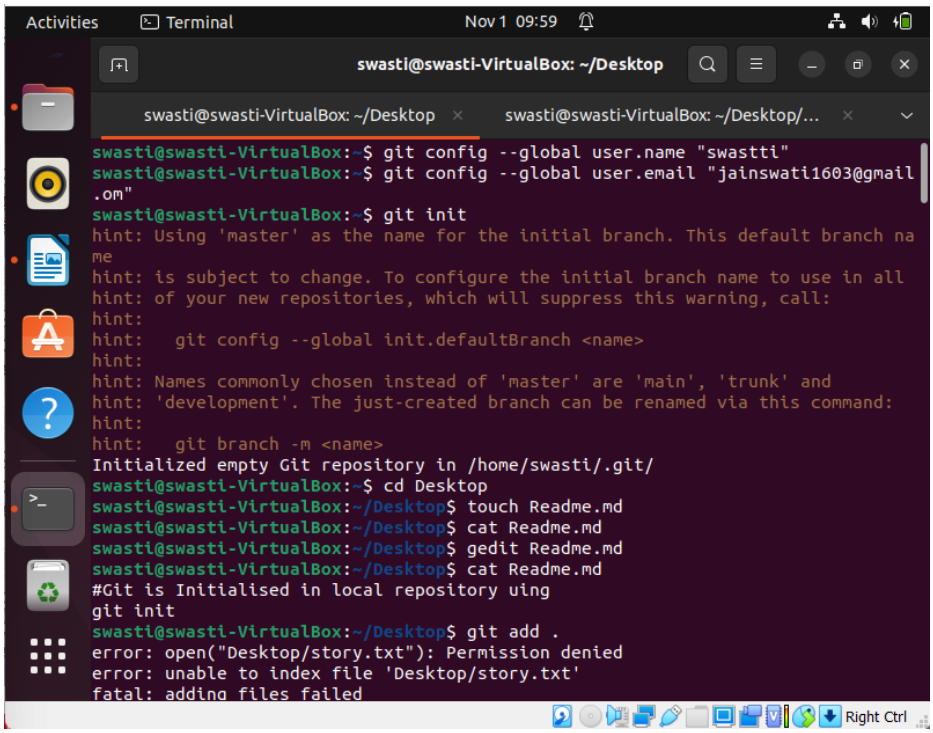
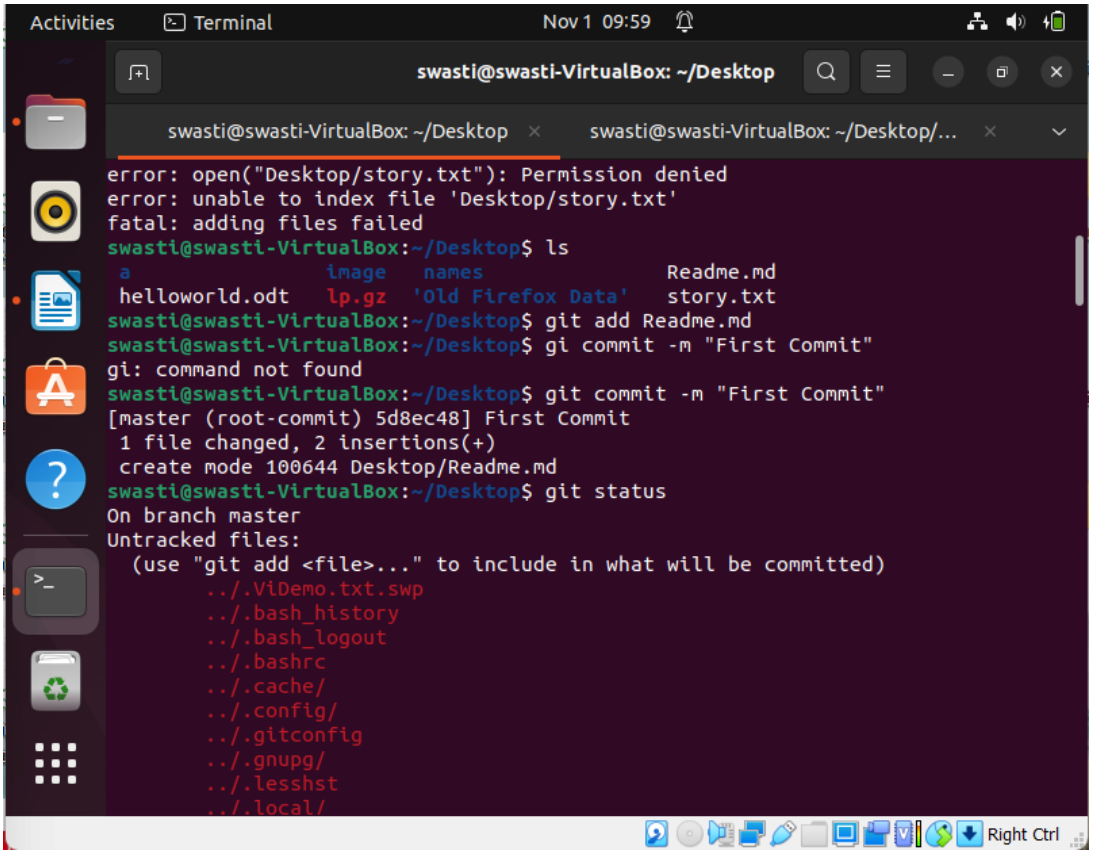
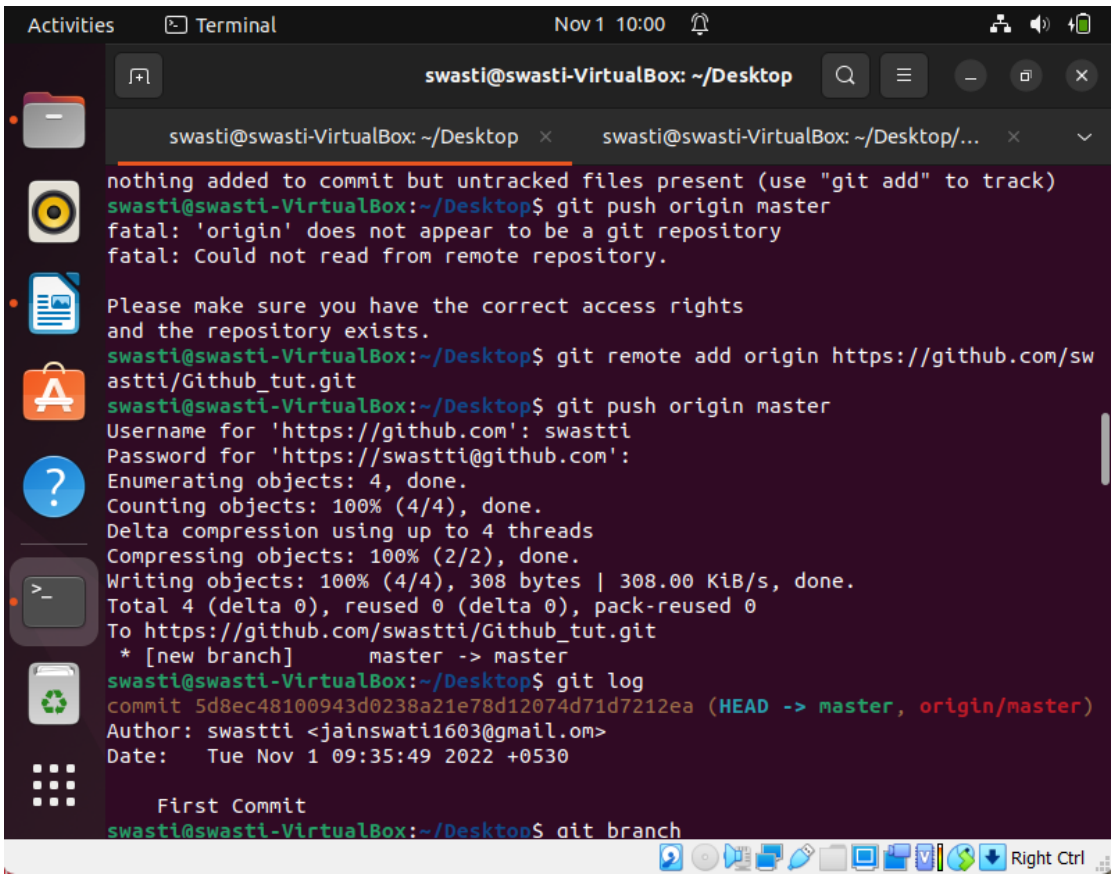
WHAT IS GIT?

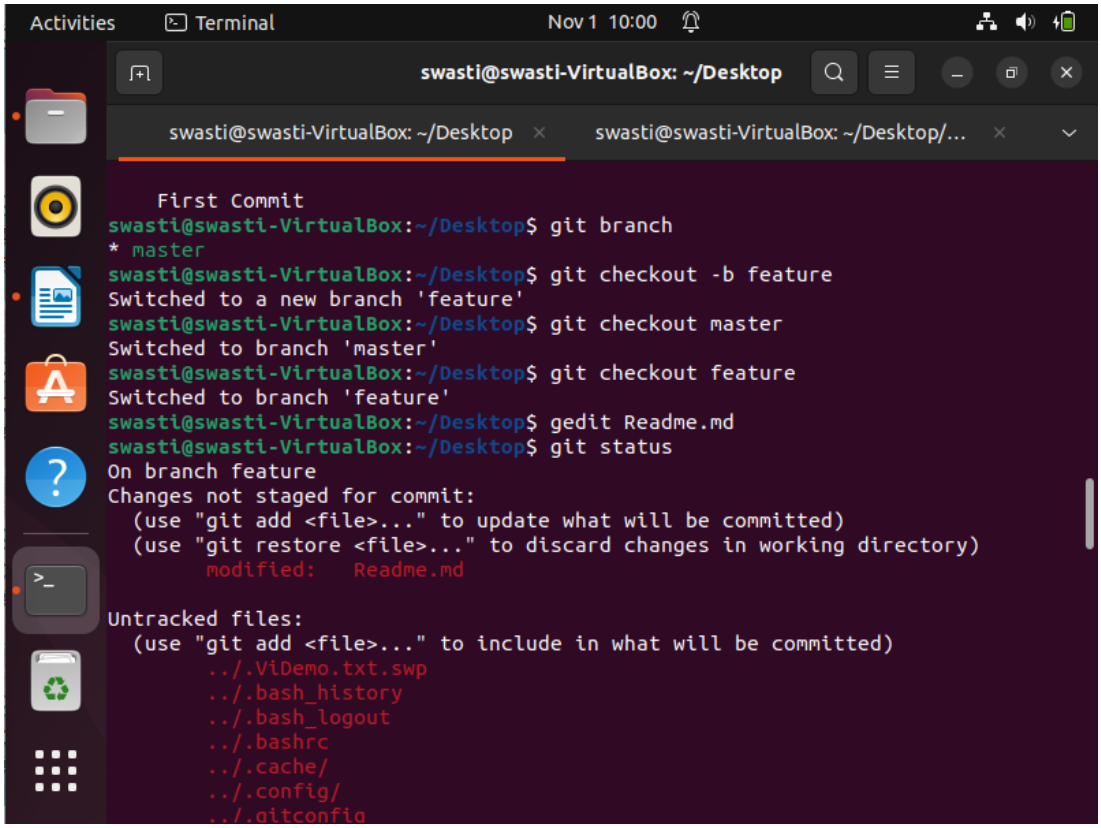
Version Control System (VCS) for tracking changes in computer files

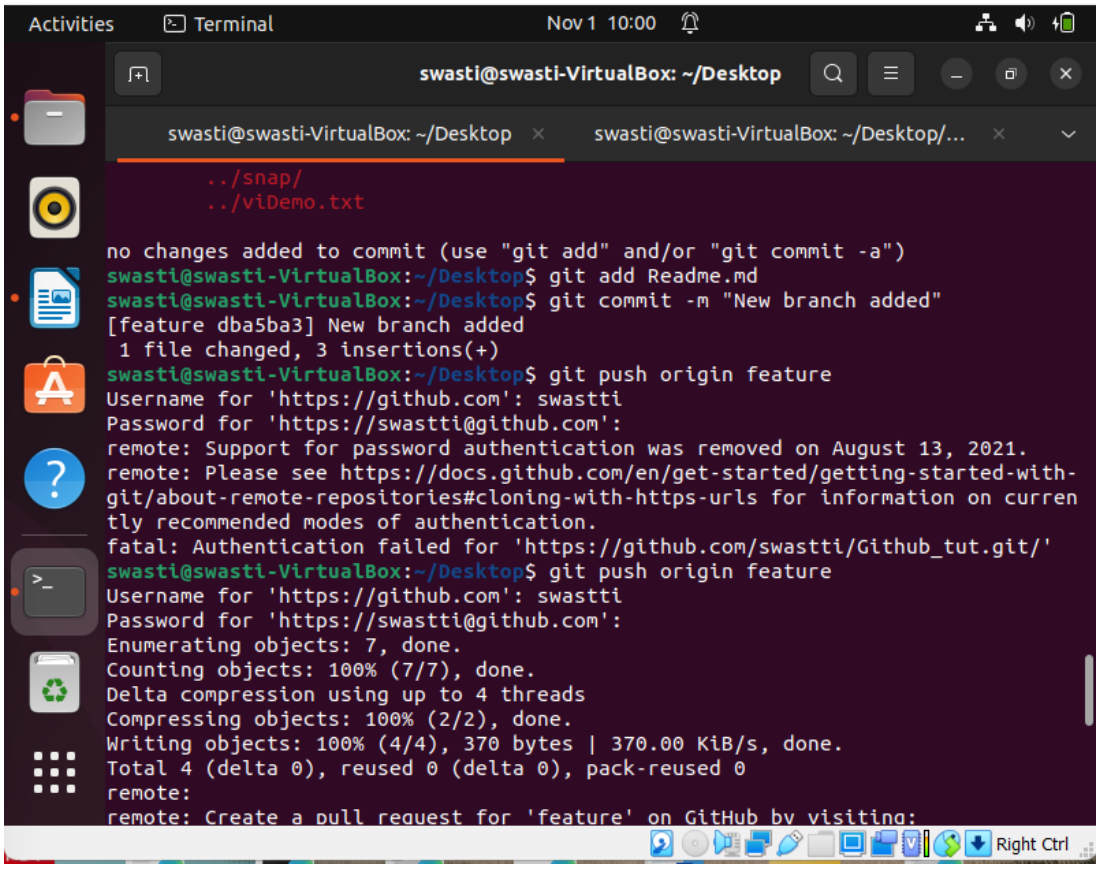
- ✓ Distributed version control
- ✓ Coordinates work between multiple developers
- ✓ Who made what changes and when
- ✓ Revert back at any time
- ✓ Local & remote repos

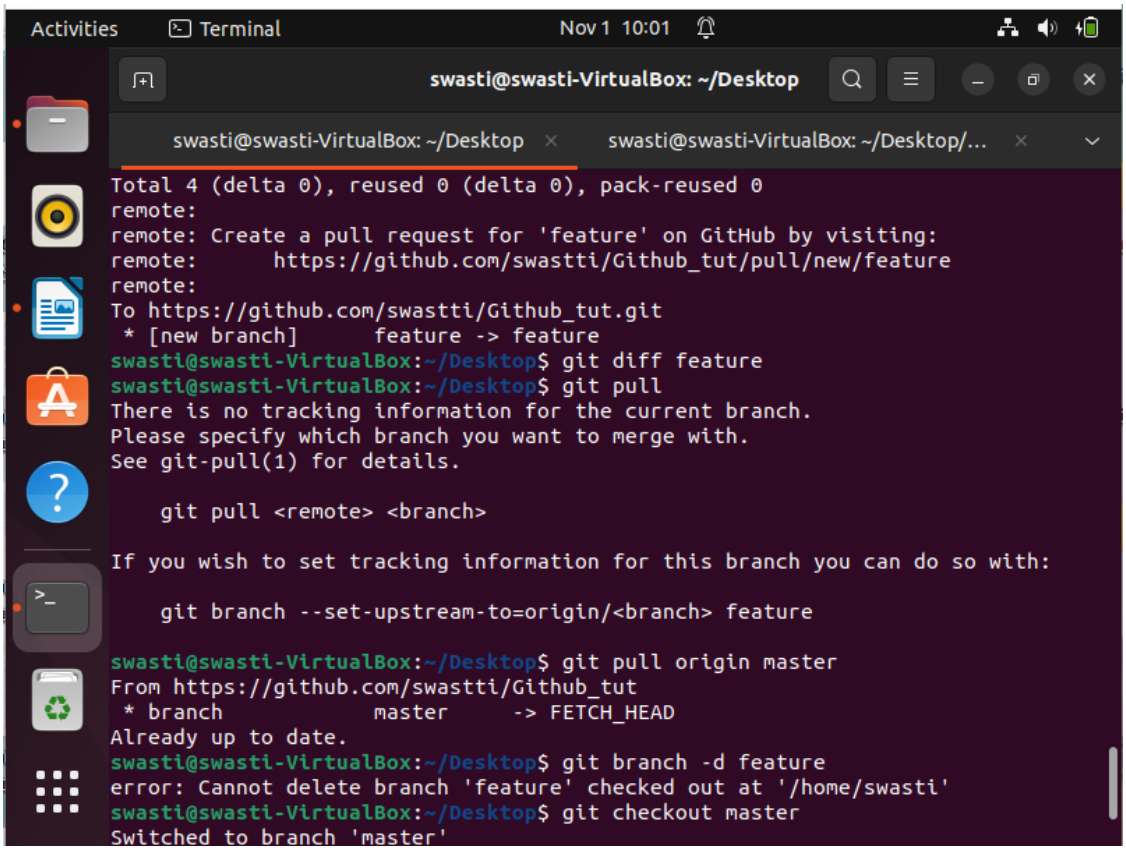


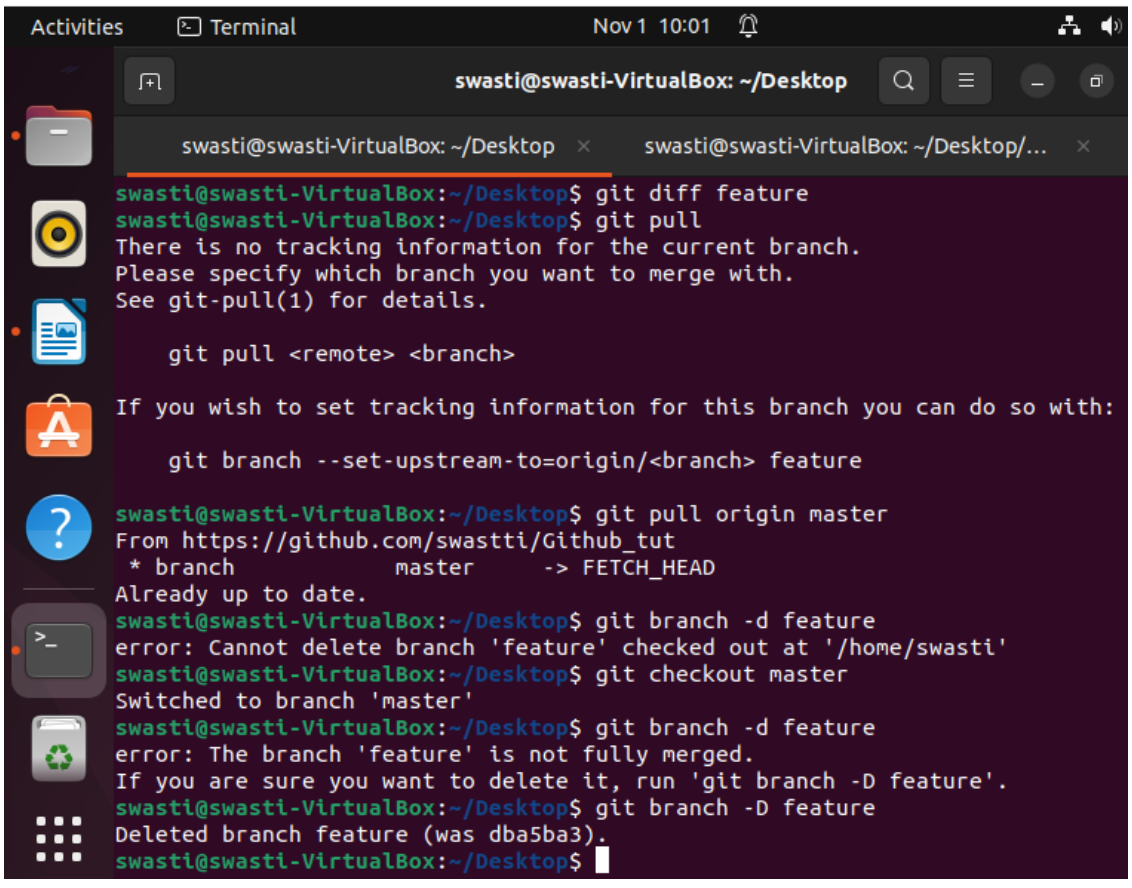

Activities
Terminal
Nov 1 09:59
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop/...
error: open("Desktop/story.txt"): Permission denied
error: unable to index file 'Desktop/story.txt'
fatal: adding files failed
swasti@swasti-VirtualBox:~/Desktop$ ls
a image names Readme.md
helloworld.odt lp.gz 'Old Firefox Data' story.txt
swasti@swasti-VirtualBox:~/Desktop$ git add Readme.md
swasti@swasti-VirtualBox:~/Desktop$ gi commit -m "First Commit"
gi: command not found
swasti@swasti-VirtualBox:~/Desktop$ git commit -m "First Commit"
[master (root-commit) 5d8ec48] First Commit
1 file changed, 2 insertions(+)
create mode 100644 Desktop/Readme.md
swasti@swasti-VirtualBox:~/Desktop$ git status
On branch master
Untracked files:
(use "git add <file>..." to include in what will be committed)
../.ViDemo.txt.swp
../.bash_history
../.bash_logout
../.bashrc
../.cache/
../.config/
../.gitconfig
../.gnupg/
../.lesshst
../.local/
Right Ctrl

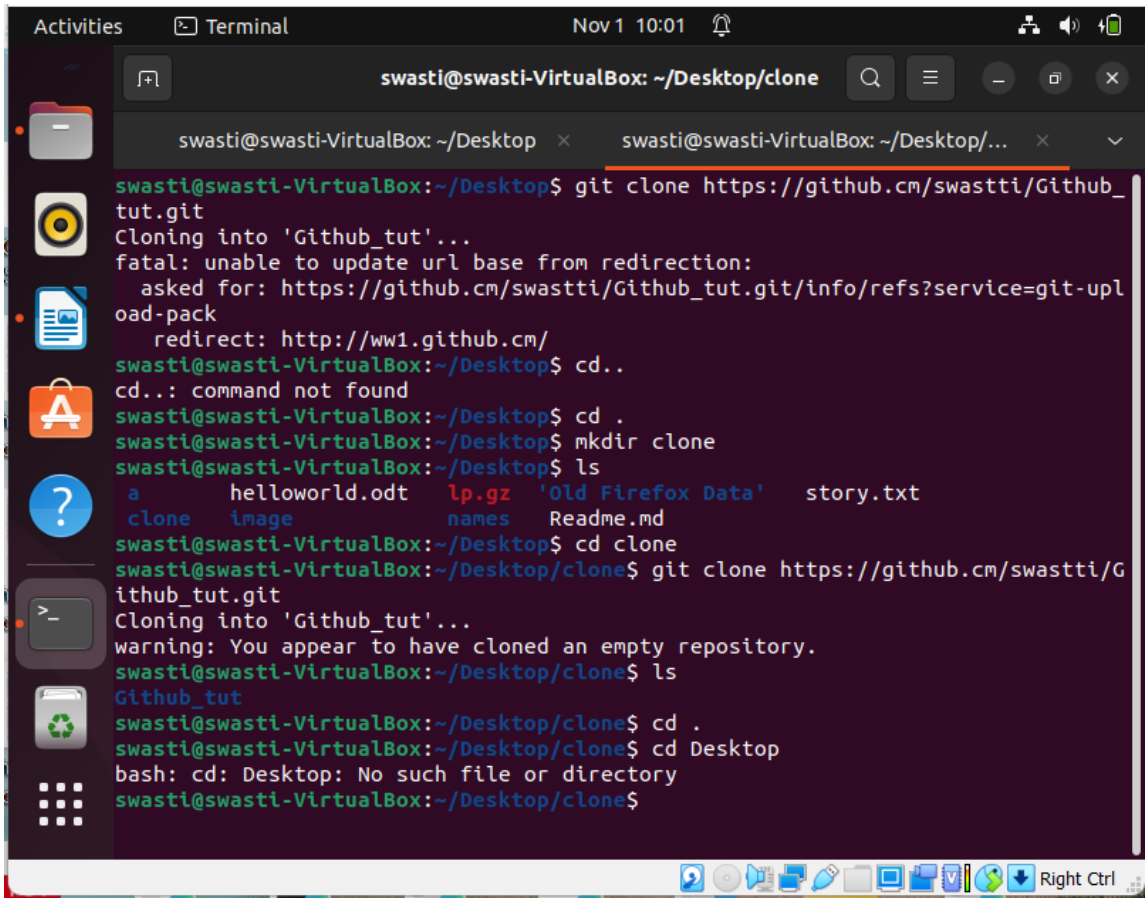

Activities
Terminal
Nov 1 10:00
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop/...
nothing added to commit but untracked files present (use "git add" to track)
swasti@swasti-VirtualBox:~/Desktop$ git push origin master
fatal: 'origin' does not appear to be a git repository
fatal: Could not read from remote repository.
Please make sure you have the correct access rights
and the repository exists.
swasti@swasti-VirtualBox:~/Desktop$ git remote add origin https://github.com/swastti/Github_tut.git
swasti@swasti-VirtualBox:~/Desktop$ git push origin master
Username for 'https://github.com': swastti
Password for 'https://swastti@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 308 bytes | 308.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/swastti/Github_tut.git
* [new branch] master -> master
swasti@swasti-VirtualBox:~/Desktop$ git log
commit 5d8ec48100943d0238a21e78d12074d71d7212ea (HEAD -> master, origin/master)
Author: swastti <jainswati1603@gmail.om>
Date: Tue Nov 1 09:35:49 2022 +0530
First Commit
swasti@swasti-VirtualBox:~/Desktop$ git branch
Right Ctrl


Activities
Terminal
Nov 1 10:00
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop/...
First Commit
swasti@swasti-VirtualBox:~/Desktop$ git branch
* master
swasti@swasti-VirtualBox:~/Desktop$ git checkout -b feature
Switched to a new branch 'feature'
swasti@swasti-VirtualBox:~/Desktop$ git checkout master
Switched to branch 'master'
swasti@swasti-VirtualBox:~/Desktop$ git checkout feature
Switched to branch 'feature'
swasti@swasti-VirtualBox:~/Desktop$ gedit Readme.md
swasti@swasti-VirtualBox:~/Desktop$ git status
On branch feature
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
modified: Readme.md
Untracked files:
(use "git add <file>..." to include in what will be committed)
../.ViDemo.txt.swp
../.bash_history
../.bash_logout
../.bashrc
../.cache/
../.config/
../.gitconfig


Activities
Terminal
Nov 1 10:00
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop/...
../snap/
../viDemo.txt
no changes added to commit (use "git add" and/or "git commit -a")
swasti@swasti-VirtualBox:~/Desktop$ git add Readme.md
swasti@swasti-VirtualBox:~/Desktop$ git commit -m "New branch added"
[feature dba5ba3] New branch added
1 file changed, 3 insertions(+)
swasti@swasti-VirtualBox:~/Desktop$ git push origin feature
Username for 'https://github.com': swastti
Password for 'https://swastti@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/en/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-urls for information on currently recommended modes of authentication.
fatal: Authentication failed for 'https://github.com/swastti/Github_tut.git/'
swasti@swasti-VirtualBox:~/Desktop$ git push origin feature
Username for 'https://github.com': swastti
Password for 'https://swastti@github.com':
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 370 bytes | 370.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
Right Ctrl


Activities
Terminal
Nov 1 10:01
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop/...
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote: https://github.com/swastti/Github_tut/pull/new/feature
remote:
To https://github.com/swastti/Github_tut.git
* [new branch] feature -> feature
swasti@swasti-VirtualBox:~/Desktop$ git diff feature
swasti@swasti-VirtualBox:~/Desktop$ git pull
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.
git pull <remote> <branch>
If you wish to set tracking information for this branch you can do so with:
git branch --set-upstream-to=origin/<branch> feature
swasti@swasti-VirtualBox:~/Desktop$ git pull origin master
From https://github.com/swastti/Github_tut
* branch master -> FETCH_HEAD
Already up to date.
swasti@swasti-VirtualBox:~/Desktop$ git branch -d feature
error: Cannot delete branch 'feature' checked out at '/home/swasti'
swasti@swasti-VirtualBox:~/Desktop$ git checkout master
Switched to branch 'master'


Activities
Terminal
Nov 1 10:01
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop
swasti@swasti-VirtualBox: ~/Desktop/...
swasti@swasti-VirtualBox:~/Desktop$ git diff feature
swasti@swasti-VirtualBox:~/Desktop$ git pull
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.
git pull <remote> <branch>
If you wish to set tracking information for this branch you can do so with:
git branch --set-upstream-to=origin/<branch> feature
swasti@swasti-VirtualBox:~/Desktop$ git pull origin master
From https://github.com/swastti/Github_tut
* branch master -> FETCH_HEAD
Already up to date.
swasti@swasti-VirtualBox:~/Desktop$ git branch -d feature
error: Cannot delete branch 'feature' checked out at '/home/swasti'
swasti@swasti-VirtualBox:~/Desktop$ git checkout master
Switched to branch 'master'
swasti@swasti-VirtualBox:~/Desktop$ git branch -d feature
error: The branch 'feature' is not fully merged.
If you are sure you want to delete it, run 'git branch -D feature'.
swasti@swasti-VirtualBox:~/Desktop$ git branch -D feature
Deleted branch feature (was dba5ba3).
swasti@swasti-VirtualBox:~/Desktop$



Docker is a centralized platform for packaging, deploying, and running applications. Before Docker, many users face the problem that a particular code is running in the developer's system but not in the user's system. So, the main reason to develop docker is to help developers to develop applications easily, ship them into containers, and can be deployed anywhere.

What is Docker?

Docker is an **open-source centralized platform designed** to create, deploy, and run applications. Docker uses **container** on the host's operating system to run applications. It allows applications to use the same **Linux kernel** as a system on the host computer, rather than creating a whole virtual operating system. Containers ensure that our application works in any environment like development, test, or production.

Docker Container and Image

Docker images are the “source code” for our containers; we use them to build containers. They can have software pre-installed which speeds up deployment. They are portable, and we can use existing images or build our own.

Containers are the organizational units and one of the Docker basics concept. When we build an image and start running it; we are running in a container. The container analogy is used because of the portability of the software we have running in our container. We can move it, in other words, “ship” the software, modify, manage, create or get rid of it, destroy it, just as cargo ships can do with real containers.

In simple terms, an image is a template, and a container is a copy of that template. You can have multiple containers (copies) of the same image.

Docker container is a running instance of an image. You can use Command Line Interface (CLI) commands to run, start, stop, move, or delete a container. You can also provide configuration for the network and environment variables. Docker container is an isolated and secure application platform, but it can share and access to resources running in a different host or container.

An image is a read-only template with instructions for creating a Docker container. A docker image is described in text file called a **Dockerfile**, which has a simple, well-defined syntax. An image does not have states and never changes. Docker Engine provides the core Docker technology that enables images and containers.

You can understand container and image with the help of the following command.

1. $ docker run hello-world

The above command **docker run hello-world** has three parts.

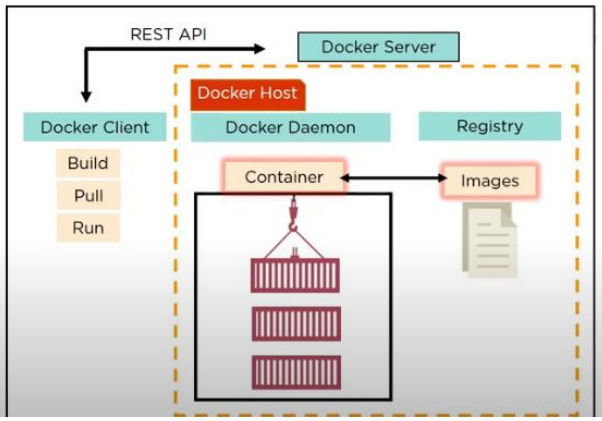
1) **docker:** It is docker engine and used to run docker program. It tells to the operating system that you are running docker program.

2) **run:** This subcommand is used to create and run a docker container.

3) **hello-world:** It is a name of an image. You need to specify the name of an image which is to load into the container.

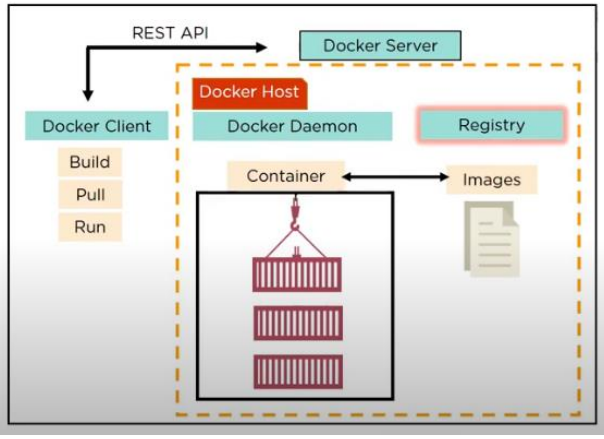
Docker containers run on top of the host’s Operation system. This helps you to improves efficiency and security. Moreover, we can run more containers on the same infrastructure than we can run Virtual machines because containers use fewer resources.

1 Explain the architecture of Docker



- Docker uses a client-server architecture
- Docker Client is a service which runs a command. The command is translated using REST API and is sent to the Docker Daemon (server)
- Docker Daemon accepts the request and interacts with the operating system in order to build Docker Images and run Docker containers
- A Docker Image is a template of instruction which is used to create containers

1 Explain the architecture of Docker



- Docker container is an executable package of application and its dependencies together
- Docker registry is a service to host and distribute Docker Images among users

2 What are the advantages of Docker over Virtual machine

Criteria	Virtual Machine	Docker
Memory space	Occupies a lot of memory space	Docker Containers occupy less space
Boot-up time	Long boot-up time	Short boot-up time
Performance	Running multiple virtual machines leads to unstable performance	Containers have a better performance as they are hosted in a single Docker engine
Scaling	Difficult to scale up	Easy to scale up
Efficiency	Low efficiency	High efficiency
Portability	Compatibility issues while porting across different platforms	Easily portable across different platforms
Space allocation	Data volumes cannot be shared	Data volumes can be shared and reused among multiple containers

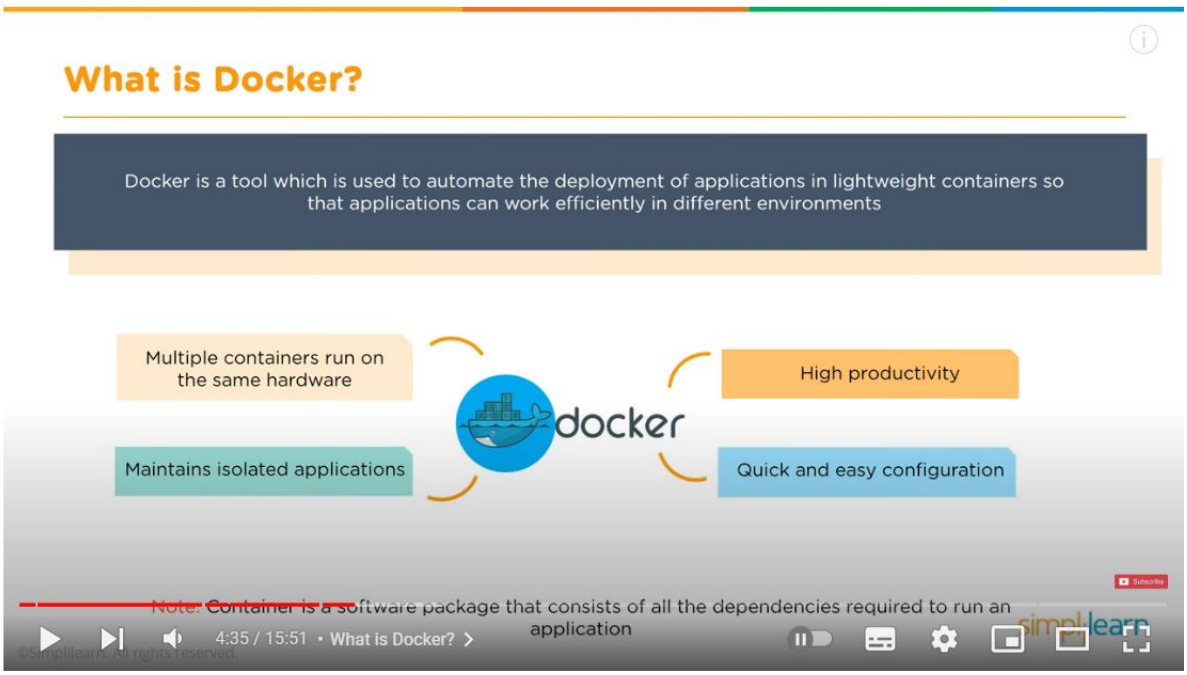
7 Differences between Docker Image and Docker Container

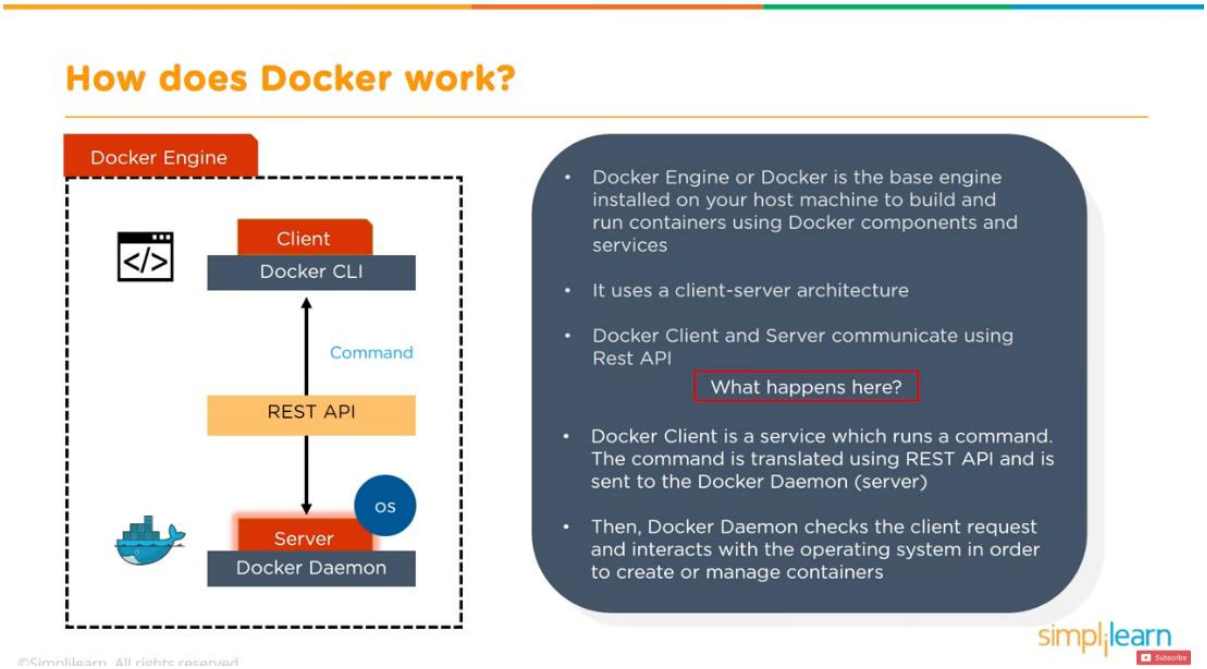
Docker Images

- Docker Images are templates of Docker Containers
- An image is built using a Dockerfile
- It is stored in a Docker repository or a Docker hub
- The image layer is a read only filesystem

Docker Container

- Containers are runtime instances of a Docker Image
- Containers are created using Docker Images
- They are stored in the Docker daemon
- Every container layer is a read-write filesystem





Command	Description
docker info	Information Command
docker pull	Download an image
docker run -i -t image_name /bin/bash	Run image as a container
docker start our_container	Start container
docker stop container_name	Stop container
docker ps	List of al running containers
docker stats	Container information
docker images	List of images downloaded

Command	Description
Docker Cleanup	Kill all running containers.

Containers

Use `docker container my_command`

`create` — Create a container from an image.
`start` — Start an existing container.
`run` — Create a new container and start it.
`ls` — List running containers.
`inspect` — See lots of info about a container.
`logs` — Print logs.
`stop` — Gracefully stop running container.
`kill` —Stop main process in container abruptly.
`rm`— Delete a stopped container.

Images

Use `docker image my_command`

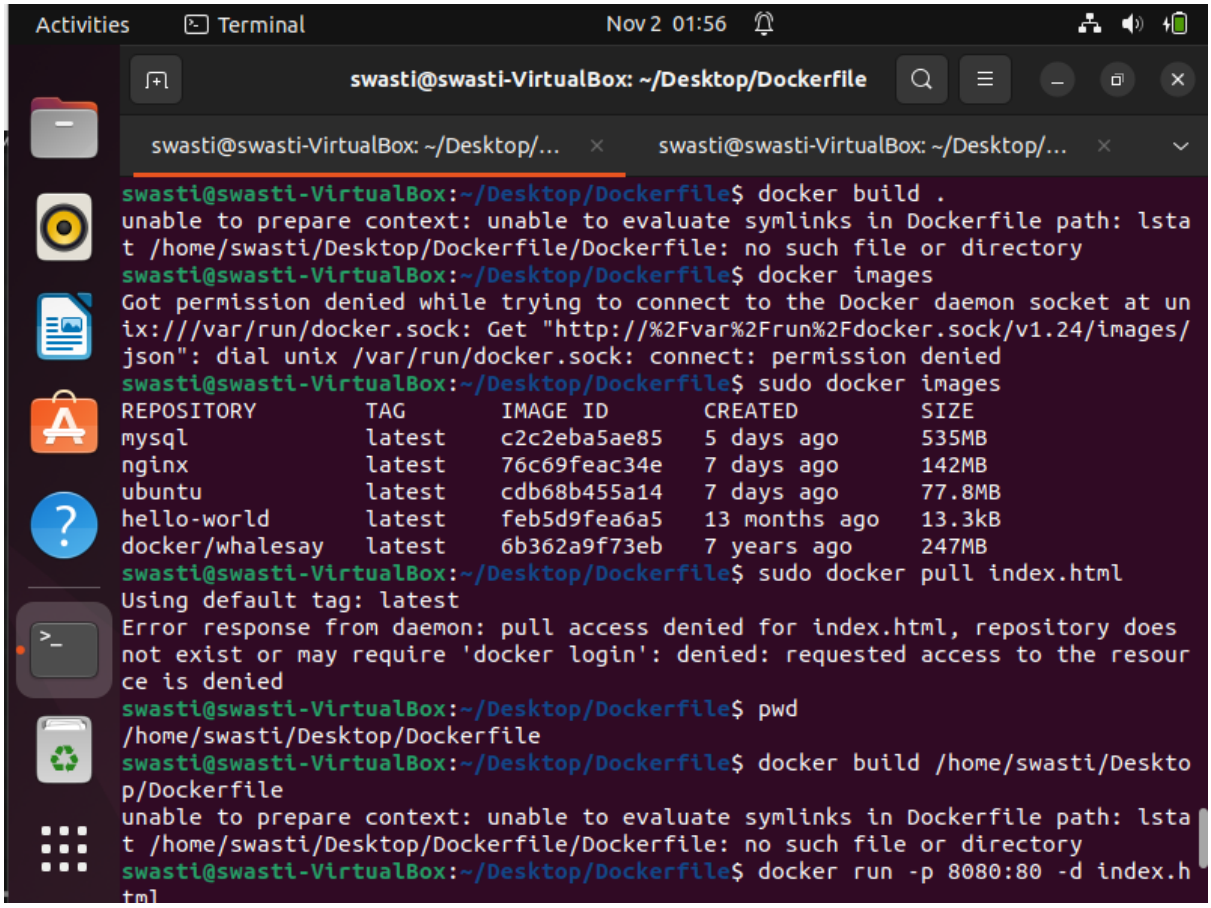
`build` — Build an image.
`push` — Push an image to a remote registry.
`ls` — List images.
`history` — See intermediate image info.
`inspect` — See lots of info about an image, including the layers.
`rm` — Delete an image.

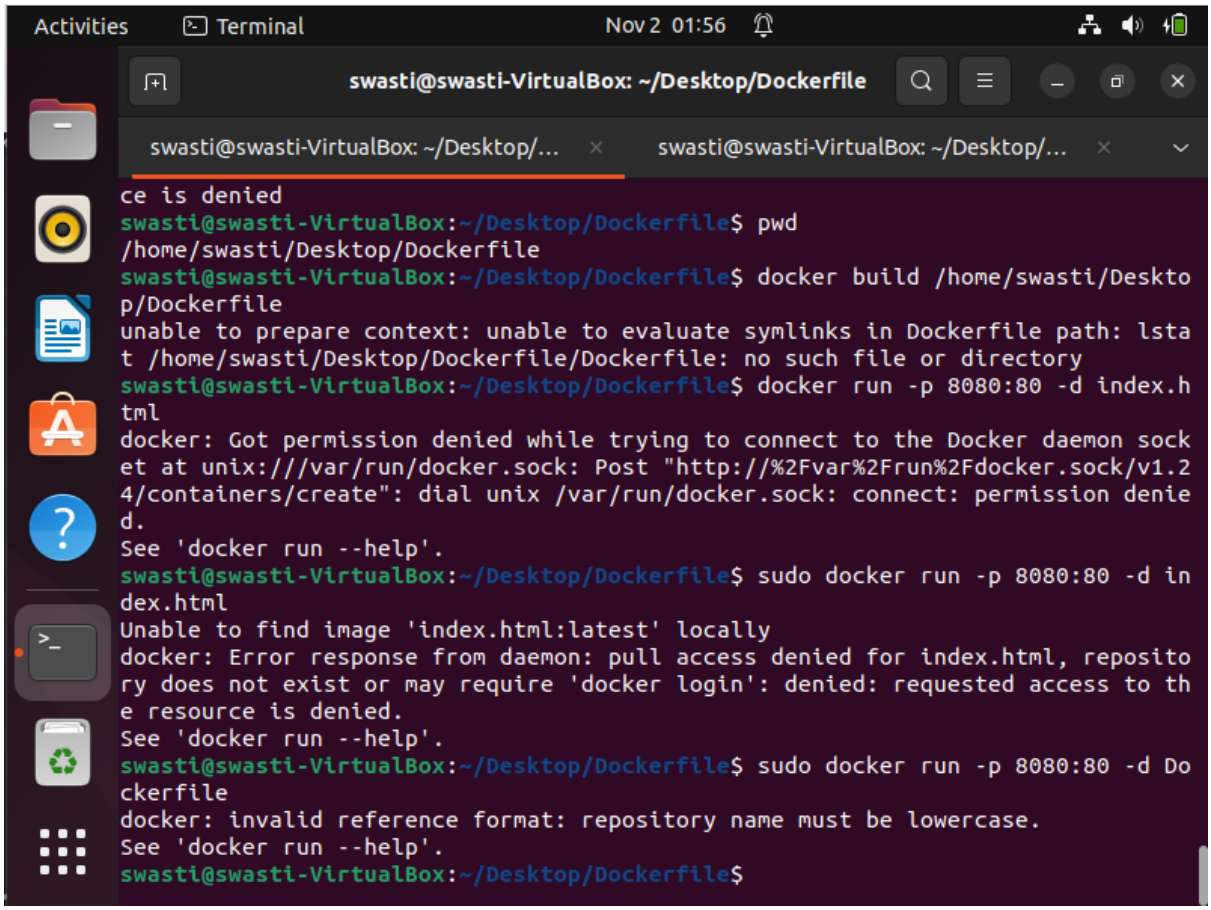
Misc

`docker version` — List info about your Docker Client and Server versions.

`docker login` — Log in to a Docker registry.

`docker system prune` — Delete all unused containers, unused networks, and dangling images.





01 What is Jenkins?

Jenkins is an open-source automation tool written in Java with plugins built for Continuous Integration purposes.

03 Define the process of Jenkins.

- *Commit the changes*
- *Detect source code changes*
- *The build either passes or fails*
- *Generate feedback*

10 What is Maven? What is the benefit of integrating Maven with Jenkins?

Maven is a build management tool. It uses a simple pom.xml to configure all the dependencies needed to build, test and run the code.