Delhi Skill and Entrepreneurship University
Govt. of NCT of Delhi

INTRODUCTION TO DevOps

PRACTICAL FILE

NAME – SHREY THAKUR

ROLL NO.- 41221160

SUBMITTED TO :

BUSHRA MAM

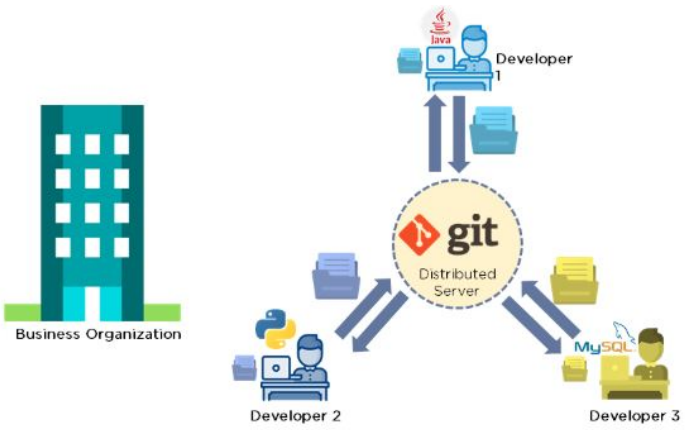
S.no	Name of practical	Date	Sign
1.	Background study, Installation and configuration of Git		
2.	• Git 3 stage Architecture • Generating SSH keys and agents on local machine • Adding origin(Github repository) • Git Tracking and use of basic commands – init, status and log		
3.	Application of Git commands – add, commit, push, pull		
4.	Perform the following in Git- • Removal of Git repository/tracking • git clone • .gitignore (scenario based usage(e.g.: a file/directory))		
5.	Show the use of - • Git fetch • git diff command, • git difftool • git restore command (to Restore files(e.g.: 2 out of total 5 files) from staging area that you don't want to commit)		
6.	Perform following code changes using Git commands (Undoing/ Reverting/ Resetting) • git checkout(undo uncommitted changes) • git revert (undo committed changes) • git reset (resetting changes to a previous timestamp)		
7.	Git branching • Adding branches • Switching branches • Merging branches • Deleting branches(from git as well as github too) • Git rebase • Usage of squash • Git cherry-pick • git tag • git upstream and downstream Note : show the status of files(before and after code changes) on both git and github after adding the branches and deleting the branches.		
8.	Usage of git stash(for temporary commits) • git stash • git stash save • git stash list • git stash apply • git stash changes • git stash pop • git stash drop • git stash clear • git stash branch		

9.	• Moving and renaming files in git • HEAD and detached HEAD		
10.	Rewrite history in git • git rebase • git reset –hard		
11.	Background study, Installation and configuration of Jenkins		
12.	Perform following in Jenkins(with snapshots of each step):- • Design Jenkins Architecture • Creating Jobs in Jenkins(e.g. : with status- success, partial, failure) • Adding slave nodes to Jenkins • Fetching content from github in Jenkins • Build the pipeline of fetched jobs from github in/using Jenkins • Build a pipeline using script in Jenkins		
13.	Background study, Installation and configuration of Docker on windows/ Linux platform /using play with docker platform		
14.	Execute Docker Basic Commands(with snapshots):- • docker images • docker pull • docker run • docker ps (with flags such as -a, -l, -q) • docker stop (for single or multiple containers) • docker rm(with flags such as -f, -v, -l) • docker commit		
15.	Execute following Docker Image Commands(with relevant snapshots):- • docker build command • docker pull command • docker images command • docker inspect command • docker push command • docker save command • docker rmi command		
16.	Design Docker container life cycle and execute following Docker Container Commands(with relevant snapshots):- • docker attach command • docker ps command • docker start and restart • docker port mapping • docker container command • docker container inspect infinite Command • docker exec command (with flags such as -d, -i, -e) • docker cp command		
17.			
18.			

PRACTICAL 1 : Background study, Installation and configuration of Git

Git is a version control system that you download onto your computer. It is essential that you use Git if you want to collaborate with other developers on a coding project or work on your own project.

Git is DevOps tool is used for source code management. It is a free and open-source version control system used to handle small to very large projects efficiently. Git is used to tracking changes in the source code, enabling multiple developers to work together on non-linear development. Linus Torvalds created Git in 2005 for the development of the Linux kernel.



- Installation of Git

Homebrew
Install homebrew if you don't already have it, then:
$ brew install git

```
(base) monika@Monikas-MacBook-Air ~ % brew install git
Running `brew update --auto-update`...
==> Auto-updated Homebrew!
Updated 2 taps (homebrew/core and homebrew/cask).
==> New Formulae
apprise                    checkdmarc                 falco                      jsor
zzi
asn                        cloudfox                   fwupd                      kosl
an-utility-libraries
biome                      colmap                     ggshield                   ldee
ext
blake3                     cyclonedx-gomod            gotestwaf                  libi
caracal                    cyclonedx-python           helidon                    libm
cargo-docset               dovi_tool                  iocextract                 llvm
chaoskube                  eza                        jr                         mfem
==> New Casks
aifun              batteryboi         cloudnet           expo-orbit         meld-studio
  zui
akuity             chainner           draw-things        floorp             monotype
ava                clinq              dropshelf          hhkb               mutedeck

You have 5 outdated formulae installed.

Warning: git 2.42.0 is already installed and up-to-date.
To reinstall 2.42.0, run:
  brew reinstall git
(base) monika@Monikas-MacBook-Air ~ % git --version
git version 2.42.0
(base) monika@Monikas-MacBook-Air ~ %
```

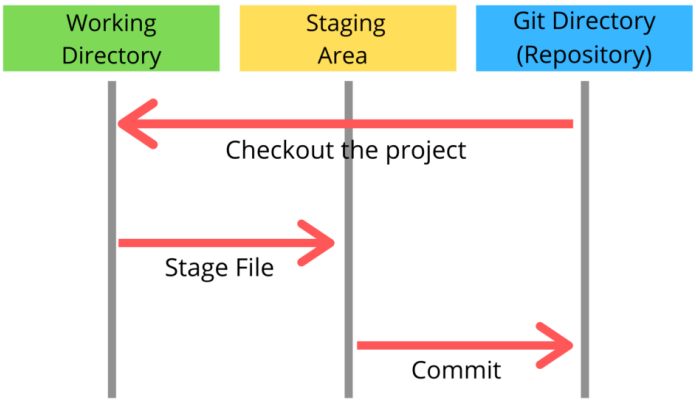
- Configuration of git

```
BASH-3.2$ git config user.name "Shrey"
BASH-3.2$ git config user.email "shreythakur0703@gmail.com"
BASH-3.2$
```

PRACTICAL 2 :

- Git 3 stage Architecture

Many VCS's use a two-tier architecture i.e a repository and a working copy. Git uses three-tier architecture i.e a working directory, staging area and local repository. The three stages of git can store different(or the same) states of the same code in each stage.



- Generating SSH keys and agents on local machine

```
BASH-3.2$ cd ~/.ssh
BASH-3.2$ LS
known_hosts
BASH-3.2$ ssh-keygen -o
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/monika/.ssh/id_rsa): /home
/schacon/.ssh
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Saving key "/home/schacon/.ssh" failed: No such file or directory
BASH-3.2$
```

- Adding origin(Github repository)

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git remote add origin https://github.com/sapna010404/LAB.git
error: remote origin already exists.
```

- Git Tracking and use of basic commands – init, status and log

```
bash-3.2$ git init
Reinitialized existing Git repository in /Users/monika/Desktop/git1/.git/
bash-3.2$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        pp.pdf

nothing added to commit but untracked files present (use "git add" to track)
bash-3.2$ git add --a
bash-3.2$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   pp.pdf
```

PRACTICAL 3: Application of Git commands – add, commit, push, pull

~git add

Add a file to staging (Index) area:
$ git add Filename
Add all files of a repo to staging (Index) area:
$ git add .

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        1.txt

nothing added to commit but untracked files present (use "git add" to track)

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro (master)
$ git add 1.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro (master)
$ git commit -m "file added"
[master (root-commit) 32010ea] file added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt
```

~git commit

Record or snapshots the file permanently in the version history **with a message**.

$ git commit -m " Commit Message"

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro (master)
$ git commit -m "file added"
[master (root-commit) 32010ea] file added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 1.txt
```

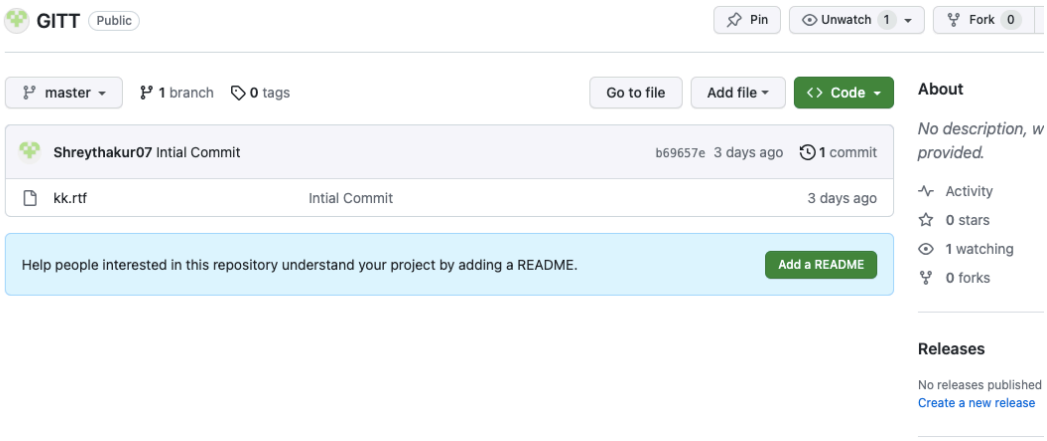
git push

Transfer the commits from your local repository to a remote server.

Push data to the remote server:

$git push -u origin master

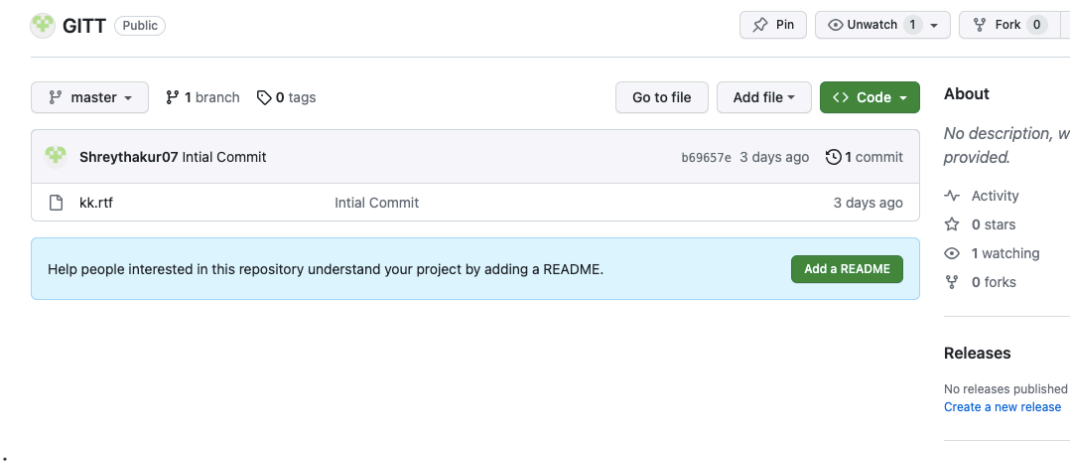
```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 407 bytes | 407.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:Rishi-ii/Devops.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```



PRACTICAL 4: Perform the following in Git-

- Removal of Git repository/tracking

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro (master)
$ mkdir test_directory

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro (master)
$ cd test_directory

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro/test_directory (master)
$ git init
Initialized empty Git repository in C:/Users/           /Desktop/pro/test_directory/.git/

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro/test_directory (master)
$ echo "content" > 1.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro/test_directory (master)
$ echo "content" > 2.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro/test_directory (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        1.txt
        2.txt

nothing added to commit but untracked files present (use "git add" to track)

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro/test_directory (master)
$ git add .

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro/test_directory (master)
$ git commit -m "commitedd"
[master (root-commit) 0cfe1da] commitedd
 2 files changed, 2 insertions(+)
 create mode 100644 1.txt
 create mode 100644 2.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/pro/test_directory (master)
$ rm -r .git
rm: remove write-protected regular file '.git/objects/0c/fe1da22a57f338f7d5b3c2dcc5707aaa6adf10'? y
rm: remove write-protected regular file '.git/objects/42/50168dc8e2c8b0463ffb6589f1347fa983c088'? y
rm: remove write-protected regular file '.git/objects/d9/5f3ad14dee633a758d2e331151e950dd13e4ed'? y
```



git clone

```
bash-3.2$ git clone https://github.com/Shreythakur07/GITT.git
Cloning into 'GITT'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
bash-3.2$ ls
GITT    kk.rtf
bash-3.2$
```

- .gitignore (scenario based usage(e.g.: a file/directory))

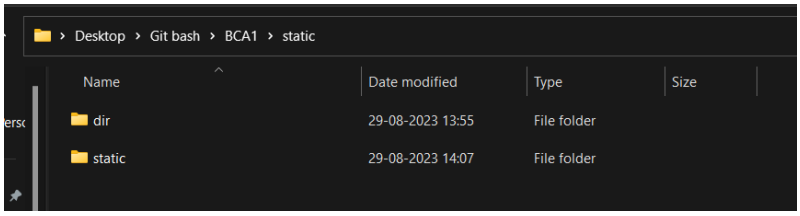
```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ touch error.log

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git status
On branch master
Your branch is ahead of 'origin/master'
by 3 commits.
  (use "git push" to publish your local
commits)

Untracked files:
  (use "git add <file>..." to include in
 what will be committed)
        error.log

nothing added to commit but untracked fi
les present (use "git add" to track)
```

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MI
NGW64 /d/GIT DevOps (master)
$ touch .gitignore

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 3 commits.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        error.log

nothing added to commit but untracked files present (use "git add" to track)
```

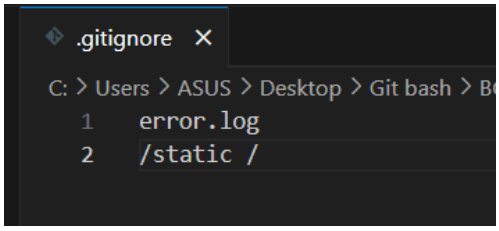
```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 3 commits.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore

nothing added to commit but untracked files present (use "git add" to track)

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git add .gitignore
```

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git add .gitignore

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 3 commits.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   .gitignore
```

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git commit -m "files added"
[master b5d1455] files added
 1 file changed, 1 insertion(+)
 create mode 100644 .gitignore
```



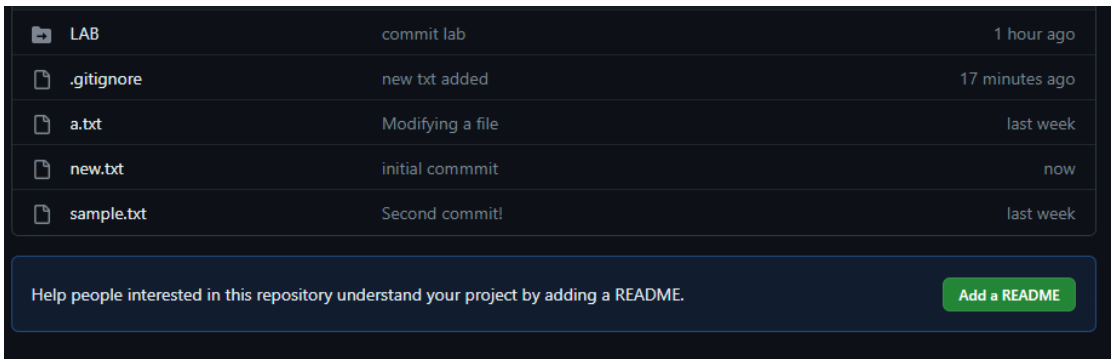
Open the .gitignore file once again and modify it.



Check the status again.

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/BCA1 (master)
$ git status
On branch master
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:   .gitignore

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        static/

no changes added to commit (use "git add" and/or "git commit -a")
```

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ mkdir static
mkdir: cannot create directory ‘static’: File exists

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ nano gitignore

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 4 commits.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   .gitignore
        new file:   gitignore

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   .gitignore
        deleted:    gitignore


DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git add .

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 4 commits.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   .gitignore


DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git commit -m " "
Aborting commit due to empty commit message.

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ mkdir
```

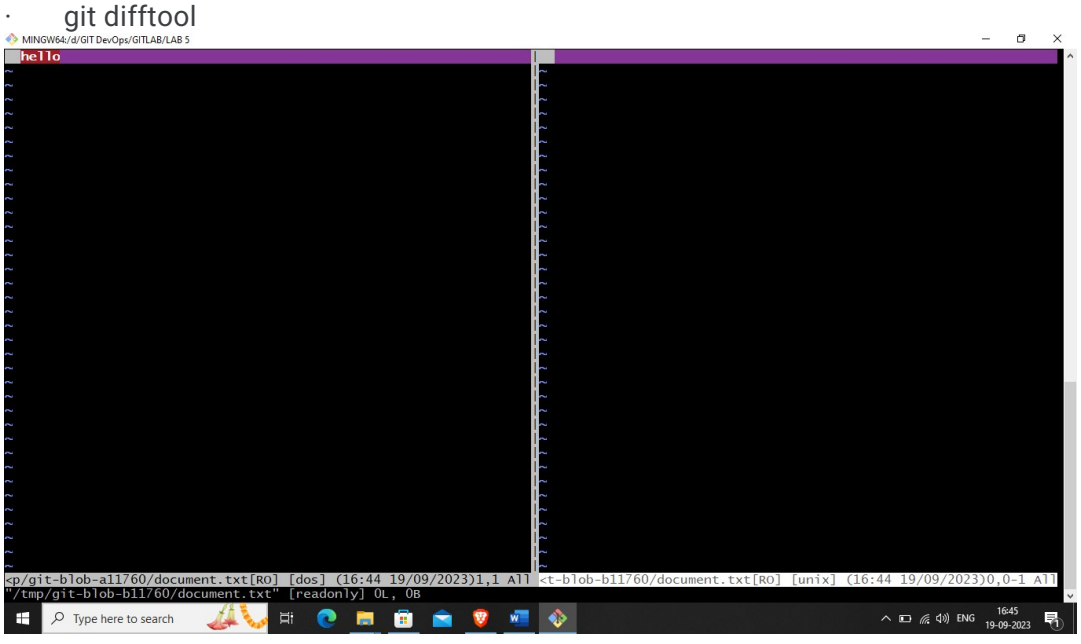


Practical 5: Show the use of -

· git diff command

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 4 commits.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   .gitignore
        new file:   new.txt

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:   new.txt


DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ git diff
diff --git a/new.txt b/new.txt
index 85954ea..896ddd8 100644
--- a/new.txt
+++ b/new.txt
@@ -2,4 +2,7 @@
 2
 3
 4
-5
\ No newline at end of file
+5
+6
+7
+8
\ No newline at end of file
```

· git difftool



· git restore command (to Restore files(e.g.: 2 out of total 5 files) from staging area that you don’t want to commit)

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 4
)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   ab.txt
        modified:   abc.txt


DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 4
)
$ git restore --staged abc.txt

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 4
)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   ab.txt

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:   abc.txt
```

PRACTICAL 6: Perform following code changes using Git commands (Undoing/ Reverting/ Resetting)

· git checkout(undo uncommitted changes)

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps (master)
$ cd LAB

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/LAB (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/LAB (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

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:   new.txt

no changes added to commit (use "git add" and/or "git commit -a")

ASUS@[illegible] MINGW64 /d/BCA5 (master)
$ git checkout example.txt
Updated 1 path from the index
```

· git revert (undo committed changes)

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 3 (master
)
$ git init
Initialized empty Git repository in D:/GIT DevOps/GITLAB/LAB 3/.git/

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 3 (master
)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        newfile.txt

nothing added to commit but untracked files present (use "git add" to track)

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 3 (master
)
$ git add .
```

git reset (resetting changes to a previous timestamp)

The git reset command is used to undo the changes in your working directory and get back to a specific commit while discarding all the commits made after that one.

You can use multiple options along with git reset, but these are the main ones. Each one is used depending on a specific situation: git reset --soft, git reset --mixed, and git reset --hard

Git reset --soft

Aims to change the Head reference to a specific commit. add a file to the commit, we can move back using the --soft .

Git reset --mixed

This is the default argument for git reset. Running this command has two impacts:

(1) uncommit all the changes and (2) unstage them

Git reset –hard

When using the hard reset on a specific commit, it forces the HEAD to get back to that commit and deletes everything else after that. Considering this as a project which contains files like data, filtered data, and result.

Scenario1: For git reset –soft

- --soft

By changing the HEAD, the files will go to the area where changes were made and need to be committed. (Working+staging)

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT
)
$ git add data.txt fdata.txt

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT
)
$ git commit -m "added 2 files"
[master (root-commit) 11bb991] added 2 files
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 data.txt
 create mode 100644 fdata.txt
```

```
$ git log --oneline
11bb991 (HEAD -> master) added 2 files
```

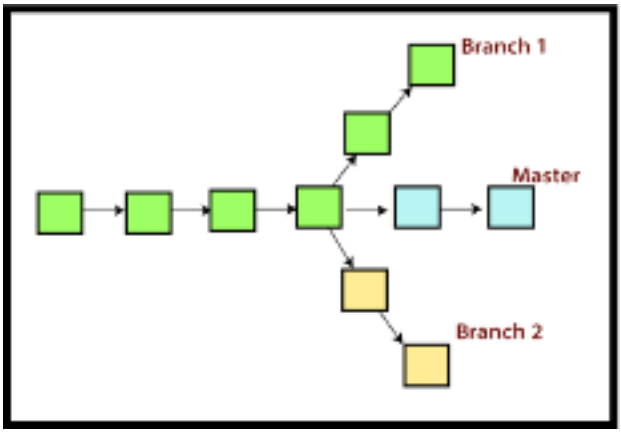
```
$ git add  thirdfile.txt

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/G
)
$ git commit -m "all files added"
[master 8ec7333] all files added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 thirdfile.txt
```

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW6
)
$ git log --oneline
8ec7333 (HEAD -> master) all files added
11bb991 added 2 files
```

```
$ git reset --soft 11bb991

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MING
)
$ git log --oneline
11bb991 (HEAD -> master) added 2 files
```

PRACTICAL 7: Git branching

Branching in Git is a powerful feature that allows you to work on multiple independent lines of development within the same repository. Each branch represents a separate line of development and can have its own set of commits, changes, and history.



Adding branches

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ ls
Ks.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git status
fatal: not a git repository (or any of the parent directories): .git

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git init
Initialized empty Git repository in C:/Users/singh/OneDrive/Desktop/DevOps/.git/

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Ks.txt

nothing added to commit but untracked files present (use "git add" to track)
```


```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ ls
Ks.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git status
fatal: not a git repository (or any of the parent directories): .git

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git init
Initialized empty Git repository in C:/Users/singh/OneDrive/Desktop/DevOps/.git/

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Ks.txt

nothing added to commit but untracked files present (use "git add" to track)
```

```
MINGW64:/c/Users/singh/OneDrive/Desktop/DevOps

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git commit
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Ks.txt

nothing added to commit but untracked files present (use "git add" to track)

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git add ks.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git commit
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Ks.txt

nothing added to commit but untracked files present (use "git add" to track)
```

```
MINGW64:/c/Users/singh/OneDrive/Desktop/DevOps

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git remote add origin "https://github.com/kullu9319/bca.git"

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 20 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 215 bytes | 215.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/kullu9319/bca.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git branch NEW

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git checkout NEW
Switched to branch 'NEW'
```

- Switching branches

```
MINGW64:/c/Users/singh/OneDrive/Desktop/DevOps

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git remote add origin "https://github.com/kullu9319/bca.git"

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 20 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 215 bytes | 215.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/kullu9319/bca.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git branch NEW

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git checkout NEW
Switched to branch 'NEW'
```

- Merging branches

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git branch
* NEW
  master

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git merge NEW
Updating 2e74942..415b81b
Fast-forward
 Ks.txt | 1 +
 ll.txt | 0
 pp.txt | 1 +
 3 files changed, 2 insertions(+)
 create mode 100644 ll.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git push -u origin master
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/kullu9319/bca.git
   2e74942..415b81b  master -> master
branch 'master' set up to track 'origin/master'.
```

· Deleting branches(from git as well as github too)

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git status
On branch NEW
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:   Ks.txt
        modified:   pp.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        ll.txt

no changes added to commit (use "git add" and/or "git commit -a")

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git add .

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git commit -m
error: switch `m' requires a value

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git commit -m "commit"
[NEW 415b81b] commit
 3 files changed, 2 insertions(+)
 create mode 100644 ll.txt
```

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git branch -d NEW
Deleted branch NEW (was 415b81b).

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git branch
* master

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git push origin --delete NEW
To https://github.com/kullu9319/bca.git
 - [deleted]         NEW
```

Git cherry-pick

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git cherry-pick e43483b4cddedd1b28ac358857ca75d23326e1f9
[master 16d018c] commit for the project's main branch
 Date: Mon Sep 16 12:21:55 2019 +0530
 1 file changed, 1 insertion(+)
 create mode 100644 mastercommit.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$
```

 git tag

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git tag v1.0

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git push origin --tags
Everything up-to-date

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git tag v1.5
```

 git upstream and downstream

- Upstream: Refers to the original repository from which a clone was made. It represents the source or parent repository.
- Downstream: Refers to forks or clones of a repository. If you fork a repository, your fork is considered downstream from the original repository.

PRACTICAL 8 : Usage of git stash(for temporary commits)

git stash

the stash's meaning is "**store something safely in a hidden place**." The sense in Git is also the same for stash; Git temporarily saves your data safely without committing.

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 5 (master
)
$ git status
On branch master
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:   document.txt
        modified:   newdoc.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 5 (master
)
$ git stash
Saved working directory and index state WIP on master: 22c6e1b initial commit

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 5 (master
)
$ git status
On branch master
nothing to commit, working tree clean
```

Git stash list

```
$ git stash list
stash@{0}: WIP on master: 22c6e1b initial commit
```

Git stash show

```
$ git stash show
 document.txt | 1 +
 newdoc.txt   | 1 +
 2 files changed, 2 insertions(+)
```

Git stash show -p

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MIN
$ git stash show -p
diff --git a/document.txt b/document.txt
index e69de29..3b2aed8 100644
--- a/document.txt
+++ b/document.txt
@@ -0,0 +1 @@
+this is a new file
diff --git a/newdoc.txt b/newdoc.txt
index e69de29..b6fc4c6 100644
--- a/newdoc.txt
+++ b/newdoc.txt
@@ -0,0 +1 @@
+hello
\ No newline at end of file
```

Git stash pop

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 5 (master
$ git stash pop
error: Your local changes to the following files would be overwritten by merge:
        document.txt
Please commit your changes or stash them before you merge.
Aborting
On branch master
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:   document.txt

no changes added to commit (use "git add" and/or "git commit -a")
The stash entry is kept in case you need it again.
```

Git stash save <message>

```
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directo
        modified:   document.txt
        modified:   newdoc.txt

no changes added to commit (use "git add" and/or "git commit -a")

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/L
$ git stash save "second stash"
Saved working directory and index state On master: second stash

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/L
$ git stash list
stash@{0}: On master: second stash
stash@{1}: WIP on master: 22c6e1b initial commit
```

Git stash drop

```
DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/G
$ git stash drop stash@{1}
Dropped stash@{1} (afc675673c12004429462c8d4a98b117a1d525a6)
```

Git stash clear

```
$ git stash clear

DESKTOP-9I7H4NR+Adm
$ git stash list
```

Rename a file:mv

```
$ git mv newdoc.txt newdoc1.txt

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT
)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        renamed:    newdoc.txt -> newdoc1.txt
```

Move a file to a folder.

```
$ git mv newdoc1.txt folder

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT D
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        renamed:    newdoc.txt -> folder/newdoc1.txt
```

Check HEAD:

```
$ git show HEAD
commit 30222e0014e29abbf096a5f776f6111e0c1201c1 (HEAD -> master)
Author:
Date:   Tue Sep 19 15:56:43 2023 +0530

    second commit

diff --git a/document.txt b/document.txt
index e69de29..ce01362 100644
--- a/document.txt
+++ b/document.txt
@@ -0,0 +1 @@
+hello
```

Git checkout <commit_ID>:

```
$ git checkout 22c6e1b575958b0d5fd831128c66c822b8999d8f
Note: switching to '22c6e1b575958b0d5fd831128c66c822b8999d8f'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 22c6e1b initial commit
A       folder/newdoc1.txt
D       newdoc.txt

DESKTOP-9I7H4NR+Admin@DESKTOP-9I7H4NR MINGW64 /d/GIT DevOps/GITLAB/LAB 5 ((22c
$ git show HEAD
commit 22c6e1b575958b0d5fd831128c66c822b8999d8f (HEAD)
Author:
Date:   Tue Sep 19 15:24:35 2023 +0530

    initial commit

diff --git a/document.txt b/document.txt
new file mode 100644
index 0000000..e69de29
diff --git a/newdoc.txt b/newdoc.txt
new file mode 100644
index 0000000..e69de29
```

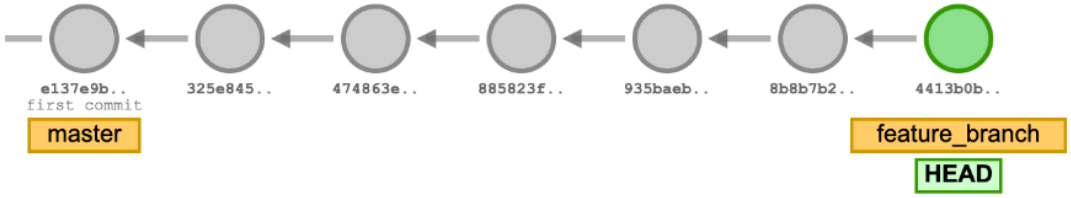
Git checkout master

```
$ git checkout master
Previous HEAD position was 22c6e1b initial commit
Switched to branch 'master'
A       folder/newdoc1.txt
D       newdoc.txt
```

PRACTICAL. 9 :

Head

A git HEAD is simply a pointer that points to a specific version or state of a git repository. It can point to either ***the latest commit***



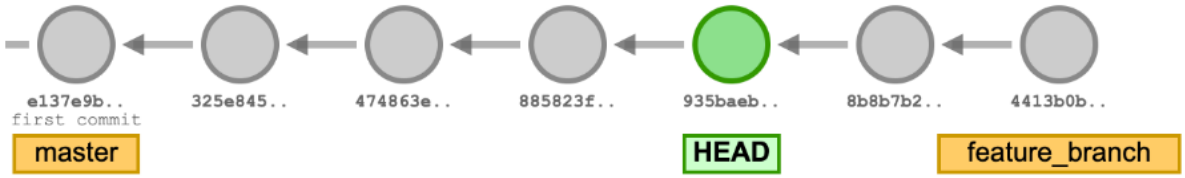

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
$ git show HEAD
commit 7089db0639f907c61e4ae150dcbca8d8f573f4d6 (HEAD -> master)
Author: Rashmi <rc2859@dseu.ac.in>
Date:   Thu Oct 26 07:18:57 2023 +0530

    3rd

diff --git a/e.txt b/e.txt
new file mode 100644
index 0000000..e69de29
```

Detached head

When you are in a *git detached HEAD*

state, you are not on any branch. The HEAD

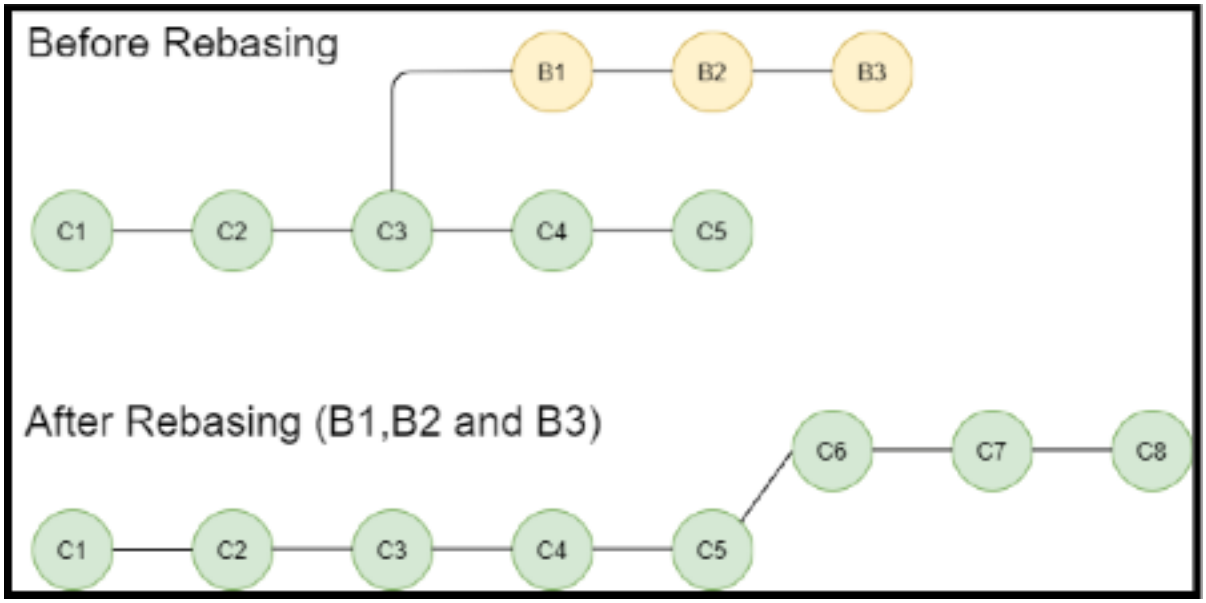
references or points to a commit directly instead of a branch.




```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
$ git checkout d786ef3
Note: switching to 'd786ef3'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at d786ef3 2nd
```

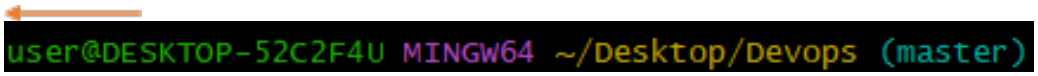
PRACTICAL 10 :

- Rebasing in Git is a process of integrating a series of commits on top of another base tip. It takes all the commits of a branch and appends them to the commits of a new branch.



- git reset –hard
- **Suppose there are two branches master and New. Step1: In master branch, Commit some files.**
-

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ touch a.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git add .

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git commit -m
error: switch `m' requires a value

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git commit -m "1st"
[master cbed5ad] 1st
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 a.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git log --oneline
cbed5ad (HEAD -> master) 1st
92996f1 (origin/master, origin/HEAD) initial commit

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
```

- 1st Commit (Master branch)
- 
```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
```
- **Step2: Checkout to New branch and add some files.**
- **Step 3: Now checkout to master branch again, and add some files there too.**

```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git checkout new
Switched to branch 'new'

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ touch b.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git add .

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git commit -m "2nd"
[new c50964c] 2nd
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 b.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ touch c.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git add .

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git commit -m "3rd"
[new 8f79b03] 3rd
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 c.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git log --oneline
8f79b03 (HEAD -> new) 3rd
c50964c 2nd
cbed5ad (master) 1st
92996f1 (origin/master, origin/HEAD) initial commit
```

-
```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
```
- 2nd Commit (New branch)
- 4th Commit (Master branch)
- 3rd Commit (New branch)
- 
```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
```
- **Step4: By using rebase command, we will combine the commits.**

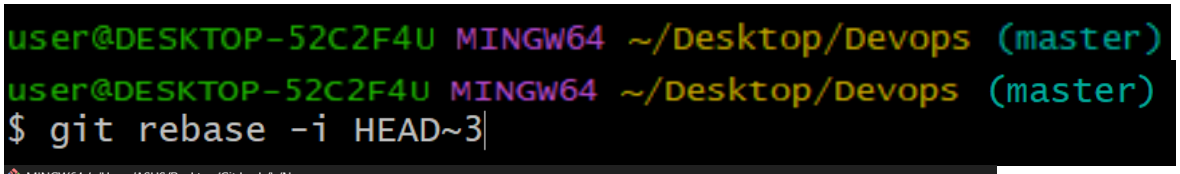
```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git rebase new master
Successfully rebased and updated refs/heads/master.

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops
$ git log --oneline
1a06f23 (HEAD -> master) 4th
8f79b03 (new) 3rd
c50964c 2nd
cbed5ad 1st
92996f1 (origin/master, origin/HEAD) initial commit
```

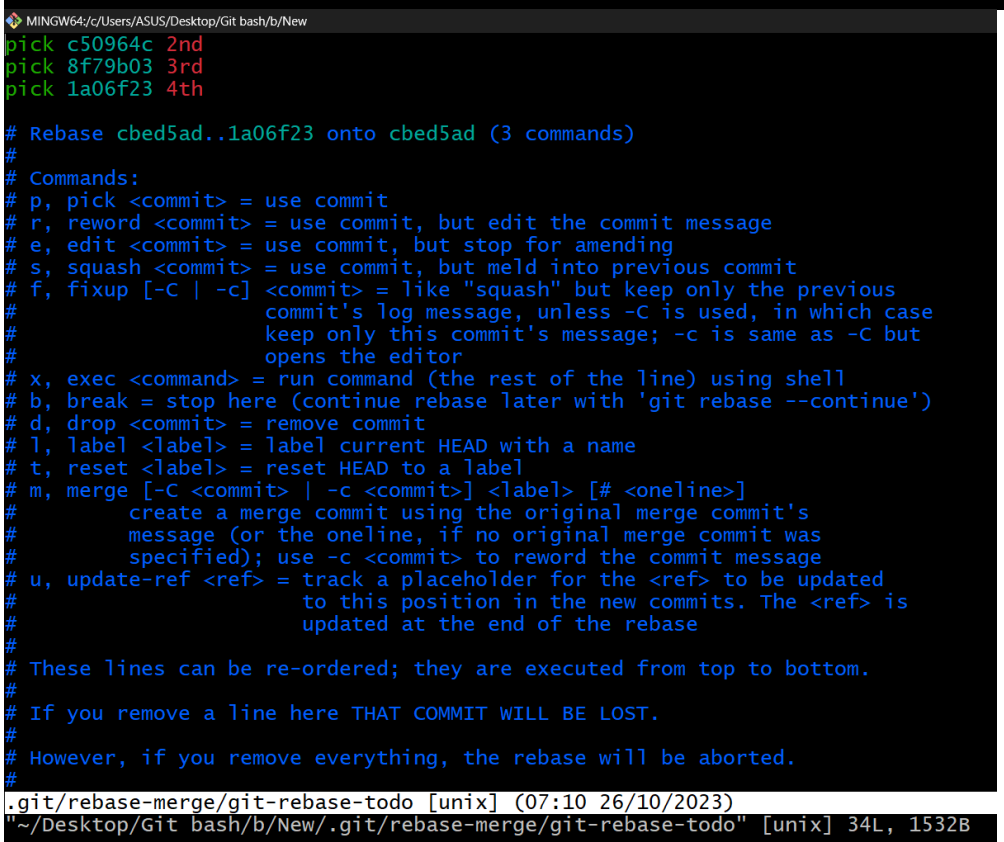
-
-

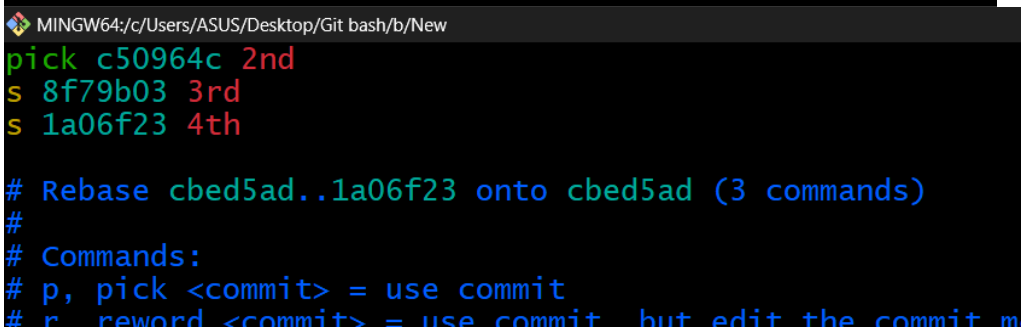
```
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops )
$ touch d.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops )
$ git add .

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops )
$ git commit -m "4th"
[master 8e391cf] 4th
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 d.txt

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops )
$ git log --oneline
8e391cf (HEAD -> master) 4th
cbed5ad 1st
92996f1 (origin/master, origin/HEAD) initial commit
```

Git squash

- To "squash" in Git means to combine multiple commits into one.
- **By using this command, an editor will open**
- **In this editor, we will replace the pick command with squash command.**







We will save these changes.

```
MINGW64:/c/Users/ASUS/Desktop/Git bash/b/New
# This is a combination of 3 commits.
# This is the 1st commit message:

2nd

# This is the commit message #2:

3rd

# This is the commit message #3:

4th

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
```

- **Next, we removed the commit messages from the editor.**

```
MINGW64:/c/Users/ASUS/Desktop/Git bash/b/New
# This is a combination of 3 commits.
# This is the 1st commit message:

2nd

# This is the commit message #2:



# This is the commit message #3:



# Please enter the commit message for your
# with '#' will be ignored, and an empty mes
#
# Date:      Wed Oct 25 22:09:08 2023 +0530
#
# interactive rebase in progress; onto cbed
# Last commands done (3 commands done):
```

We will save these changes and exit from editor.

```
[detached HEAD d786ef3] 2nd
 Date: Wed Oct 25 22:09:08 2023 +0530
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 b.txt
 create mode 100644 c.txt
 create mode 100644 d.txt
Successfully rebased and updated refs/heads/master.

user@DESKTOP-52C2F4U MINGW64 ~/Desktop/Devops (master)
$ git log --oneline
d786ef3 (HEAD -> master) 2nd
cbed5ad 1st
92996f1 (origin/master, origin/HEAD) initial commit
```

Here we successfully used the squash command to combine commits.

Practical 11: Background study, Installation and configuration of Jenkins

Jenkins is an open source automation tool written in Java programming language that allows continuous integration.

Jenkins **builds** and **tests** our software projects which continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build.

It also allows us to continuously **deliver** our software by integrating with a large number of testing and deployment technologies.

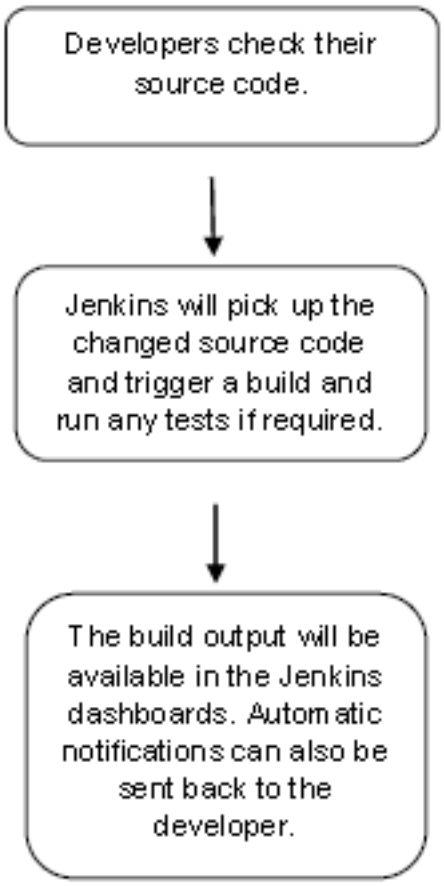
Jenkins offers a straightforward way to set up a continuous integration or continuous delivery environment for almost any combination of languages and source code repositories using pipelines, as well as automating other routine development tasks.

With the help of Jenkins, organizations can speed up the software development process through automation. Jenkins adds development life-cycle processes of all kinds, including build, document, test, package, stage, deploy static analysis and much more.

Jenkins achieves CI (Continuous Integration) with the help of plugins. Plugins is used to allow the integration of various DevOps stages. If you want to integrate a particular tool, you have to install the plugins for that tool. For example: Maven 2 Project, Git, HTML Publisher, Amazon EC2, etc.

For example: If any organization is developing a project, then **Jenkins** will continuously test your project builds and show you the errors in early stages of your development.

Work Flow:



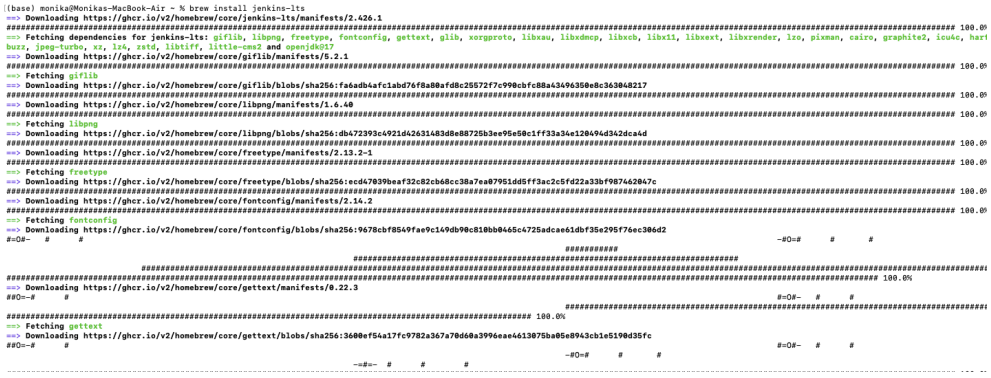
INSTALLATION OF JENKING:

1. Install Java Development Kit (JDK)

- Download JDK 8 and choose windows 32-bit or 64-bit according to your system configuration. Click on "accept the license agreement."

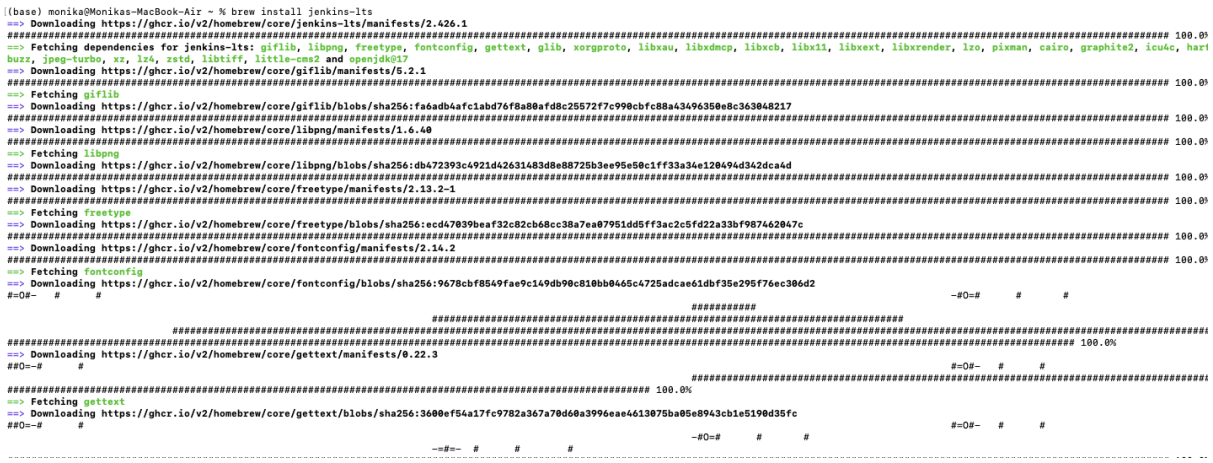
2. Set the Path for the Environmental Variable for JDK

- Go to System Properties. Under the "Advanced" tab, select "Environment Variables."
- Under system variables, select "new." Then copy the path of the JDK folder and paste it in the corresponding value field. Similarly, do this for JRE.
- Under system variables, set up a bin folder for JDK in PATH variables.
- Go to command prompt and type the following to check if Java has been successfully installed



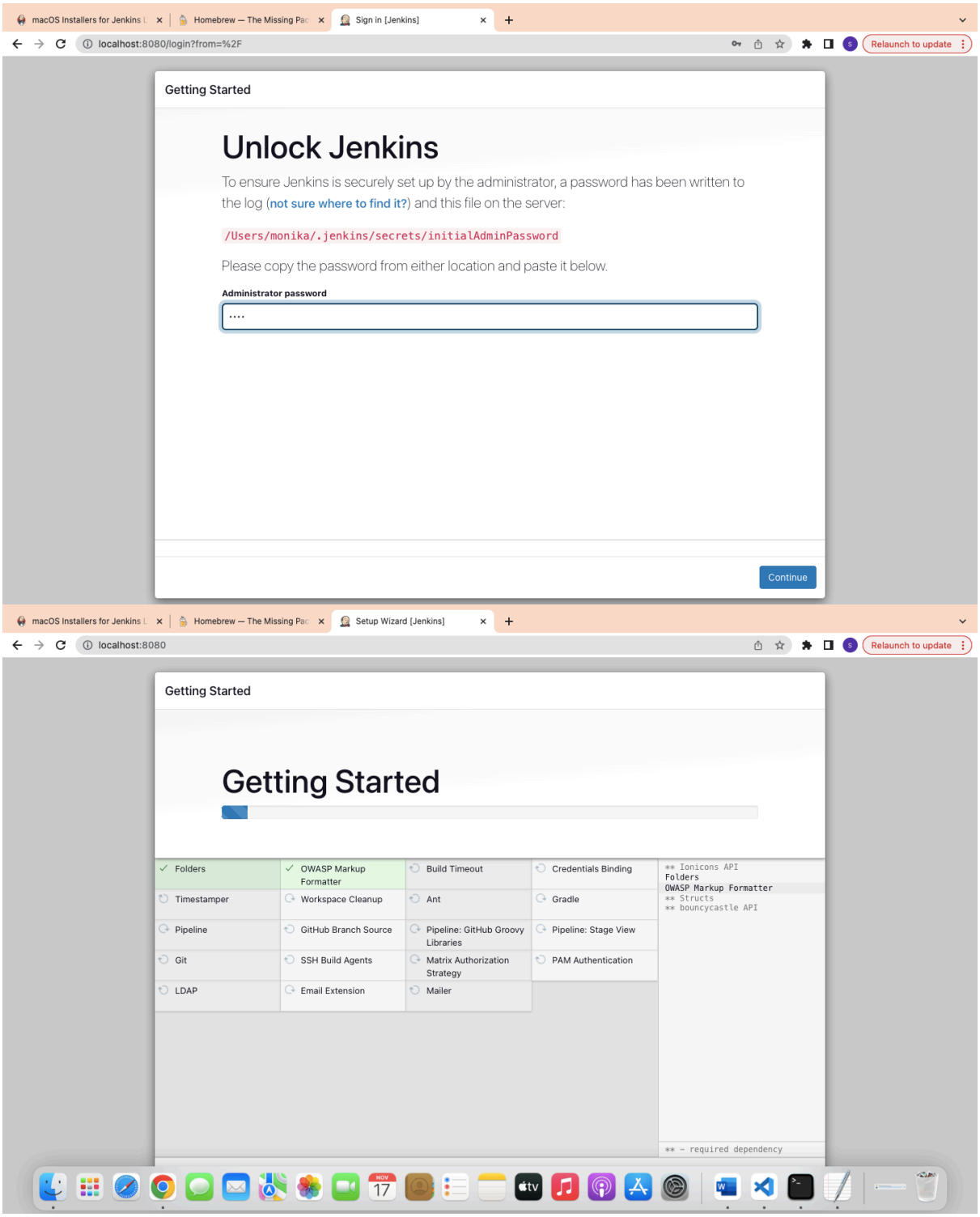
3. Download and Install Jenkins

- Download Jenkins. Under LTS, click on windows.
- After the file is downloaded, unzip it. Click on the folder and install it. Select "finish" once done.



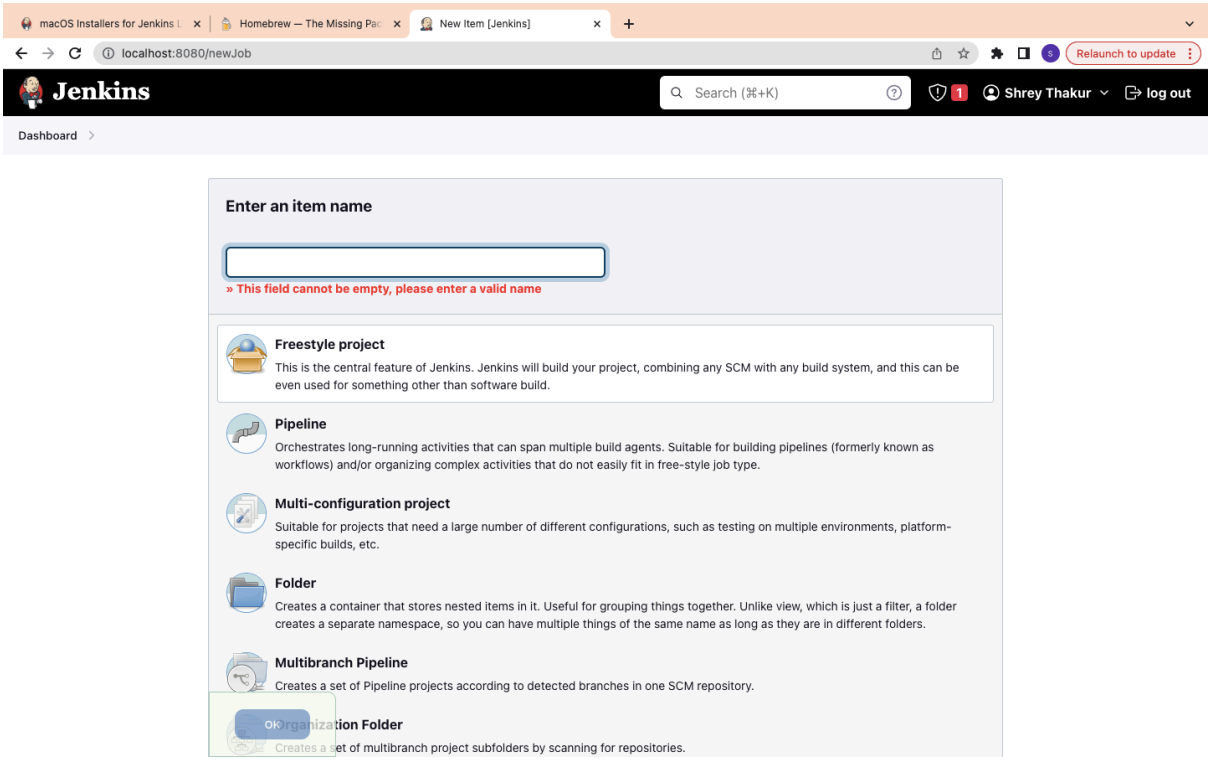
4. Run Jenkins on Localhost 8080

- Once Jenkins is installed, explore it. Open the web browser and type "localhost:8080".
- Enter the credentials and log in. If you install Jenkins for the first time, the dashboard will ask you to install the recommended plugins. Install all the recommended plugins.


macOS Installers for Jenkins L
Homebrew — The Missing Pac
Sign in [Jenkins]
localhost:8080/login?from=%2F
Relaunch to update
Getting Started
Unlock Jenkins
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:
/Users/monika/.jenkins/secrets/initialAdminPassword
Please copy the password from either location and paste it below.
Administrator password
....
Continue
macOS Installers for Jenkins L
Homebrew — The Missing Pac
Setup Wizard [Jenkins]
localhost:8080
Relaunch to update
Getting Started
Getting Started
Folders
OWASP Markup Formatter
Build Timeout
Credentials Binding
Timestamper
Workspace Cleanup
Ant
Gradle
Pipeline
GitHub Branch Source
Pipeline: GitHub Groovy Libraries
Pipeline: Stage View
Git
SSH Build Agents
Matrix Authorization Strategy
PAM Authentication
LDAP
Email Extension
Mailer
** Ionicons API
Folders
OWASP Markup Formatter
** Structs
** bouncycastle API
** - required dependency

5. Jenkins Server Interface

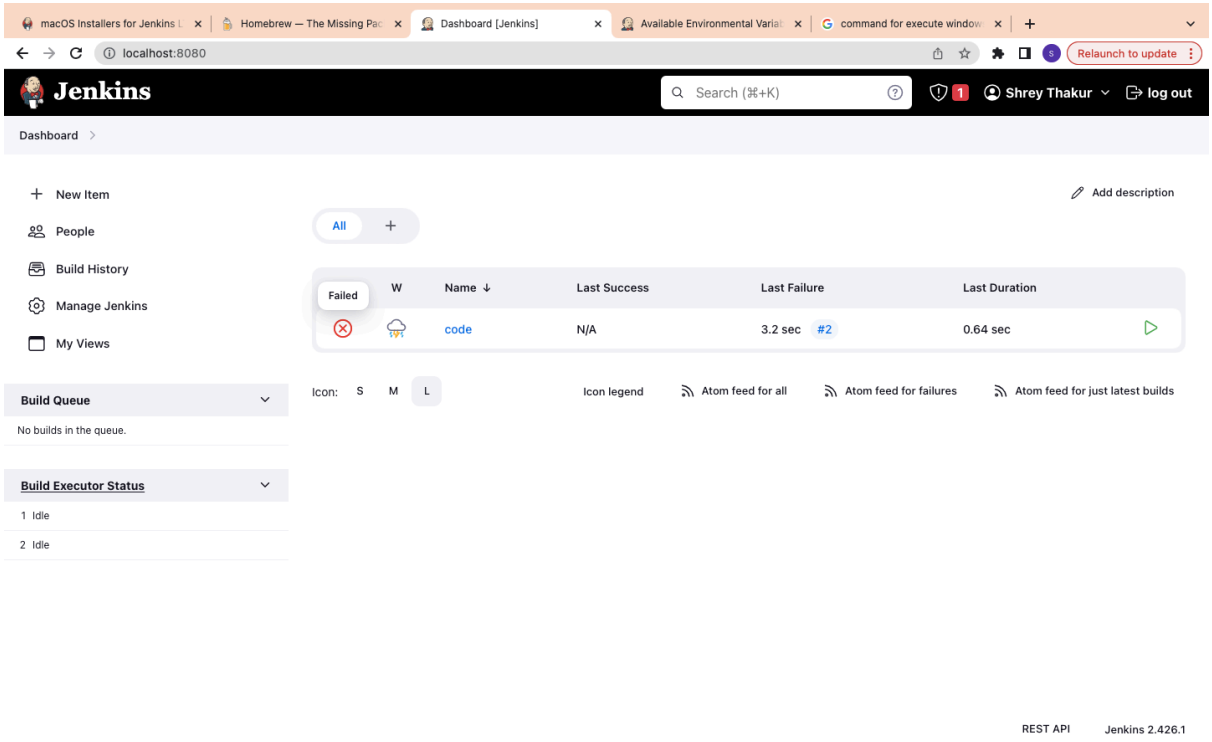
- New Item allows you to create a new project.
- Build History shows the status of your builds.
- Manage System deals with the various configurations of the system.



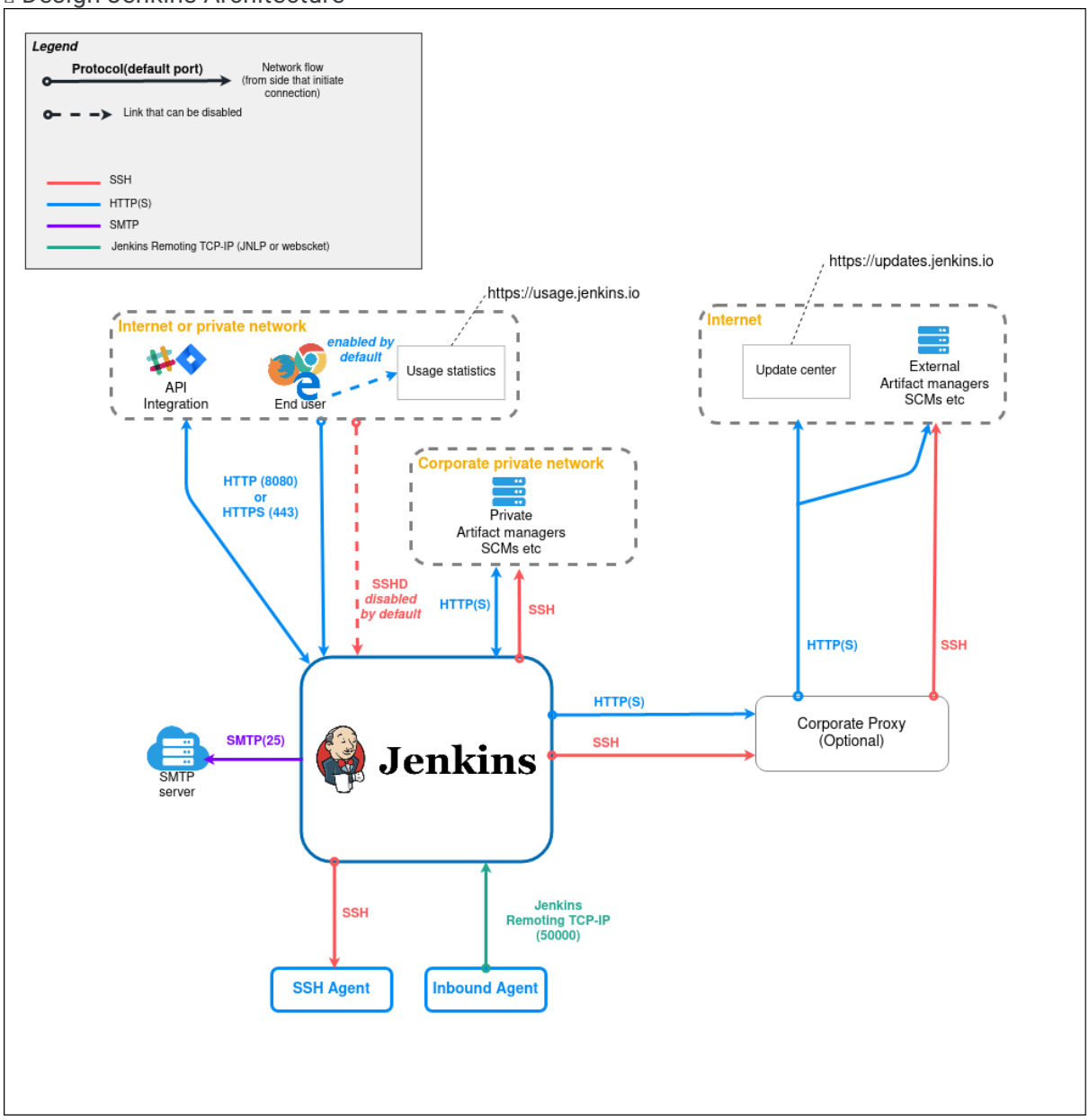
6. Build and Run a Job on Jenkins

- Select a new item (Name - Jenkins_demo). Choose a freestyle project and click Ok.
- Under the General tab, give a description like "This is my first Jenkins job." Under the "Build Triggers" tab, select add built step and then click on the "Execute Windows" batch command.
- In the command box, type the following: echo "Hello... This is my first Jenkins Demo: %date%: %time% ". Click on apply and then save.

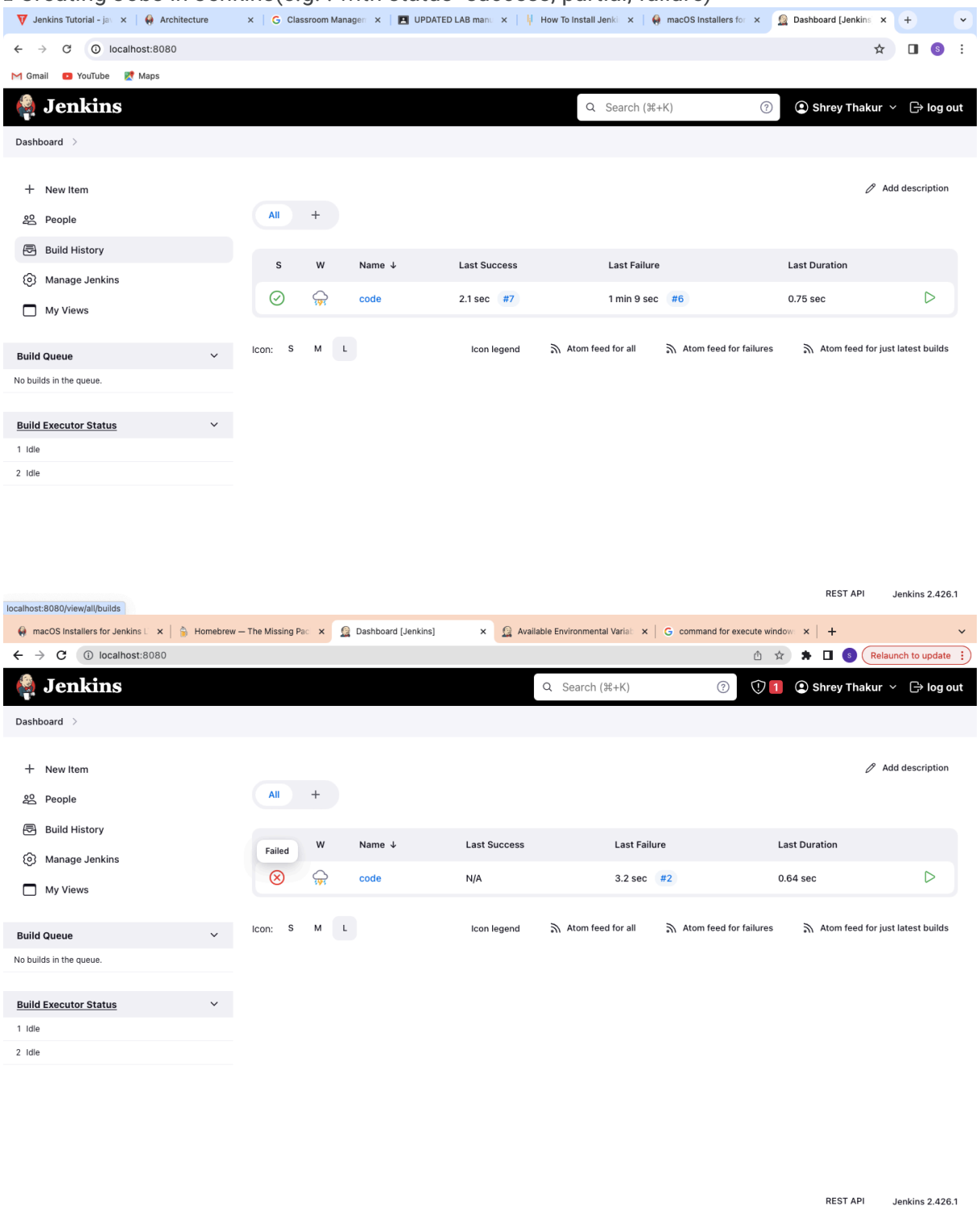
- Select build now. You can see a building history has been created. Click on that. In the console output, you can see the output of the first Jenkins job with time and date



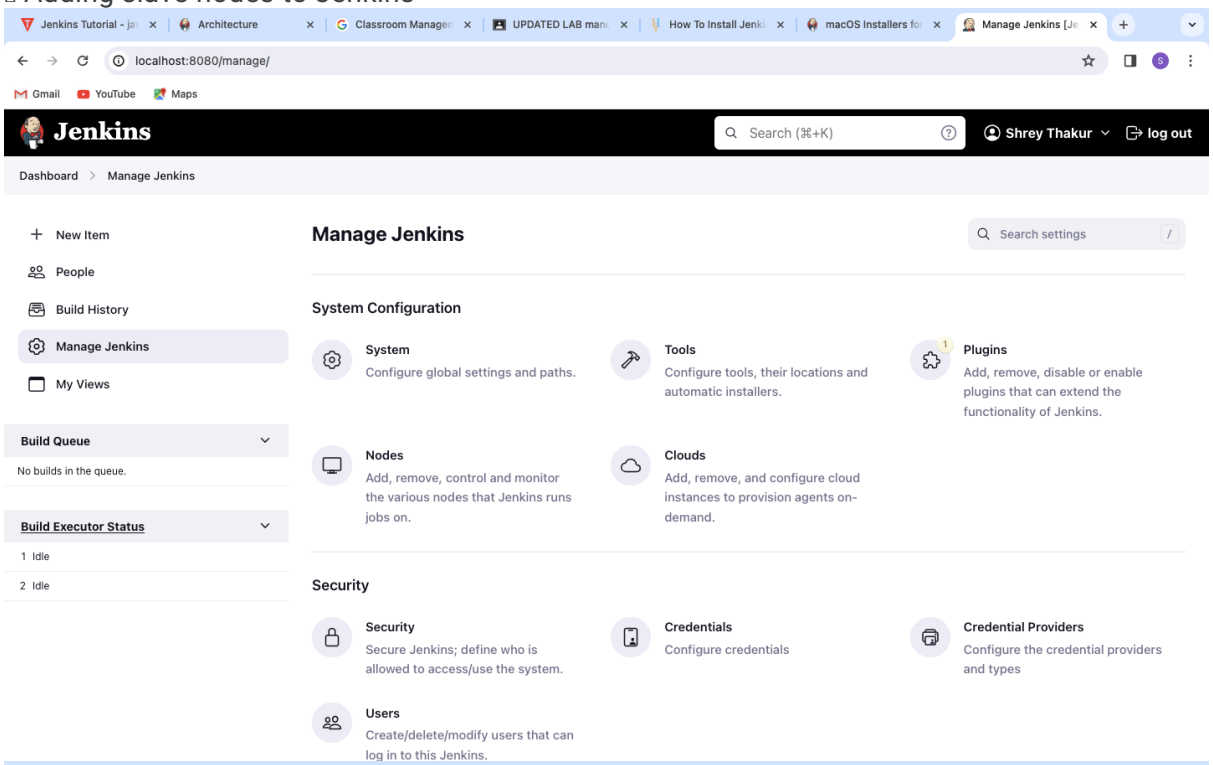
PRACTICAL 12 : Perform following in Jenkins(with snapshots of each step):-
 Design Jenkins Architecture

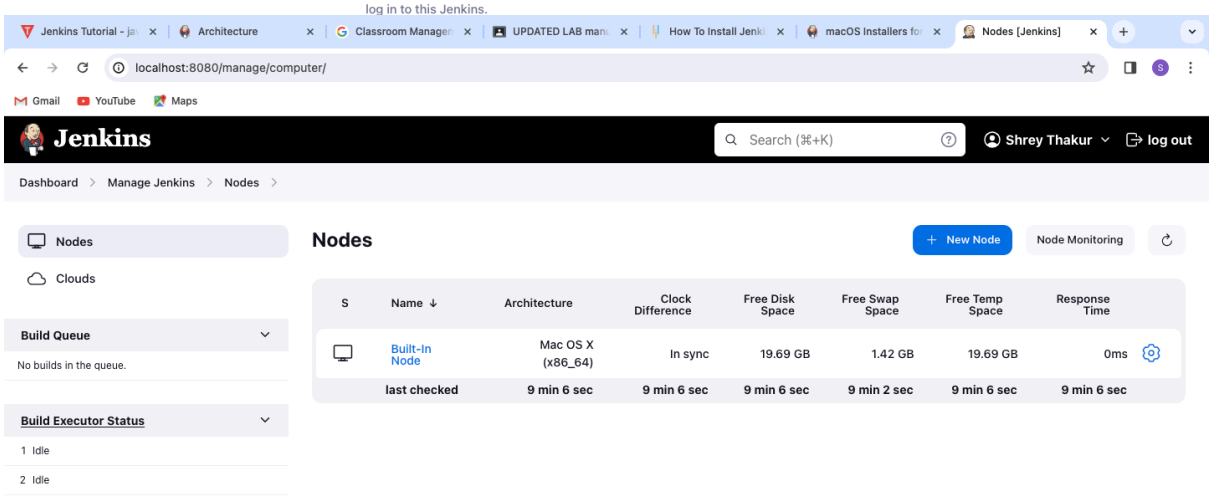


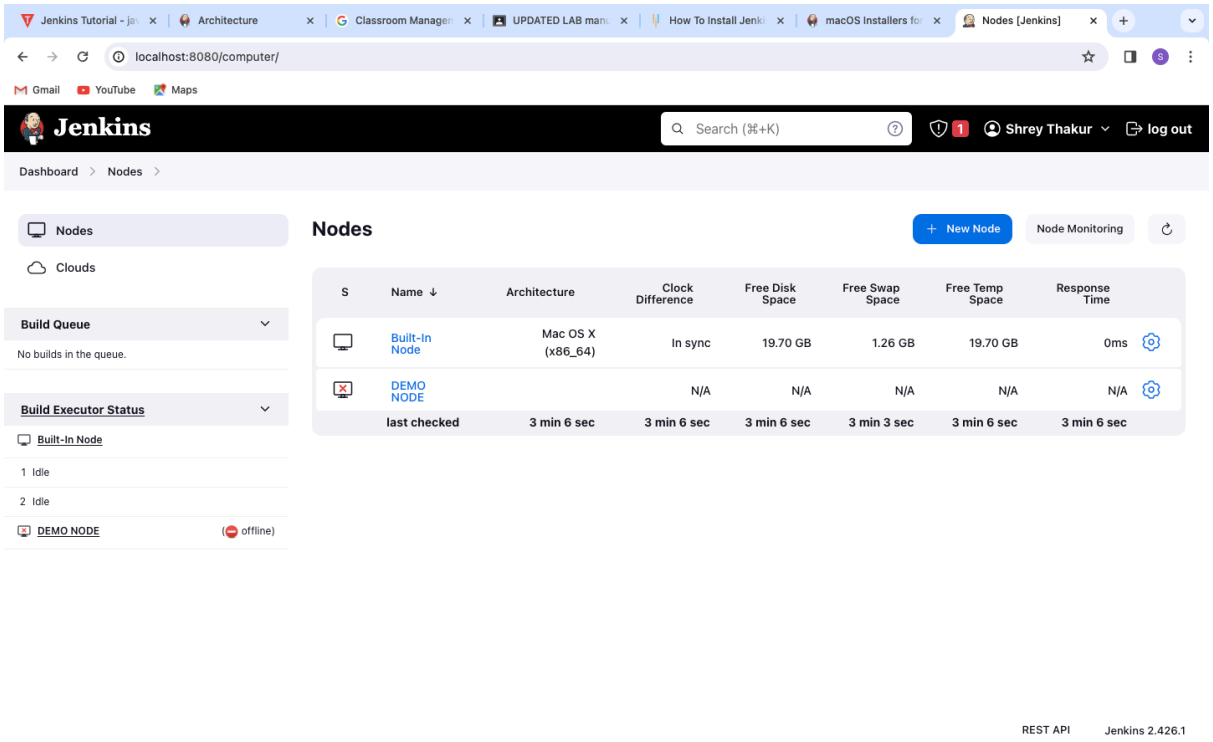
Creating Jobs in Jenkins(e.g. : with status- success, partial, failure)



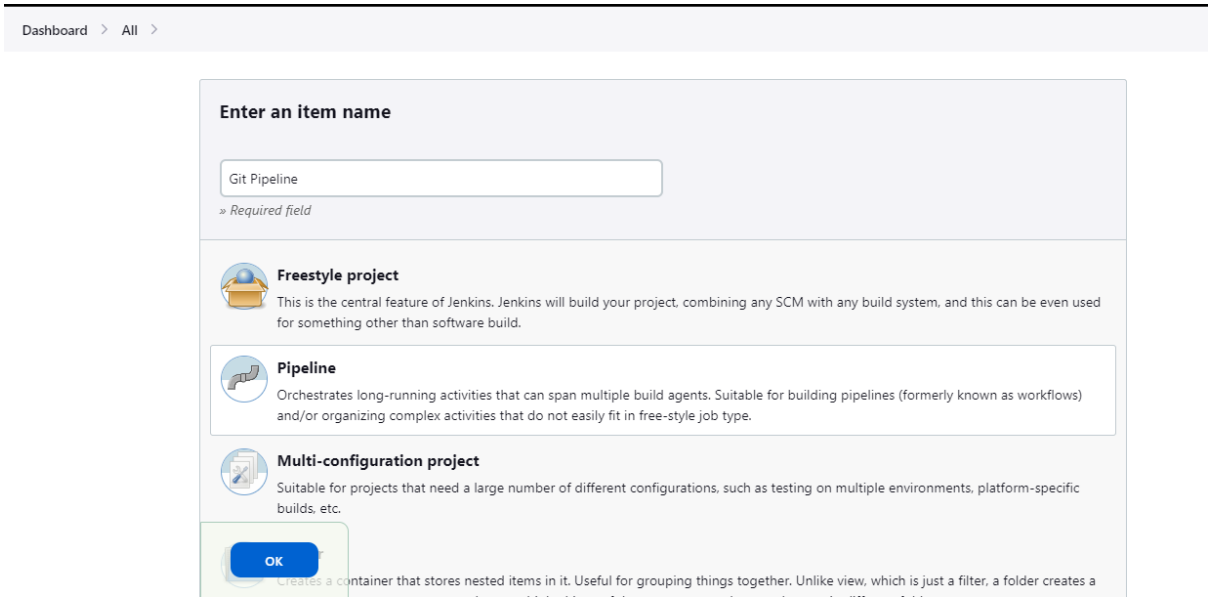
Adding slave nodes to Jenkins







Fetching content from github in Jenkins



SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/TanyaMehrolia/devops.git

Credentials ?

- none -

Add

Advanced

Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/master

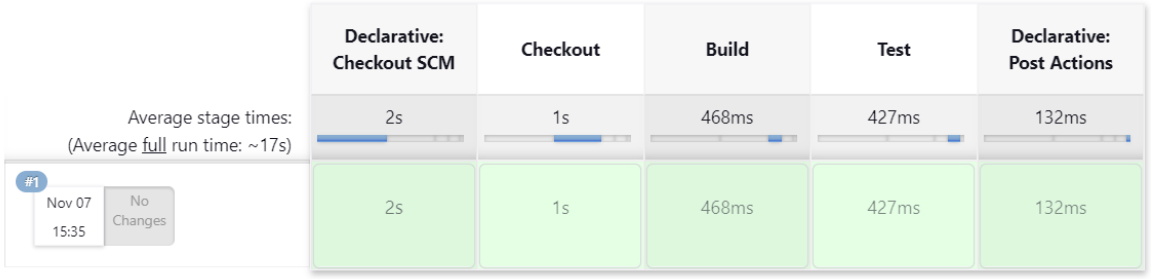
Code | Blame | 36 lines (33 loc) · 1.11 KB | Code 55% faster with GitHub Copilot

```
pipeline {
    agent any
    stages {
        stage('Checkout') {
            steps {
                // This stage checks out your source code from the Git repository.
                checkout([$class: 'GitSCM', branches: [[name: '*/master']], userRemoteConfigs: [[url: 'https://github.com/TanyaMehrolia/devops.git']]])
            }
        }

        stage('Build') {
            steps {
                // This stage simulates a build step. You should replace this with your actual build commands.
                bat 'echo "Building..."'
            }
        }

        stage('Test') {
            steps {
                // This stage simulates a test step. You should replace this with your actual test commands.
                bat 'echo "Testing..."'
            }
        }
```

Name	Last commit message
apnacollege-demo	HII
devops	HII
.gitignore	files added
Jenkinsfile	Create Jenkinsfile
file1.txt	HII
file2.txt	HII
sample.txt	2nd Commit!
tanya.txt	Add files via upload

Pipeline Git pipeline

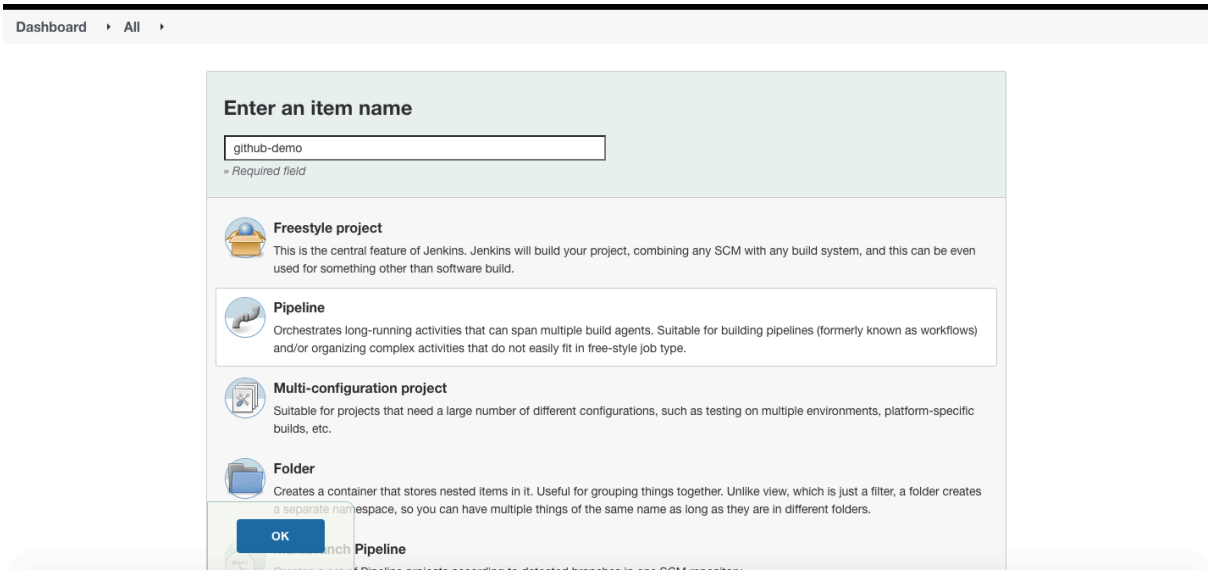
Stage View



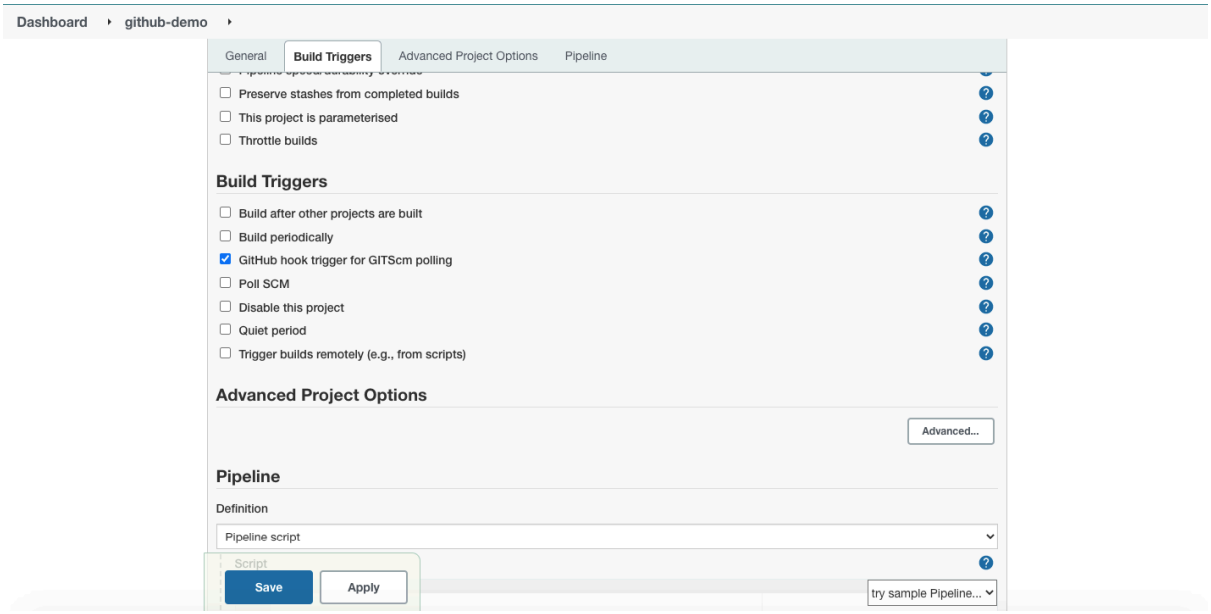
	Declarative: Checkout SCM	Checkout	Build	Test	Declarative: Post Actions
Average stage times: (Average full run time: ~17s)	2s	1s	468ms	427ms	132ms
#1 Nov 07 15:35 No Changes	2s	1s	468ms	427ms	132ms

 Build the pipeline of fetched jobs from github in/using Jenkins

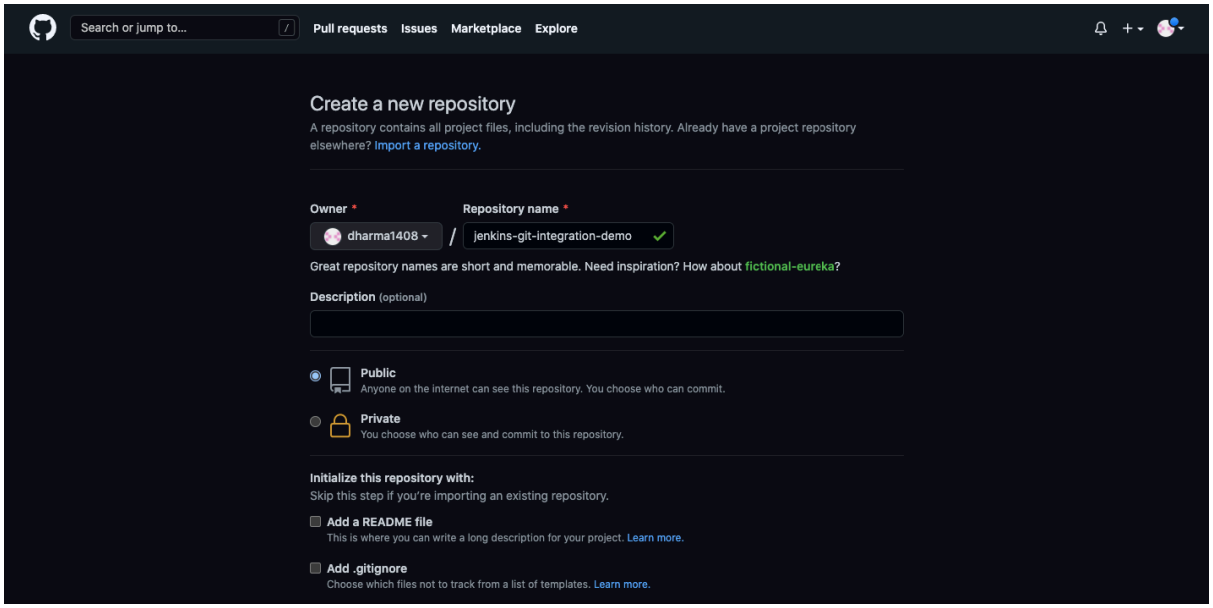
Step 1: Create a Jenkins Job



Step 2: Configure GitHub Hook Trigger

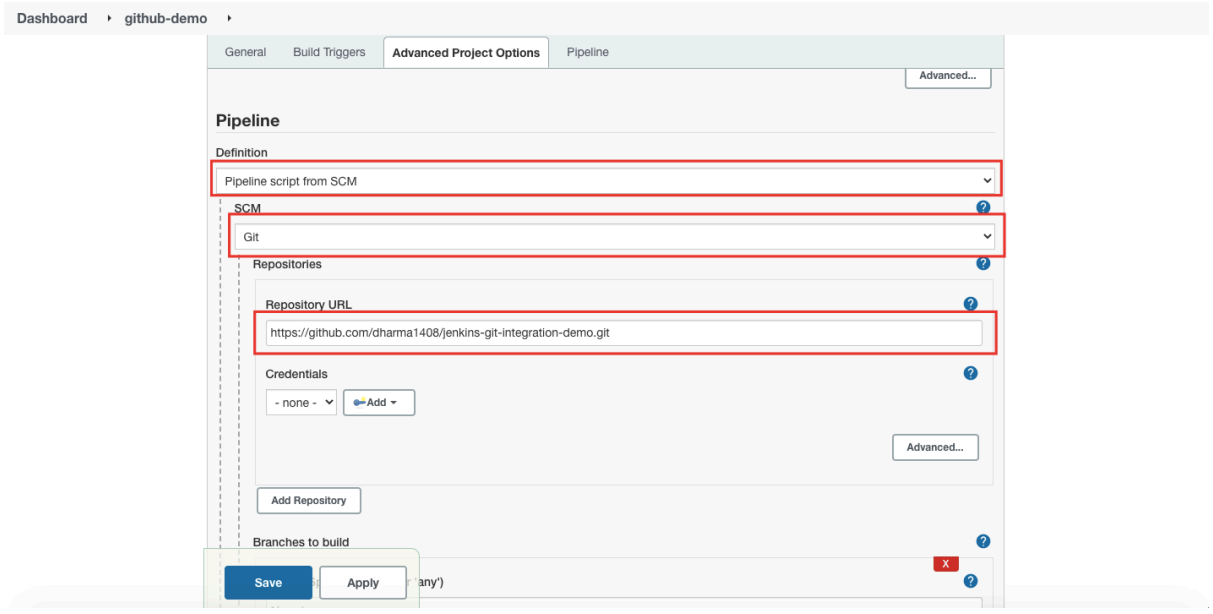


Step 3: Create a GitHub Repository

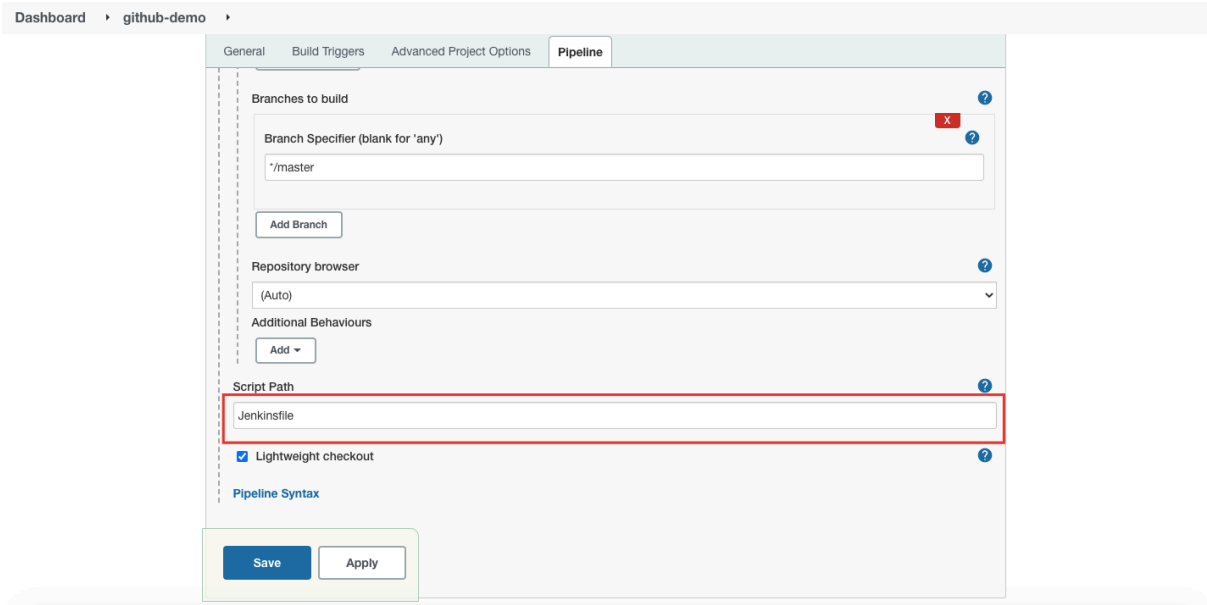


Step 4: Connect a GitHub Repository

Scroll down to the "Pipeline" section. We will configure Jenkins to use GitHub repository as source. Select "Pipeline Script from SCM" under the definition option.

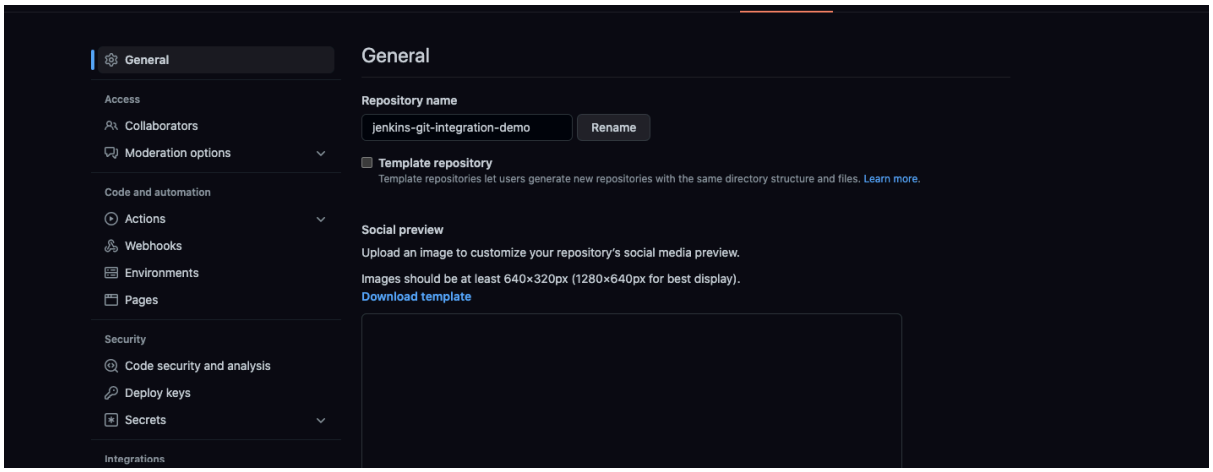


Type the Jenkinsfile path and click on save.

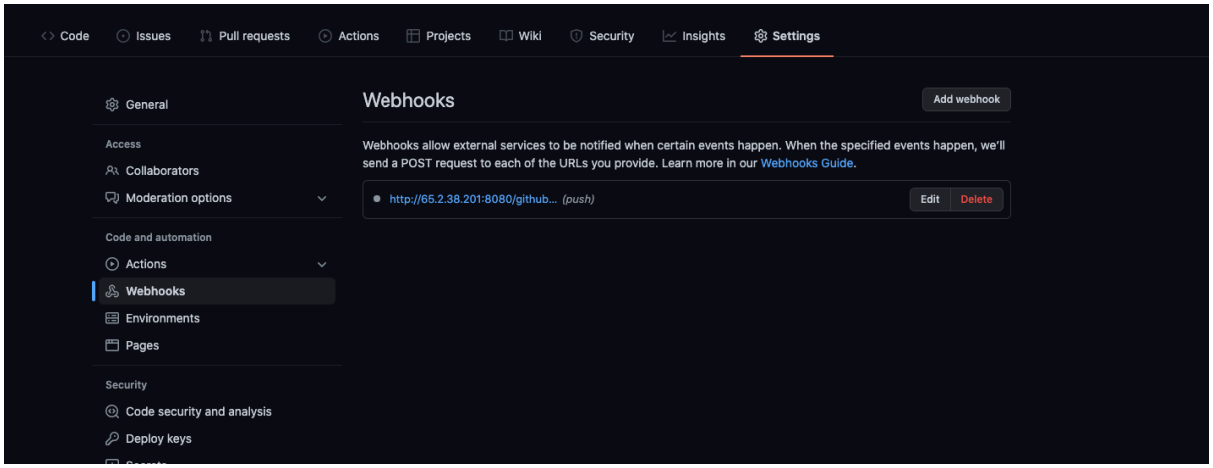


Step 5: Adding a WebHook in GitHub

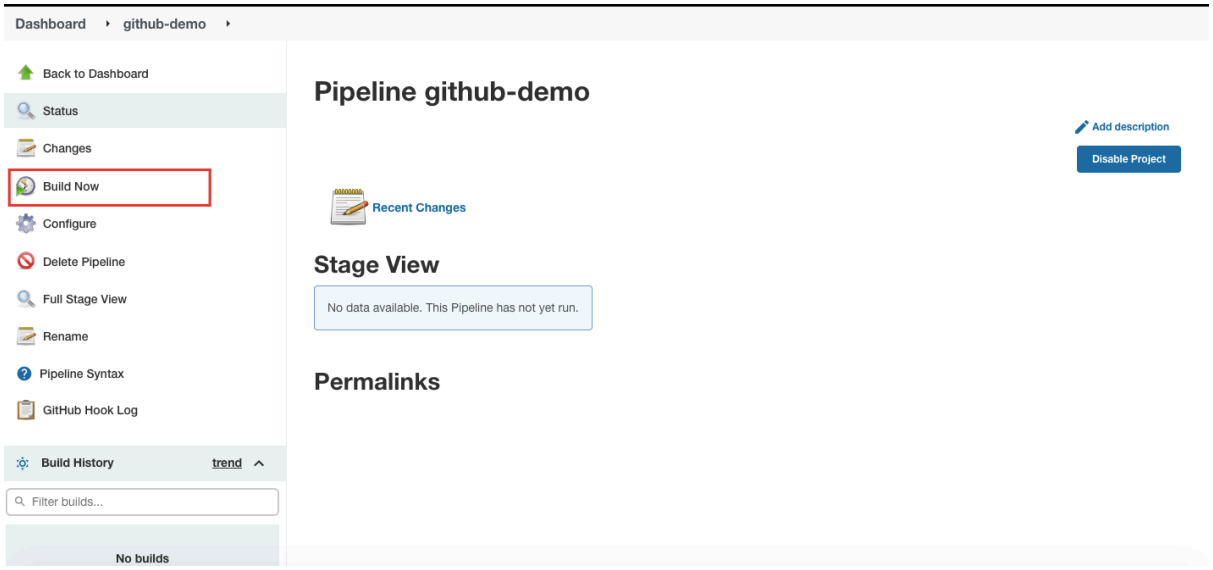
Login to GitHub and go to the repository which you have connected in the previous step.



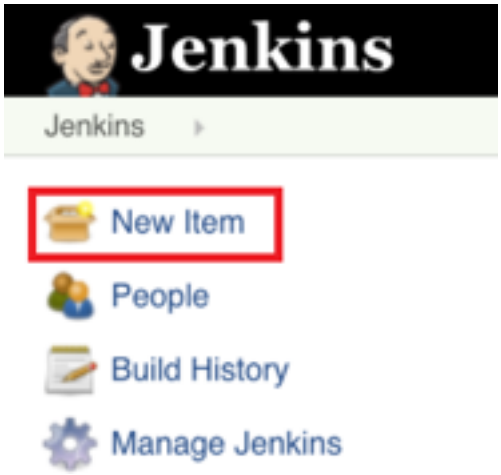
Click on “WebHooks” and then click on “Add WebHooks”.

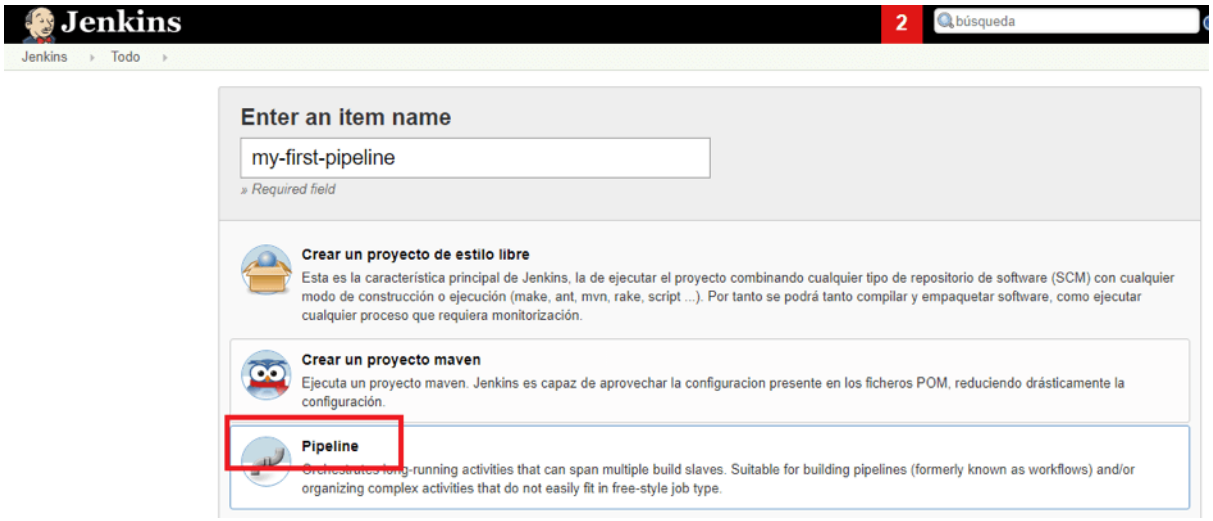


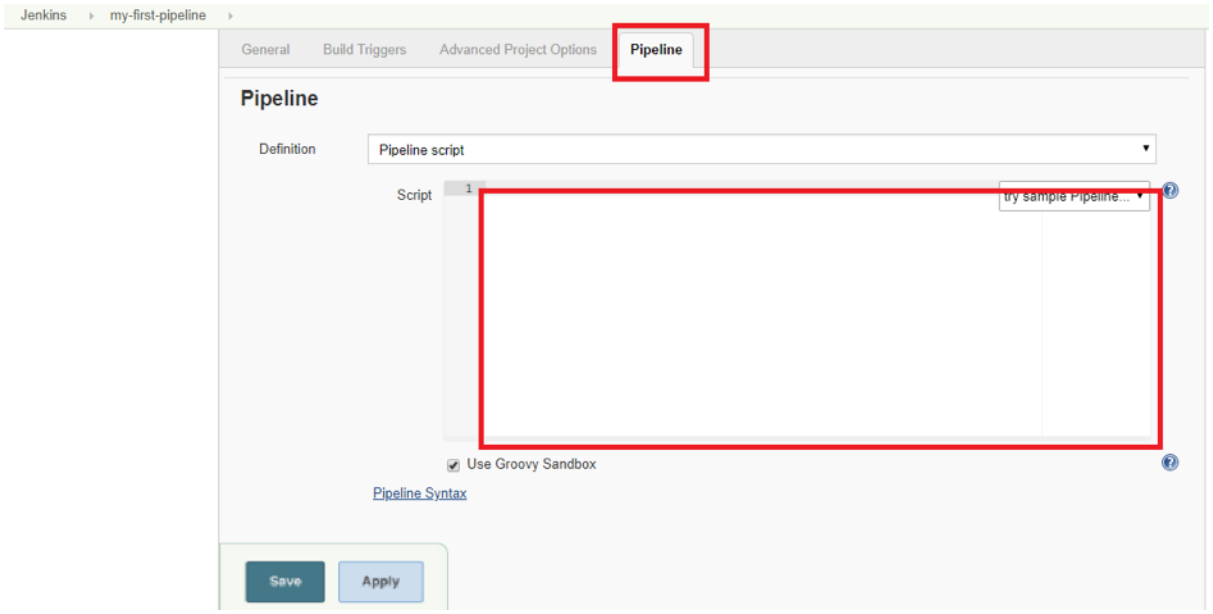
Step 6: Manually Building Jenkins Job



 Build a pipeline using script in Jenkins







Creating Your Jenkins Pipeline Script

.

Node Blocks

The first block to be defined is the “node”:

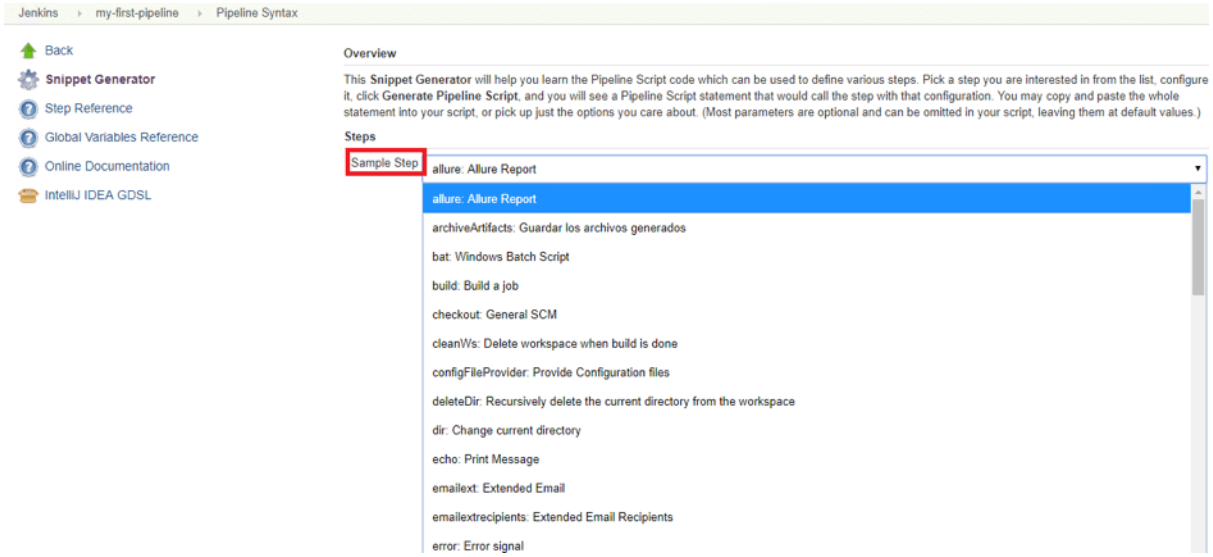
```
node {
}
```

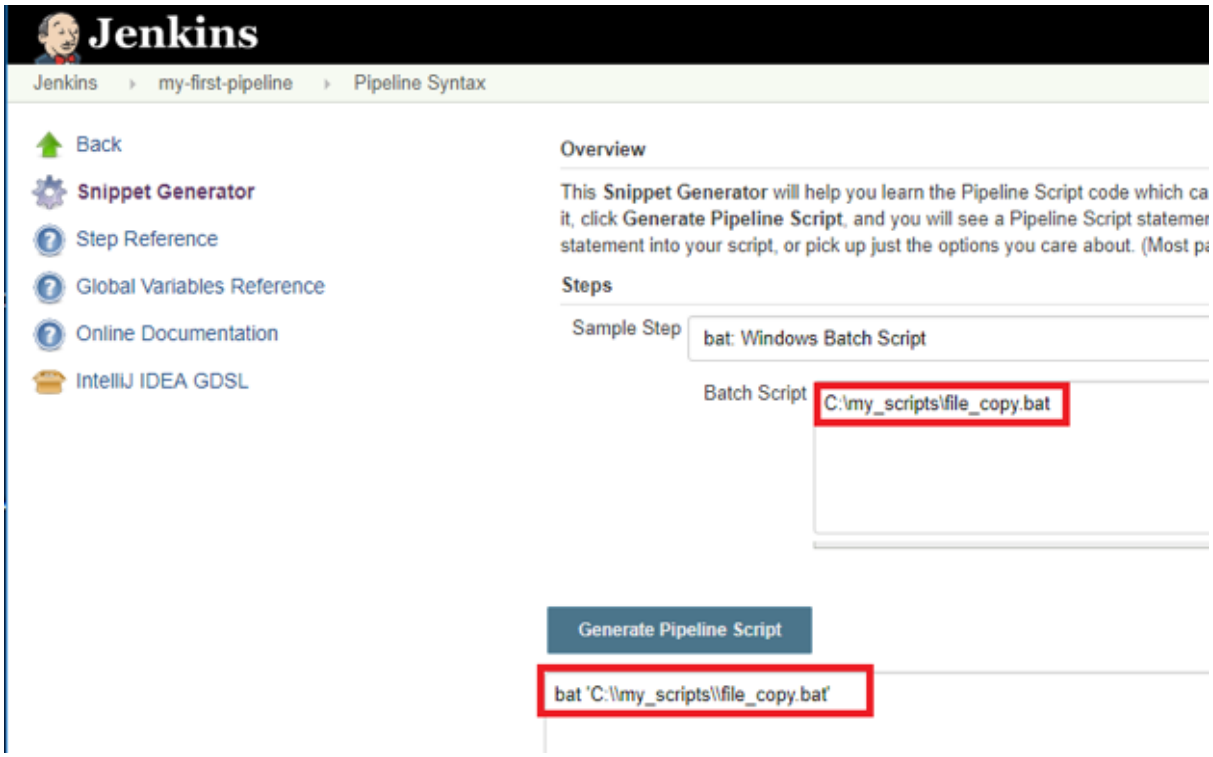
Stage Blocks

The next required section is the “stage”:

```
stage {
}
```

```
node {
              stage (‘Build’) {
                     bat "msbuild
${C:\\Jenkins\\my_project\\workspace\\test\\my_project.sln}"
              }

stage('Selenium tests') {
dir(automation_path) {  //changes the path to “automation_path”
bat "mvn clean test -Dsuite=SMOKE_TEST -Denvironment=QA"
}
}
       }
```


Jenkins › my-first-pipeline › Pipeline Syntax
Back
Snippet Generator
Step Reference
Global Variables Reference
Online Documentation
IntelliJ IDEA GDSL
Overview
This Snippet Generator will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click Generate Pipeline Script, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your script, or pick up just the options you care about. (Most parameters are optional and can be omitted in your script, leaving them at default values.)
Steps
Sample Step
allure: Allure Report
allure: Allure Report
archiveArtifacts: Guardar los archivos generados
bat: Windows Batch Script
build: Build a job
checkout: General SCM
cleanWs: Delete workspace when build is done
configFileProvider: Provide Configuration files
deleteDir: Recursively delete the current directory from the workspace
dir: Change current directory
echo: Print Message
emailext: Extended Email
emailextrecipients: Extended Email Recipients
error: Error signal


Jenkins
Jenkins › my-first-pipeline › Pipeline Syntax
Back
Snippet Generator
Step Reference
Global Variables Reference
Online Documentation
IntelliJ IDEA GDSL
Overview
This Snippet Generator will help you learn the Pipeline Script code which ca
it, click Generate Pipeline Script, and you will see a Pipeline Script statemer
statement into your script, or pick up just the options you care about. (Most pa
Steps
Sample Step
bat: Windows Batch Script
Batch Script
C:\my_scripts\file_copy.bat
Generate Pipeline Script
bat 'C:\\my_scripts\\file_copy.bat'

PRACTICAL 13 : Background study, Installation and configuration of Docker on windows/ Linux platform /using play with docker platform

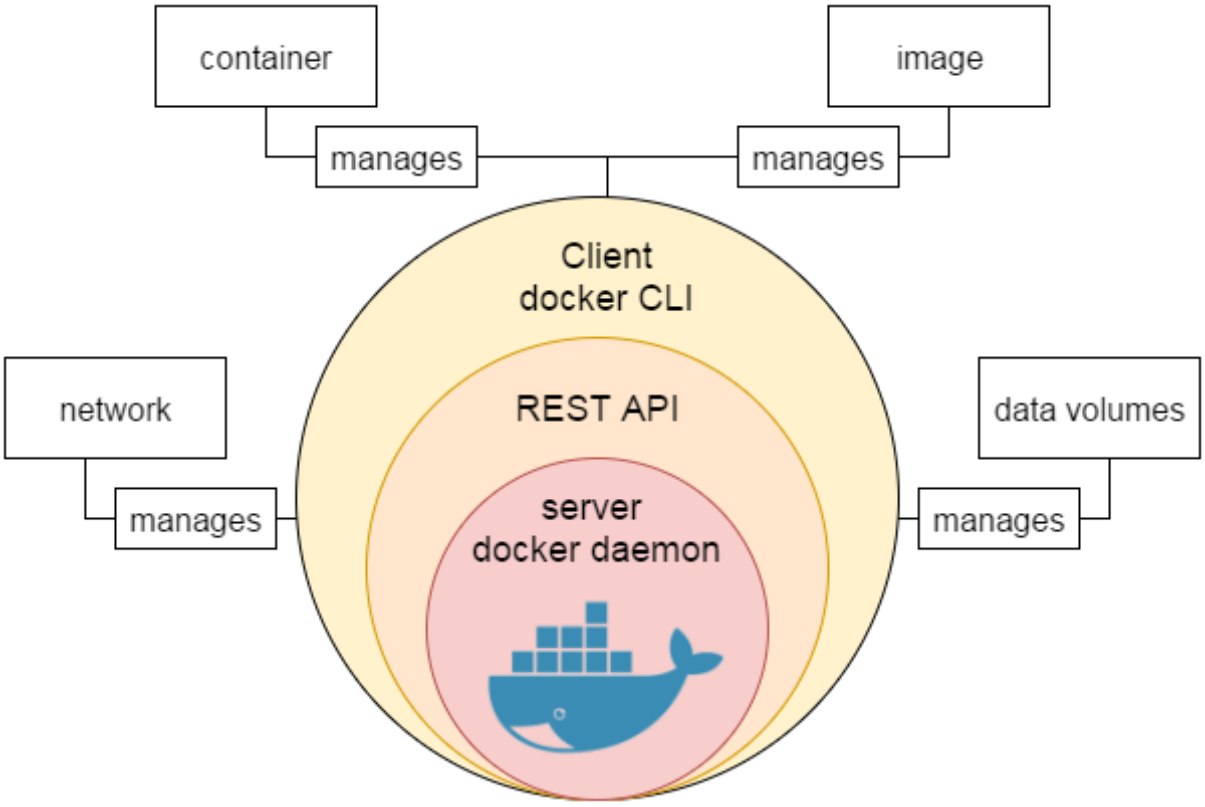
Docker is an **open-source centralized platform designed** to create, deploy, and run applications. Docker uses **container** on the host's operating system to run applications. It allows applications to use the same **Linux kernel** as a system on the host computer, rather than creating a whole virtual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

Docker includes components such as **Docker client, Docker server, Docker machine, Docker hub, Docker composes,** etc.

Let's understand the Docker containers and virtual machine.

Docker Containers

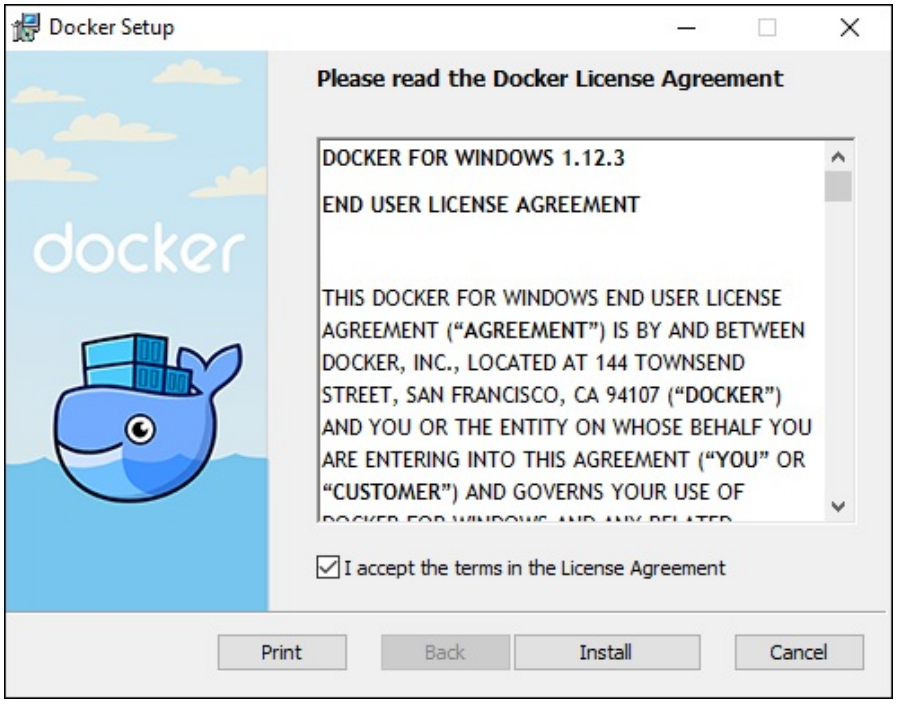
Docker containers are the **lightweight** alternatives of the virtual machine. It allows developers to package up the application with all its libraries and dependencies, and ship it as a single package. The advantage of using a docker container is that you don't need to allocate any RAM and disk space for the applications. It automatically generates storage and space according to the application requirement.



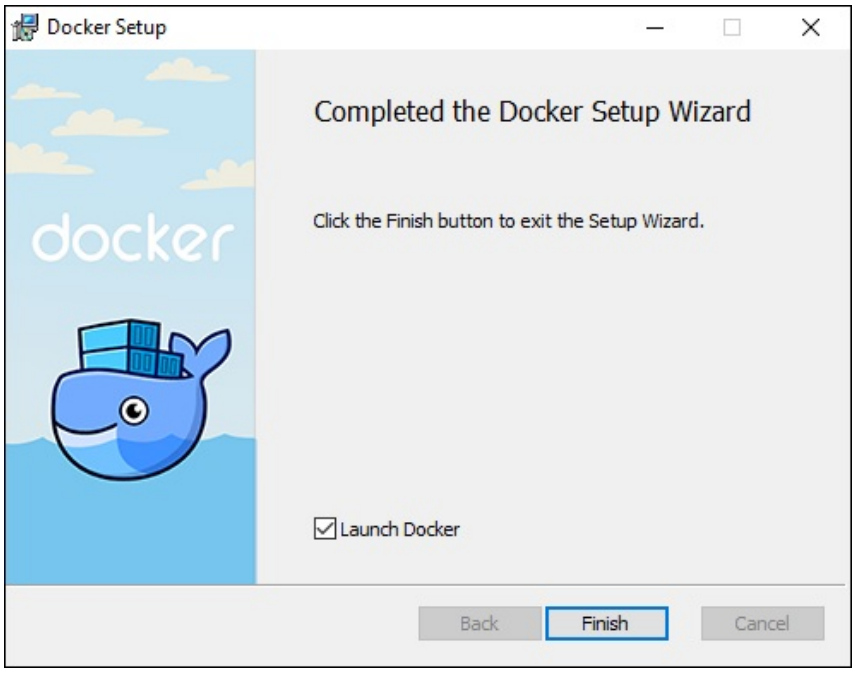
Docker for Windows

Once the installer has been downloaded, double-click it to start the installer and then follow the steps given below.

Step 1 – Click on the Agreement terms and then the Install button to proceed ahead with the installation.



Step 2 – Once complete, click the Finish button to complete the installation.

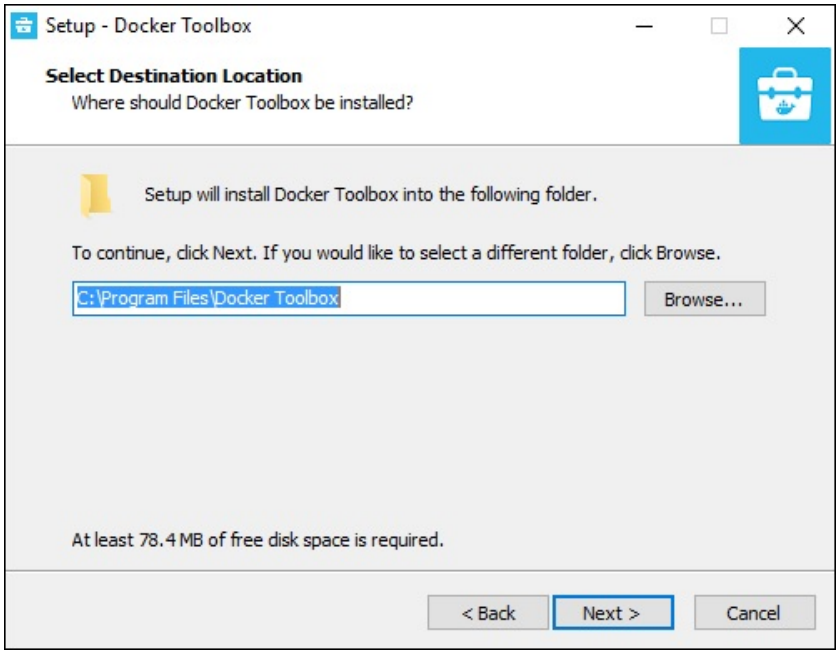


Docker ToolBox

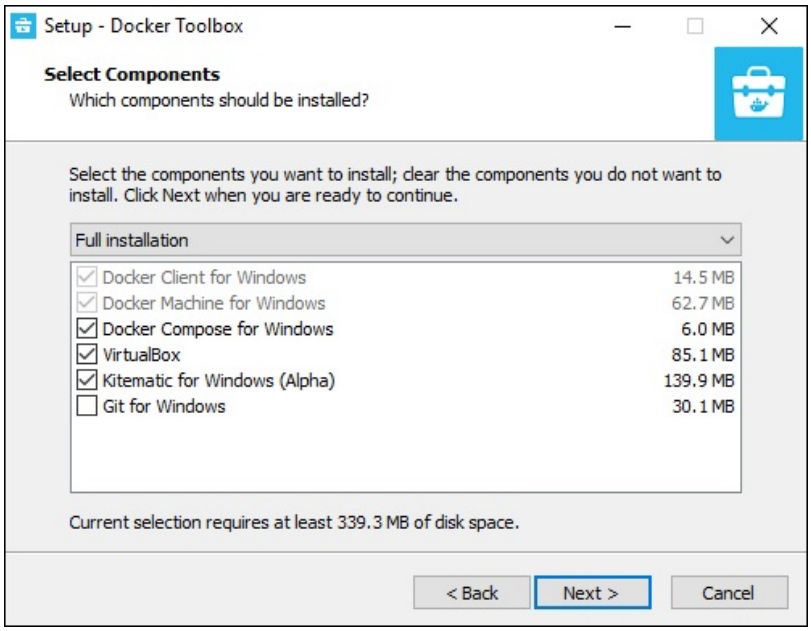
Once the installer has been downloaded, double-click it to start the installer and then follow the steps given below.

Step 1 – Click the Next button on the start screen.

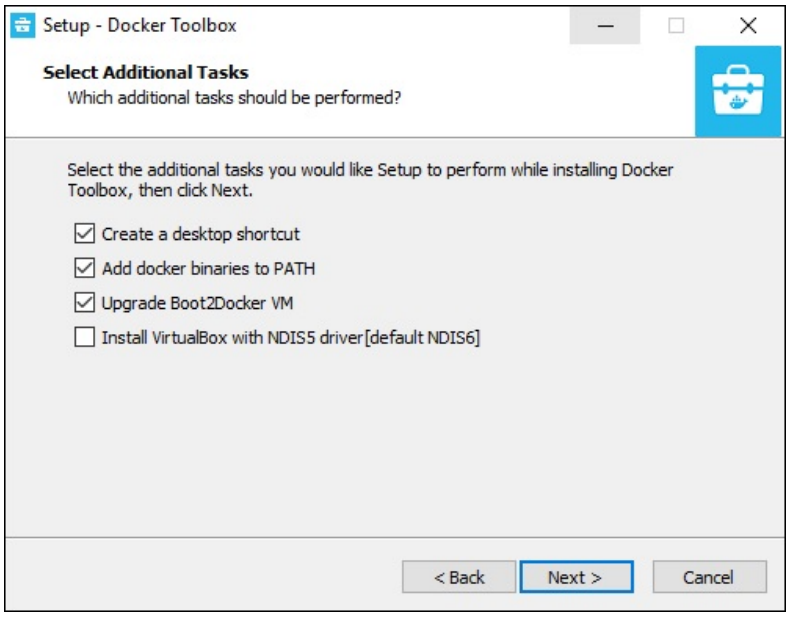
Step 2 – Keep the default location on the next screen and click the Next button.



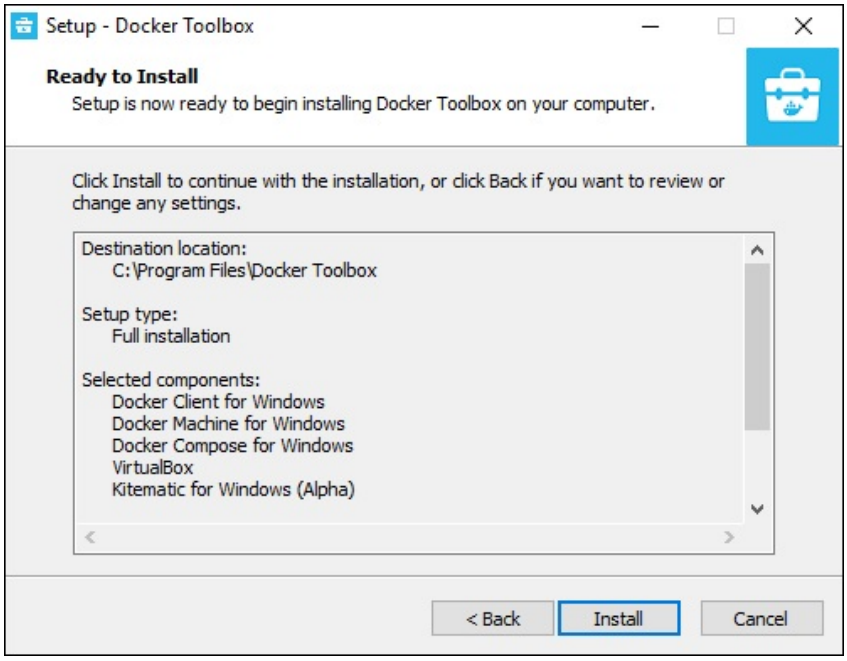
Step 3 – Keep the default components and click the Next button to proceed.



Step 4 – Keep the Additional Tasks as they are and then click the Next button.

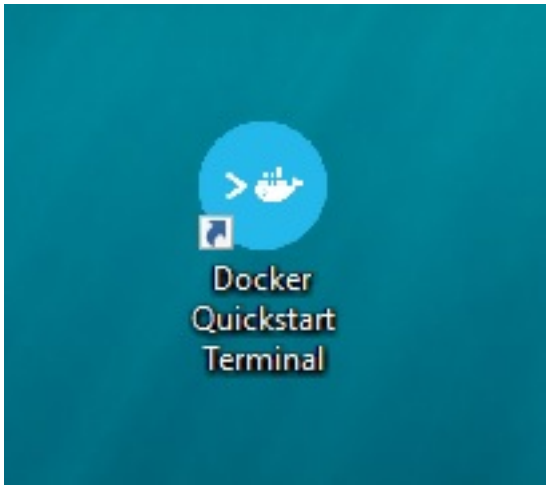


Step 5 – On the final screen, click the Install button.

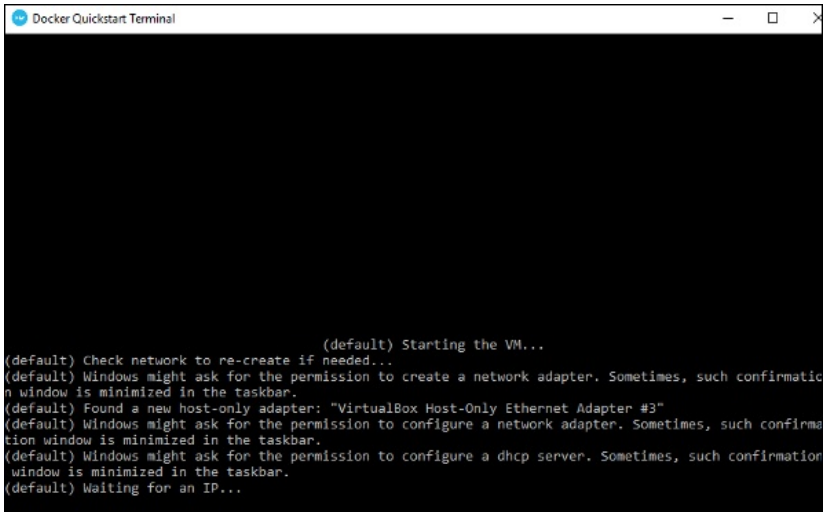


Working with Docker Toolbox

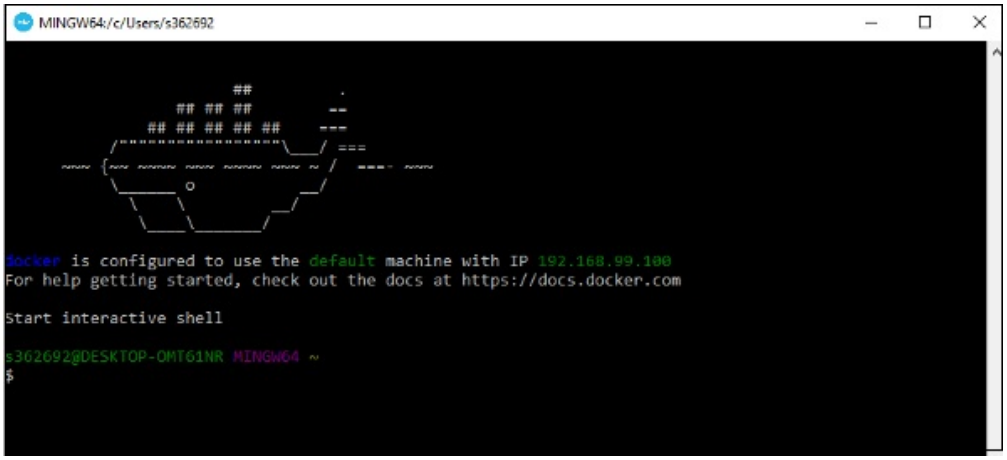
Let's now look at how Docker Toolbox can be used to work with Docker containers on Windows. The first step is to launch the Docker Toolbox application for which the shortcut is created on the desktop when the installation of Docker toolbox is carried out.



Next, you will see the configuration being carried out when Docker toolbox is launched.



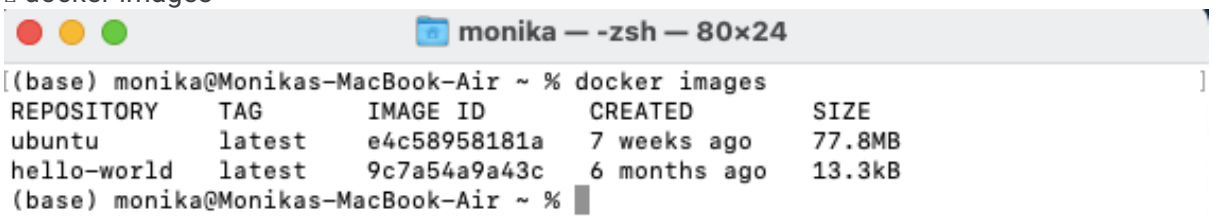
Once done, you will see Docker configured and launched. You will get an interactive shell for Docker.



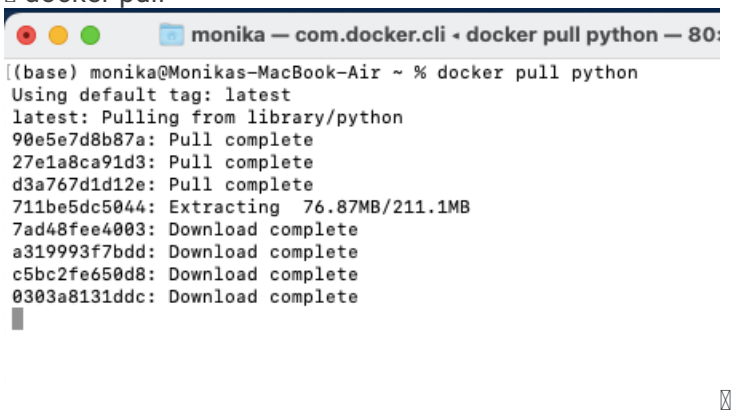
To test that Docker runs properly, we can use the Docker **run command** to download and run a simple **HelloWorld Docker container**.

PRACTICAL 14: Execute Docker Basic Commands(with snapshots):-

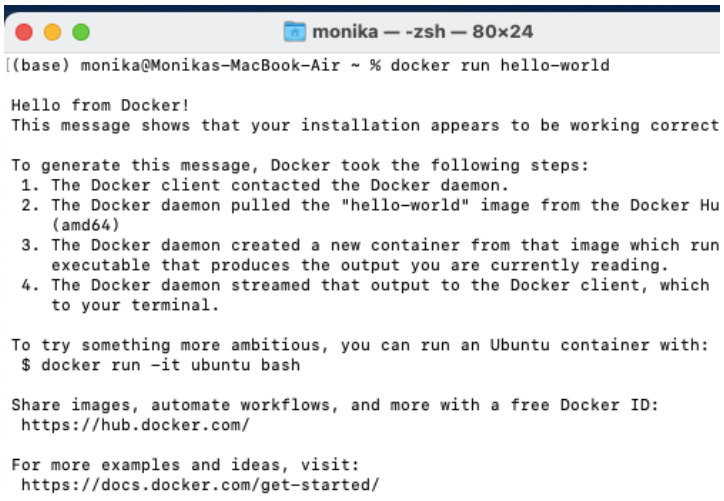
 docker images



 docker pull

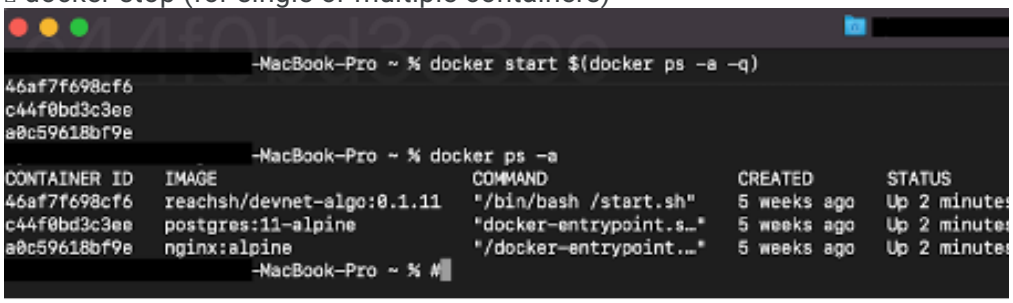


docker run

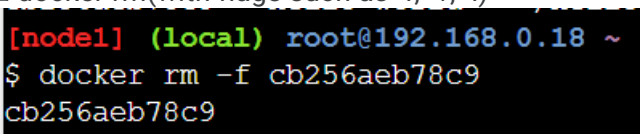


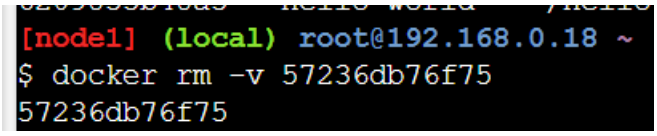
 docker ps (with flags such as -a, -l, -q)

 docker stop (for single or multiple containers)

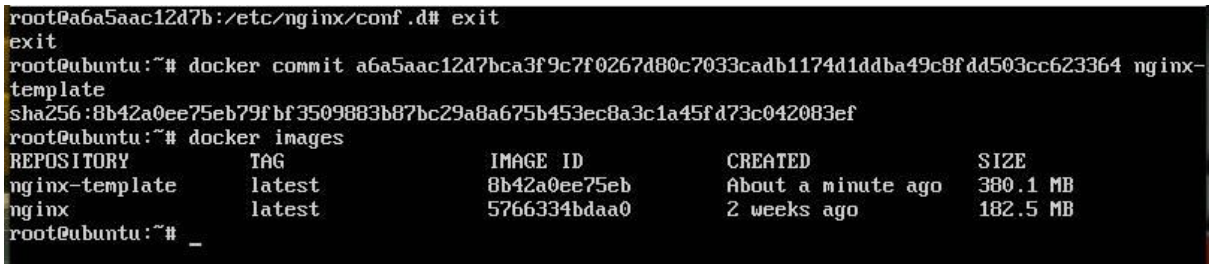


 docker rm(with flags such as -f, -v, -l)





 docker commit



PRACTICAL -15: Execute following Docker Image Commands(with relevant snapshots):-

 docker build command

```
(base) monika@Monikas-MacBook-Air DockerFile % docker build /Users/monika/Desktop/Dockerfile
[+] Building 0.1s (5/5) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 51B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/ubuntu:latest
 => CACHED [1/1] FROM docker.io/library/ubuntu
 => exporting to image
 => => exporting layers
 => => writing image sha256:af93a73f17db33b6c3114478c983c08ed925efc80f759308c2efe991edaedfc5
```

 docker pull command

```
(base) monika@Monikas-MacBook-Air DockerFile % docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
Digest: sha256:c79d06dfdfd3d3eb04cafd0dc2bacab0992ebc243e083cabe208bac4dd7759e0
Status: Image is up to date for hello-world:latest
docker.io/library/hello-world:latest
```

 docker images command

```
Last login: Sun Nov 26 19:27:07 on ttys001
(base) monika@Monikas-MacBook-Air ~ % docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
<none>        <none>    af93a73f17db   7 weeks ago    77.8MB
ubuntu        latest    e4c58958181a   7 weeks ago    77.8MB
hello-world   latest    9c7a54a9a43c   6 months ago   13.3kB
(base) monika@Monikas-MacBook-Air ~ %
```

 docker inspect command

```
(base) monika@Monikas-MacBook-Air ~ % docker image inspect ubuntu:latest
[
    {
        "Id": "sha256:e4c58958181a5925816faa528ce959e487632f4cfd192f8132f71b32df2744b4",
        "RepoTags": [
            "ubuntu:latest"
        ],
        "RepoDigests": [
            "ubuntu@sha256:2b7412e6465c3c7fc5bb21d3e6f1917c167358449fecac8176c6e496e5c1f05f"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2023-10-05T07:33:32.902036554Z",
        "Container": "07c41dd66393978365d5d9a8946bbf45dce5dd50ab3e473284b82a9010d6b265",
        "ContainerConfig": {
            "Hostname": "07c41dd66393",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
            ],
            "Cmd": [
                "/bin/sh",
```

 docker push command

```
(base) monika@Monikas-MacBook-Air ~ % docker tag ubuntu:latest shreythakur07/devops
(base) monika@Monikas-MacBook-Air ~ % docker images
REPOSITORY              TAG       IMAGE ID       CREATED         SIZE
<none>                  <none>    af93a73f17db   7 weeks ago     77.8MB
ubuntu                  latest    e4c58958181a   7 weeks ago     77.8MB
shreythakur07/devops    latest    e4c58958181a   7 weeks ago     77.8MB
hello-world             latest    9c7a54a9a43c   6 months ago    13.3kB
(base) monika@Monikas-MacBook-Air ~ % docker push shreythakur07/devops:latest
The push refers to repository [docker.io/shreythakur07/devops]
256d88da4185: Preparing
```

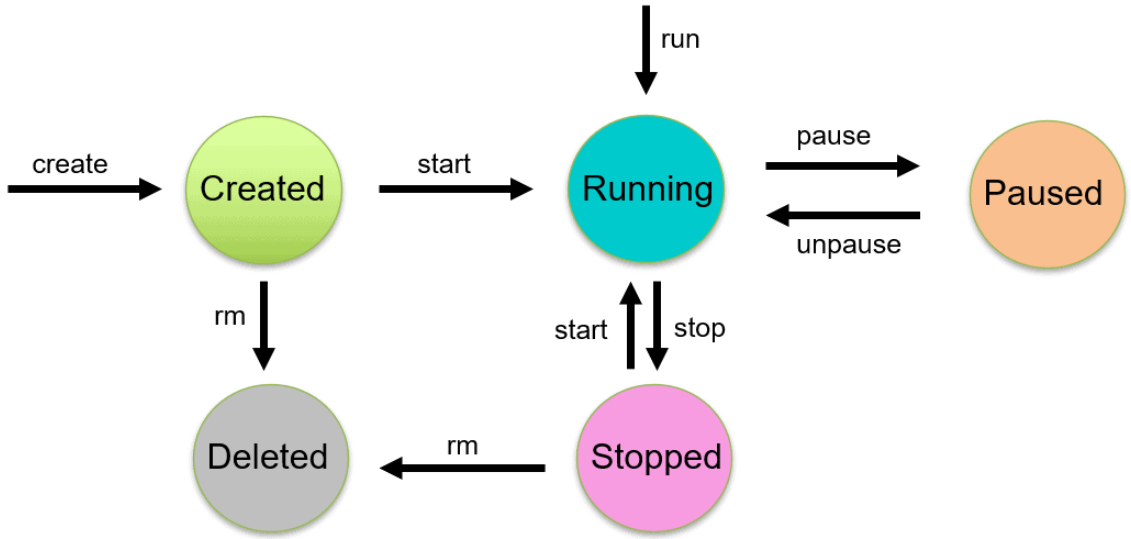
 docker save command

```
(base) monika@Monikas-MacBook-Air ~ % docker save ubuntu > ubuntu.tar
(base) monika@Monikas-MacBook-Air ~ % ls -sh ubuntu.tar
 163968 ubuntu.tar
(base) monika@Monikas-MacBook-Air ~ % docker save --output ubuntu.tar ubuntu
(base) monika@Monikas-MacBook-Air ~ % ls -sh ubuntu.tar
 164416 ubuntu.tar
(base) monika@Monikas-MacBook-Air ~ % docker save -o ubuntu-all.tar ubuntu
(base) monika@Monikas-MacBook-Air ~ % docker save -o ubuntu-all.tar ubuntu:latest
 (base) monika@Monikas-MacBook-Air ~ %
```

 docker rmi command

```
(base) monika@Monikas-MacBook-Air ~ % docker rmi 9c7a54a9a43c
Untagged: hello-world:latest
Untagged: hello-world@sha256:c79d06dfdfd3d3eb04cafd0dc2bacab0992ebc243e083cabe208bac4dd7759e0
Deleted: sha256:9c7a54a9a43cca047013b82af109fe963fde787f63f9e016fdc3384500c2823d
Deleted: sha256:01bb4fce3eb1b56b05adf99504dafd31907a5aadac736e36b27595c8b92f07f1
(base) monika@Monikas-MacBook-Air ~ %
```

Practical -16 Design Docker container life cycle and execute following Docker Container Commands(with relevant snapshots):-



 docker attach

```
Bhargavs-MacBook-Pro:user_api bhargavbachina$ docker attach nodeapi
root@874fdf2b34ba:/usr/src/app#
root@874fdf2b34ba:/usr/src/app# ls
api.bundle.js
root@874fdf2b34ba:/usr/src/app# node api.bundle.js
Sat Feb 16 2019 17:14:25 GMT+0000 (UTC)info:::::Listening on port 3070
^C
root@874fdf2b34ba:/usr/src/app# node -v
v8.11.4
root@874fdf2b34ba:/usr/src/app# exit
exit
Bhargavs-MacBook-Pro:user_api bhargavbachina$
```

 docker ps command

```
(base) monika@Monikas-MacBook-Air myapp % docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                    NAMES
0adebbfb87d4   myapp     "/docker-entrypoint.…"   12 seconds ago   Up 11 seconds   0.0.0.0:8080->80/tcp     laughing_ishizaka
(base) monika@Monikas-MacBook-Air myapp %

(base) monika@Monikas-MacBook-Air ~ % docker ps -a
CONTAINER ID   IMAGE     COMMAND       CREATED             STATUS                          PORTS     NAMES
8cc58096d143   ubuntu    "/bin/bash"   About an hour ago   Exited (0) About an hour ago              exciting_allen
(base) monika@Monikas-MacBook-Air ~ %
```

 docker start and restart

```
(base) monika@Monikas-MacBook-Air ~ % docker container start e45b79cfaf13
e45b79cfaf13
(base) monika@Monikas-MacBook-Air ~ % docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED              STATUS
       PORTS                    NAMES
e45b79cfaf13   nginx     "/docker-entrypoint.…"   About a minute ago   Up 5 seco
nds    80/tcp                   kind_williamson
0adebbfb87d4   myapp     "/docker-entrypoint.…"   11 minutes ago       Up 11 min
utes   0.0.0.0:8080->80/tcp    laughing_ishizaka

(base) monika@Monikas-MacBook-Air ~ % docker container restart e45b79cfaf13
e45b79cfaf13
(base) monika@Monikas-MacBook-Air ~ % docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS
   PORTS                    NAMES
e45b79cfaf13   nginx     "/docker-entrypoint.…"   2 minutes ago    Up 6 seconds
   80/tcp                   kind_williamson
0adebbfb87d4   myapp     "/docker-entrypoint.…"   12 minutes ago   Up 12 minutes
   0.0.0.0:8080->80/tcp     laughing_ishizaka
```

docker port mapping

```
(base) monika@Monikas-MacBook-Air ~ % docker run -p 9090:90 python
(base) monika@Monikas-MacBook-Air ~ % docker run -d -p 9090:90 python
34188d79d7e05d2bbac84470d9aa9a0ac14f20e86d76e70a259e716d4330d0e7
```

```
(base) monika@Monikas-MacBook-Air ~ % docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                  NAMES
e45b79cfaf13   nginx     "/docker-entrypoint.…"   14 minutes ago   Up 12 minutes   80/tcp                 kind_williamson
0adebbfb87d4   myapp     "/docker-entrypoint.…"   24 minutes ago   Up 24 minutes   0.0.0.0:8080->80/tcp   laughing_ishizaka
(base) monika@Monikas-MacBook-Air ~ %
```

docker container command

```
(base) monika@Monikas-MacBook-Air ~ % docker container ls
CONTAINER ID   IMAGE     COMMAND                  CREATED
e45b79cfaf13   nginx     "/docker-entrypoint.…"   14 minutes ago
0adebbfb87d4   myapp     "/docker-entrypoint.…"   24 minutes ago
(base) monika@Monikas-MacBook-Air ~ %
```

docker container inspect infinite Command

```
(base) monika@Monikas-MacBook-Air ~ % docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS                  NAMES
e45b79cfaf13   nginx     "/docker-entrypoint.…"   18 minutes ago   Up 15 minutes   80/tcp                 kind_williamson
0adebbfb87d4   myapp     "/docker-entrypoint.…"   28 minutes ago   Up 28 minutes   0.0.0.0:8080->80/tcp   laughing_ishizaka
(base) monika@Monikas-MacBook-Air ~ % docker inspect  kind_williamson
[
    {
        "Id": "e45b79cfaf133f1666cfadc167850ba20accf1ea40c9489f122e039c5e25f626",
        "Created": "2023-11-27T17:00:04.085040475Z",
        "Path": "/docker-entrypoint.sh",
        "Args": [
            "nginx",
            "-g",
            "daemon off;"
        ],
        "State": {
            "Status": "running",
            "Running": true,
            "Paused": false,
            "Restarting": false,
            "OOMKilled": false,
            "Dead": false,
            "Pid": 2242,
            "ExitCode": 0,
            "Error": "",
            "StartedAt": "2023-11-27T17:02:24.490775368Z",
            "FinishedAt": "2023-11-27T17:02:24.055455034Z"
        },
```

docker exec command (with flags such as -d, -i, -e)

```
(base) monika@Monikas-MacBook-Air ~ % docker exec 0adebbfb87d4 ls
bin
boot
dev
docker-entrypoint.d
docker-entrypoint.sh
etc
home
lib
lib32
lib64
libx32
media
```

```
(base) monika@Monikas-MacBook-Air ~ % docker exec -it 0adebbfb87d4 /bin/sh
# ls
bin   docker-entrypoint.d   home   lib64   mnt   root  srv  user
boot  docker-entrypoint.sh  lib    libx32  opt   run   sys  usr
dev   etc                   lib32  media   proc  sbin  tmp  var
# ^C
#
```

docker cp command

```
root@docker-512mb-nyc3-01:~# docker ps -a
CONTAINER ID        IMAGE                   COMMAND                  CREATED             STATUS              PORTS                     NAMES
617e324ef88a        chessbrainfrontend_web   "npm start"              2 hours ago         Up 2 hours          0.0.0.0:3000->3000/tcp    chessbrainfrontend_web_1
62e2a397e69a        mongo                   "docker-entrypoint..."   2 hours ago         Up 2 hours          0.0.0.0:27017->27017/tcp  chessbrainfrontend_db_1
811f36499fb6        chessbrainfrontend_api   "python app.py"          2 hours ago         Up 2 hours          0.0.0.0:5000->5000/tcp    chessbrainfrontend_api_1
2c93438f608c        docker/compose:1.13.0   "docker-compose up"      2 hours ago         Up 2 hours                                    wizardly_galileo
root@docker-512mb-nyc3-01:~# ls
ChessBrain-FrontEnd  hello.py  node-tutorial-for-frontend-devs
root@docker-512mb-nyc3-01:~# docker exec 617e324ef88a ls
API
Dockerfile
app.json
chessBrainTodo
data
docker-compose.yml
docs
node_modules
npm-debug.log
package-lock.json
package.json
procfile
public
server.js
views
root@docker-512mb-nyc3-01:~# docker cp 617e324ef88a:procfile .
Error response from daemon: Could not find the file procfile in container 617e324ef88a
root@docker-512mb-nyc3-01:~#
```