__/__/____

Git & Github

Git is a Version control system

→ Easily recover files

→ Who introduce an issue and when ?

→ Roll back to previously working state

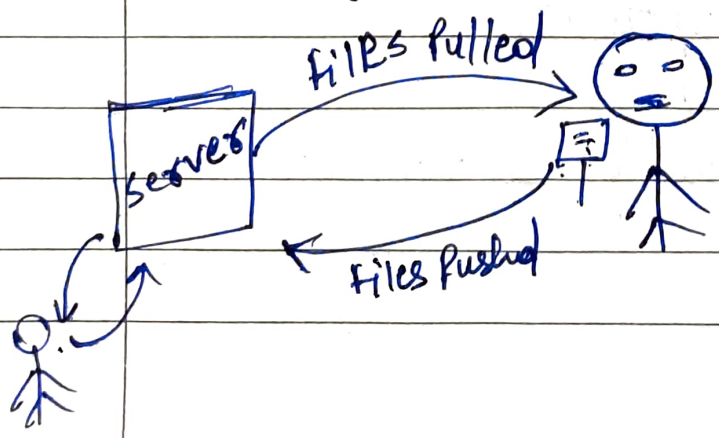
History of Version Control System

→ Local VCS → Database to keep track of files

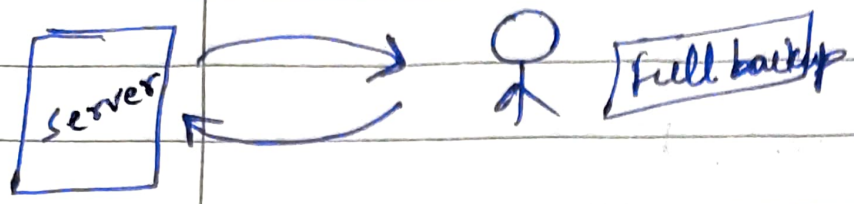
pros → can track files & roll back

Cons → If u lose ur hardisk, everything will be lost.

→ Centralized Vcs



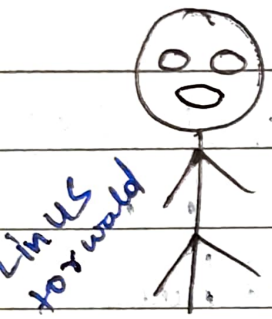
→ Distributed Vcs



Grit is a Distributed Version control system

How was Grit Created

1991 — 2002



2002 → Bitkeeper Vcs

2002 — 2005

2005 → free of charge status was revoked

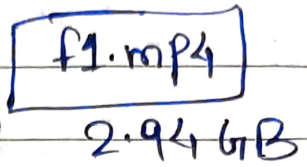
Then Linus torwald created Grit

Grit is a Software

Github is a hosting-service

→ Almost every operation is local

→ Git has integrity



SHA-1 checksum

$ git config --global user.name "Bharat"

$ git config --global user.email "bharatk503@gmail.c

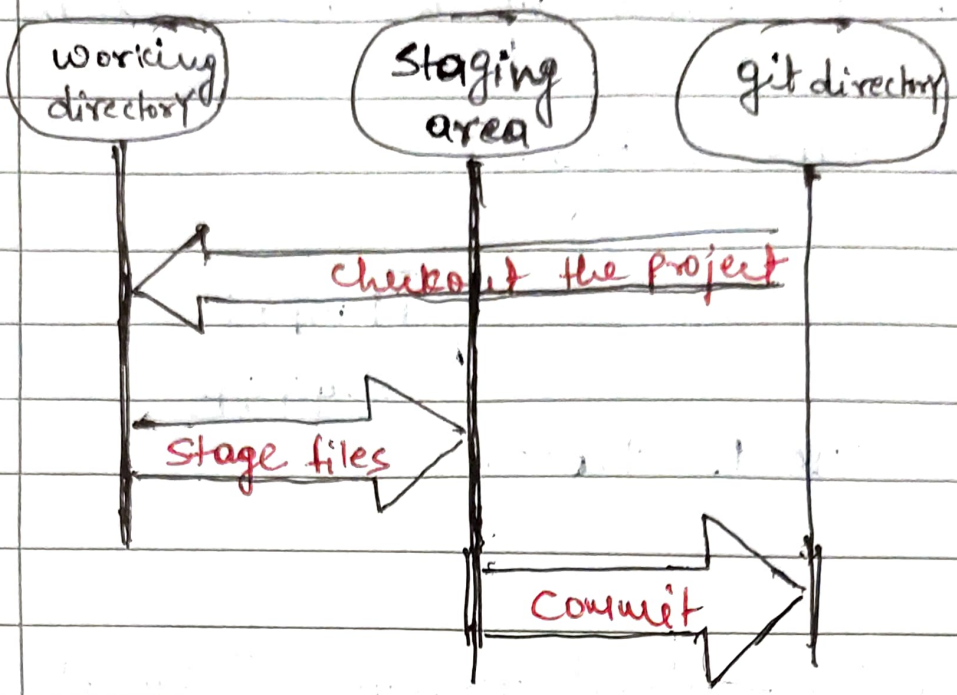
To know the name type

→ $ git config user.name

Git: Three stage Architecture

__/__/____

Local operations



save the snapshot

⟹ commit

index.html ✓

engine.js → error ✗

index.css ✓

4: Tracking our first Git project

$ git status

↳ Not a git repository

Then do

↳ $ git init

__/__/____

Then .git repository is created

Again do

↳ git status

Then it will show all the untracked directories

$ git add --a

↳ put all the files in the staging area

changes to be committed

$git commit -m "Initial commit"

↳ All the files is commited

$ git log

5: Cloning a Remote Git Repository from Github

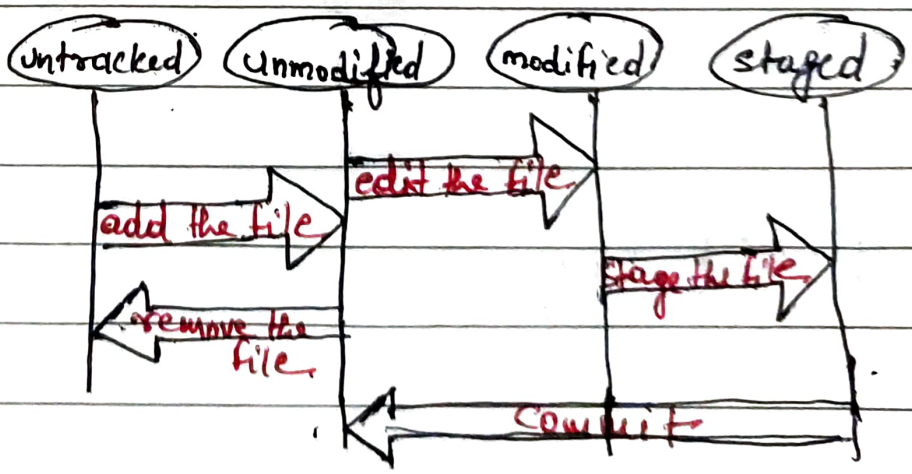
To delete git Repository

→ rm -rf .git ← Repository

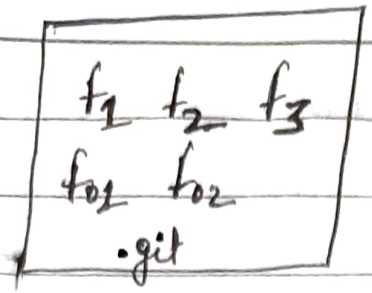
$ git clone url

6: File Status Lifecycle

File status Lifecycle



__/__/____



untracked

→ git add --a

f1, f2, f3, fo1, fo2 --- unmodified Now tracked

changed

f1, f2 → modified

git add ⇒ staged

---

7 .gitignore: Ignoring files in Git

$ touch error.log → A new file is created

$ touch .gitignore → file is created

__/__/____

If we put error.log in .gitignore file then there is no trace of error.log

Ignore all log files *.log

for direct /dir

folder name

/static

8: Git Diff : Showing changes Between Commits /staging

Git diff compare working directory with staging area

$ git diff --staged

↳ compare previous commit to current staging area

9: Git: Skipping The Staging Area

$ git status

$ git add . → put the file in staging Area

$ git commit -a -m "Direct commit"

$ git status → we will see that untracked file is not commited

10: Moving and Renaming files In Git

$ git rm third.txt ← Removed from file

$ git commit -m "Removed file third.txt"

→ move command

$ git mv first.txt first_renamed.txt

← file is also staged

$ git rm --cached db.accdb

← Now db.accdb will not be tracked

11: Git Log: Viewing & Changing Commit in Git

$ rm -rf .git

$ git log --pretty=oneline

$ git log --pretty=short

$ git log --since=2.days
=2.weeks
=2.months

$ git log --pretty=format:"%h -- %an"

%h → Abbreviated commit

%an → author name

-- %ae"

$ git log -p -1

12 Unstaging & Unmodifying Files in Git

$ git restore --staged second.txt

Go back to previous commit

↳ $ git checkout -- first_rename.txt

$ git checkout -f

13 GitHub : Working with Remote

$ git remote

$ git remote add origin git@github.com/bharathk5003/First.git

$ git remote -V

$ git push -u origin master

⟹ This will give error

Permission denied

Now to get the permission we will

__/__/____

move to settings in GitHub then

SSH & GPG keys

ssh-keygen

14 Setting Alias In Git

$ git st ← error

$ git config --global alias.st status

$ git st ← now this will work

Now make alias for 'restore --staged'

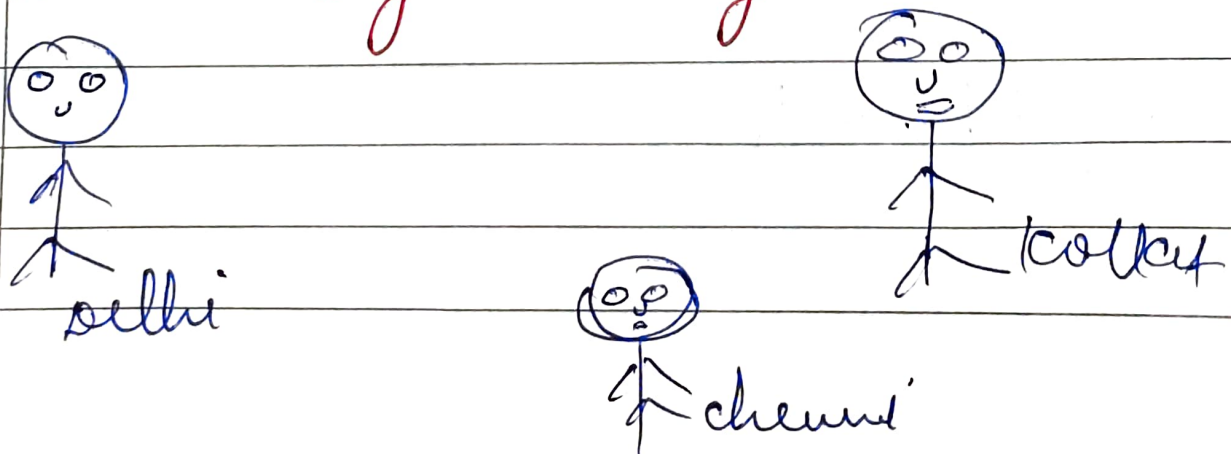
$ git config --global alias.unstage 'restore --staged'

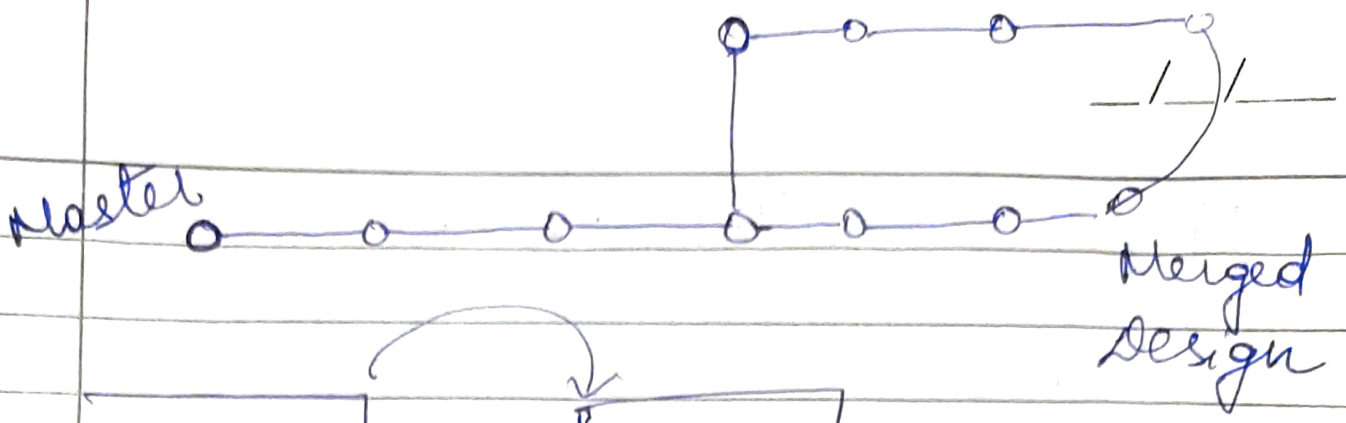
$ git unstage file1.txt

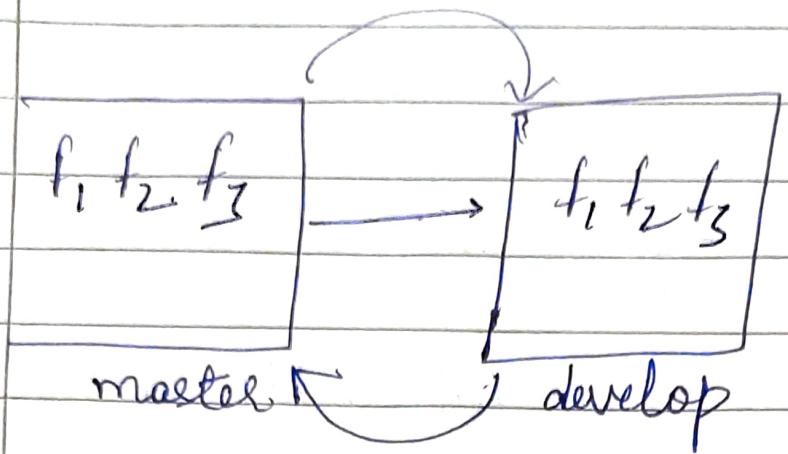
Last commit → git config --global alias.last 'log -p -1'

$ git last

15 Git : Creating & Switching Branches In Git







develop branch not present

```
$ git checkout -b develop
```

↳ will create a new branch develop & take us there

```
$ git checkout master
```

↑ Move to master branch

```
$ git checkout develop
```

```
$ git branch
```

__/__/____

16: Branching & Merging a production Grade project

$ rm -rf .git ⇐ delete the hidden .git file (remove git init)

→ the git repository being deleted

$ git checkout -b trycleanup ← new branch

$ do some changes then commit the changes to this branch

then

$ git checkout master ← return to master branch

← From master branch

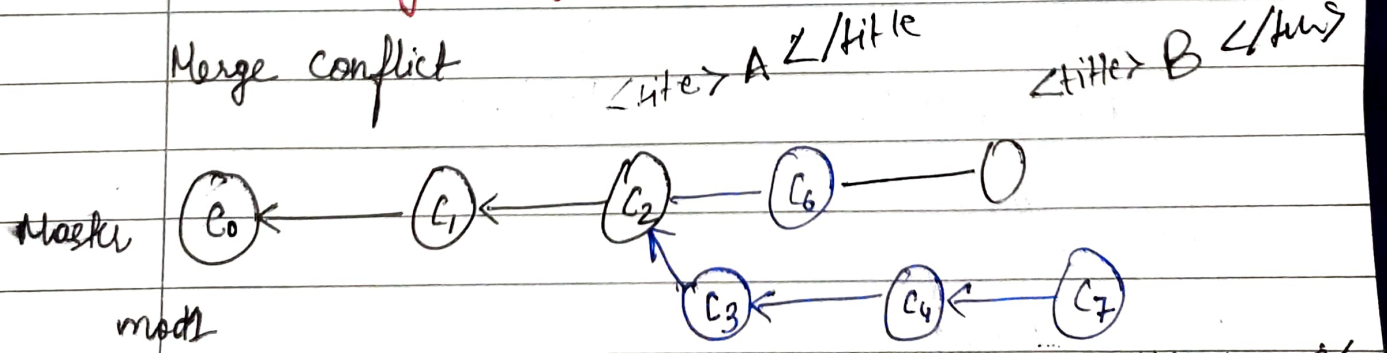
$ git merge trycleanup

17: Resolving Merge Conflicts

Merge conflict

<title> A </title>

<title> B </title>



<title> C </title>

git merge mod1

Branch Management

git branch
* master
mod1
develop

git branch -v

git branch --merged

git branch --no-merged

Deleting Branches

git branch -d develop
mod1

branch name

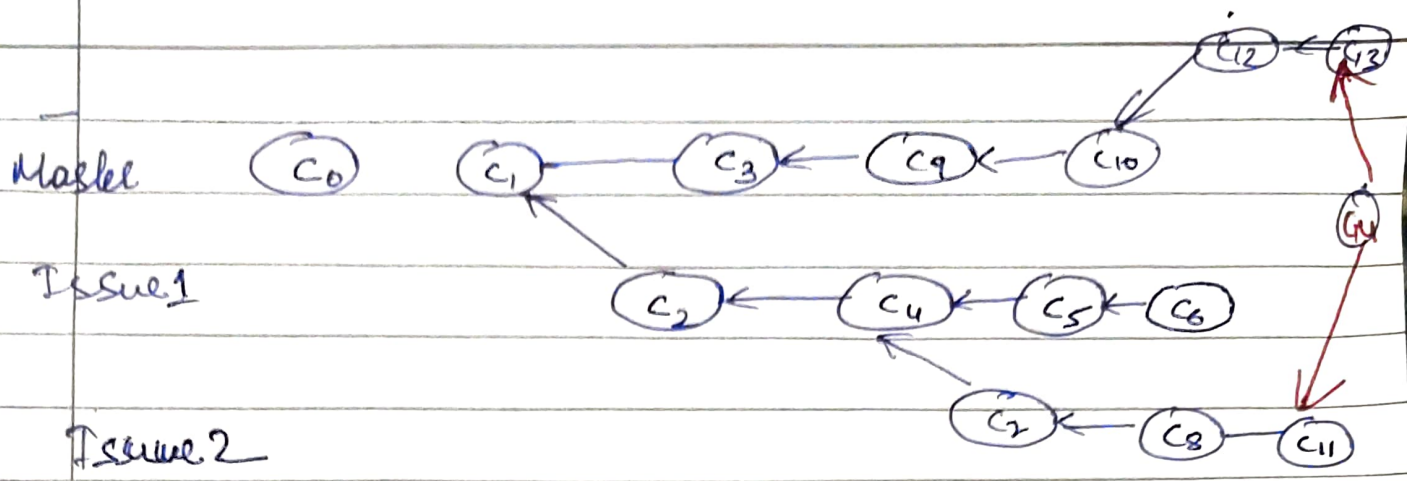
gives error if develop is not merged

18: Git Branching Workflow in production

⇒ Branching workflow:

- Long Running Branches
  - master
  - develop
  - mod1
- Topic Branches

__/__/____



19: pushing Git Branch To Remote Repositories

$ git checkout -b bugfix

↳ Then do some changes then

$ git commit -m "Bug is Fixed"

$ git push origin master

$ git push origin bugfix

Delete branch from remote

$ git push -d origin bugfix