Git & GitHub

GitHub
GIT CHEAT SHEET

Git is the free and open source distributed version control system that's responsible for everything GitHub related that happens locally on your computer. This cheat sheet features the most important and commonly used Git commands for easy reference.

INSTALLATION & GUIS

With platform specific installers for Git, GitHub also provides the ease of staying up-to-date with the latest releases of the command line tool while providing a graphical user interface for day-to-day interaction, review, and repository synchronization.

GitHub for Windows
https://windows.github.com

GitHub for Mac
https://mac.github.com

For Linux and Solaris platforms, the latest release is available on the official Git web site.

Git for All Platforms
http://git-scm.com

SETUP

Configuring user information used across all local repositories

`git config --global user.name "[firstname lastname]"`

set a name that is identifiable for credit when review version history

`git config --global user.email "[valid-email]"`

set an email address that will be associated with each history marker

`git config --global color.ui auto`

set automatic command line coloring for Git for easy reviewing

git config --list (what we setted up using git config)

SETUP & INIT

Configuring user information, initializing and cloning repositories

`git init`

initialize an existing directory as a Git repository

`git clone [url]`

retrieve an entire repository from a hosted location via URL

STAGE & SNAPSHOT

Working with snapshots and the Git staging area

`git status`

show modified files in working directory, staged for your next commit

`git add [file]` git add . [add all changes]

add a file as it looks now to your next commit (stage)

`git reset [file]`

unstage a file while retaining the changes in working directory

`git diff` main → main branch এর সাথে তার branch এর diff (create করে)

diff of what is changed but not staged

`git diff --staged`

diff of what is staged but not yet committed

`git commit -m "[descriptive message]"`

commit your staged content as a new commit snapshot

git push origin main

Push command : upload local repo content to remote repo

BRANCH & MERGE

Isolating work in branches, changing context, and integrating changes

`git branch`

list your branches. a * will appear next to the currently active branch

`git branch [branch-name]`

create a new branch at the current commit

`git checkout`

switch to another branch and check it out into your working directory

`git merge [branch]`

merge the specified branch's history into the current one

`git log`

show all commits in the current branch's history

INSPECT & COMPARE

Examining logs, diffs and object information

`git log`
show the commit history for the currently active branch
`git log branchB..branchA`
show the commits on branchA that are not on branchB
`git log --follow [file]`
show the commits that changed file, even across renames
`git diff branchB...branchA`
show the diff of what is in branchA that is not in branchB
`git show [SHA]`
show any object in Git in human-readable format

SHARE & UPDATE

Retrieving updates from another repository and updating local repos

`git remote add [alias] [url]`
add a git URL as an alias
`git fetch [alias]`
fetch down all the branches from that Git remote
`git merge [alias]/[branch]`
merge a remote branch into your current branch to bring it up to date
`git push [alias] [branch]`
Transmit local branch commits to the remote repository branch
`git pull`
fetch and merge any commits from the tracking remote branch

TRACKING PATH CHANGES

Versioning file removes and path changes

`git rm [file]`
delete the file from project and stage the removal for commit
`git mv [existing-path] [new-path]`
change an existing file path and stage the move
`git log --stat -M`
show all commit logs with indication of any paths that moved

REWRITE HISTORY

Rewriting branches, updating commits and clearing history

`git rebase [branch]`
apply any commits of current branch ahead of specified one
`git reset --hard [commit]`
clear staging area, rewrite working tree from specified commit

IGNORING PATTERNS

Preventing unintentional staging or commiting of files

```
logs/
*.notes
pattern*/
```

Save a file with desired patterns as .gitignore with either direct string matches or wildcard globs.

`git config --global core.excludesfile [file]`
system wide ignore pattern for all local repositories

TEMPORARY COMMITS

Temporarily store modified, tracked files in order to change branches

`git stash`
Save modified and staged changes
`git stash list`
list stack-order of stashed file changes
`git stash pop`
write working from top of stash stack
`git stash drop`
discard the changes from top of stash stack

Git

skill

Version Control System is a tools that helps to track changes in code.
Git is a Version Control System. It is :

popular
free & Open Source
fast & scalable

1) Track the history
2) Collaboration

Github

Website that allows developers to store & manage their code using Git.
https://github.com

- Project upload as Folder (Repository)
- Read me file → project info
- Finalize changes in github → "Commit"

Add → commit

HTML

Setting up Git

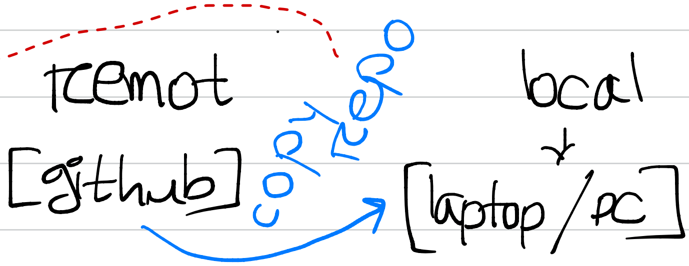
Visual Studio Code
Windows (Git Bash)
Mac (Terminal)
git - - version (To check the set up)

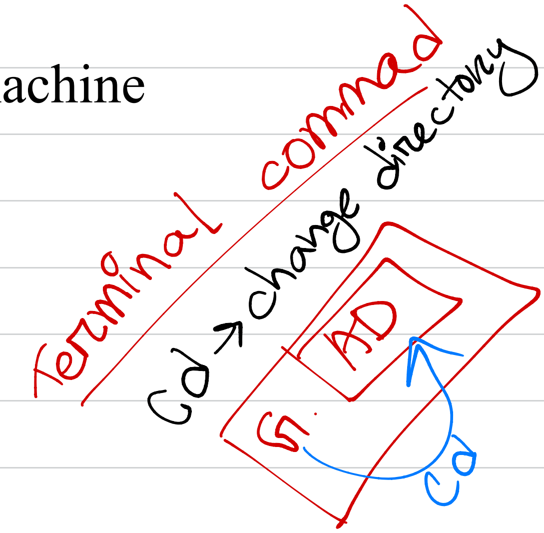
~ (Tilde) → root directory
↓
primary Folder (From where we will configure git)

Clone & Status

- Clone - Cloning a repository on our local machine

git clone <- some link ->





- status - displays the state of the code

git status

⇒ cd Folder name
or
cd Tab button → auto fill

ls → file list
ls -a → Hidden file
ls -Hidden

⇒ cd ..
directory/folder থেকে বাইরে বেরোবে

- After modifying any file in vs
We have to save the change by
Commit command

add → commit

⇒ mkdir name
make new directory
PC তে Folder বানাবে

↳ git এ বানাবে Folder খোলে নেবে vs এ New Folder বানাবে

- git status (4 types)

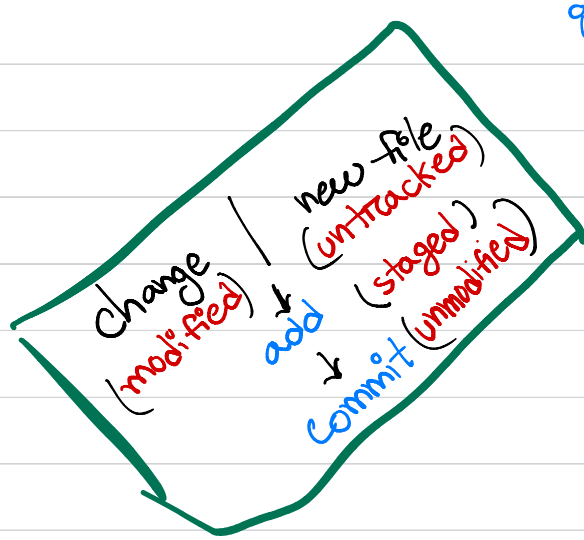
untracked
new files that git doesn't yet track

⇒ cd new file name

⇒ git init (pc তে created file কে git repo হিসেবে initialize)

modified
changed

staged
file is ready to be committed



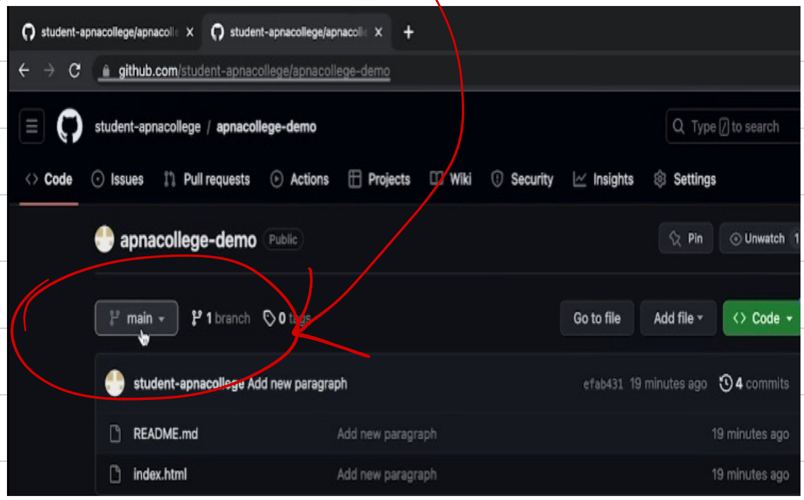
unmodified
unchanged

git push origin main

origin → যতগুলো Remote Repo আছে এর মধ্যে আমাদের origin নামে যেখান থেকে clone নিয়েছিলাম সেটাতে push করতে চাচ্ছি

main → branch name

origin আমাদের রিপোজিটরির নাম



Init Command

init - used to create a new git repo

git init

git remote add origin <- link -> (remote → github repo; origin → নাম)

git remote -v (to verify remote)

github এ যেই নামে new repo খুলছি বা যে repo এর Link দিছি সেটার নাম দিবে

git branch (to check branch)

(এখন কোন branch এ আছি)

git branch -M main (to rename branch)

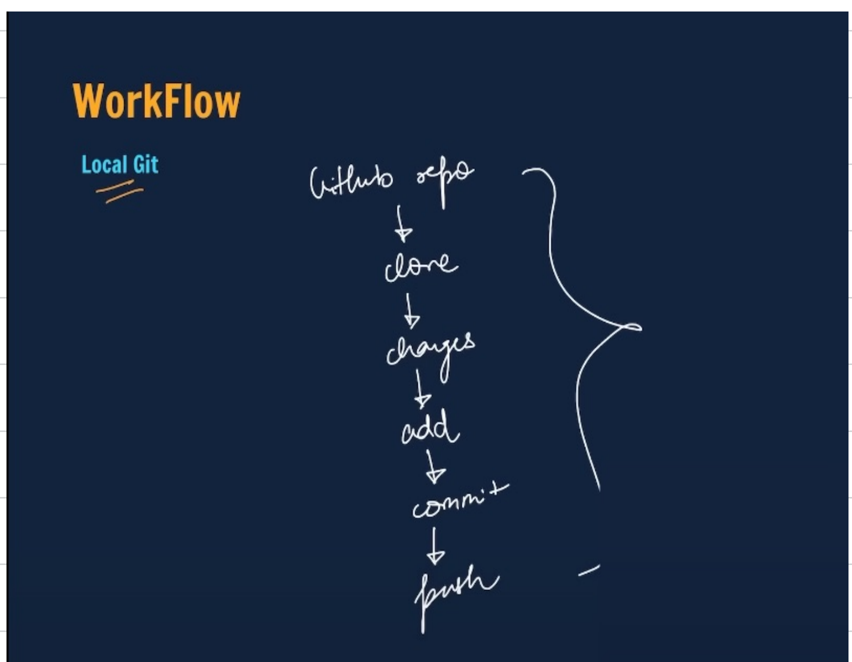
git push origin main

↓

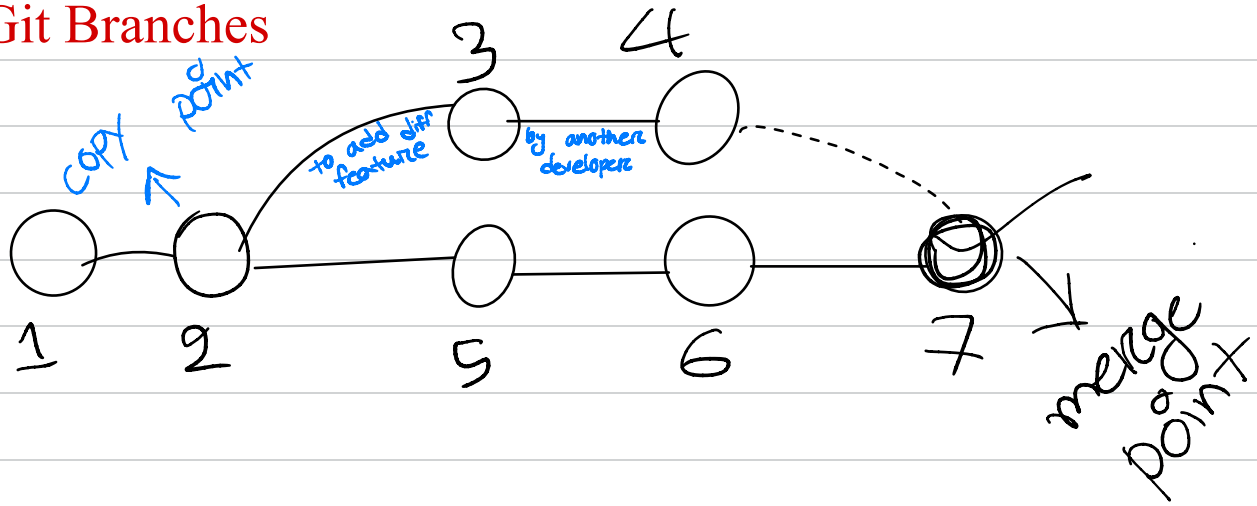
git push -u origin main

-u → get upstream

পরবর্তী থেকে Just git push দিলেই enough



Git Branches



Branch Commands

git branch (to check branch)
git branch -M main (to rename branch)

any name for new branch

git checkout <- branch name -> (to navigate) → copy branch into another branch
git checkout -b <- new branch name -> (to create new branch)
git branch -d <- branch name -> (to delete branch)

Merging Code

Way 1 (Through Vs code)

git diff <- branch name-> (to compare commits, branches, files & more)
git merge <- branch name-> (to merge 2 branches)

Way 2 → Through GitHub

Create a PR

Pull Request
It lets you tell others about changes you've pushed to a branch in a repository on GitHub.

Pull Command

git pull origin main

used to fetch and download content from a remote repo and immediately update the local repo to match that content.

github ↓ PR করার পর এখানে branch গুলো merge হয়ে যায় but VS ↓ show করে না, তাই git থেকে merged টা local ↓ pull করতে হবে।

Resolving Merge Conflicts

An event that takes place when Git is unable to automatically resolve differences in code between two commits.

যখন main এ আগে copied branch এর / ২টা branch ↓ same line ↓ diff থাকে তখন github বুঝে না কোনটা রাখবে manually সেলে দিতে হয়।

1) PR
2) Git merge

Undoing Changes

HEAD=1

(code)

hash

Case 1 : staged changes (add)

git reset <- file name ->

git reset

Case 2 : commited changes (for one commit)

git reset HEAD~1

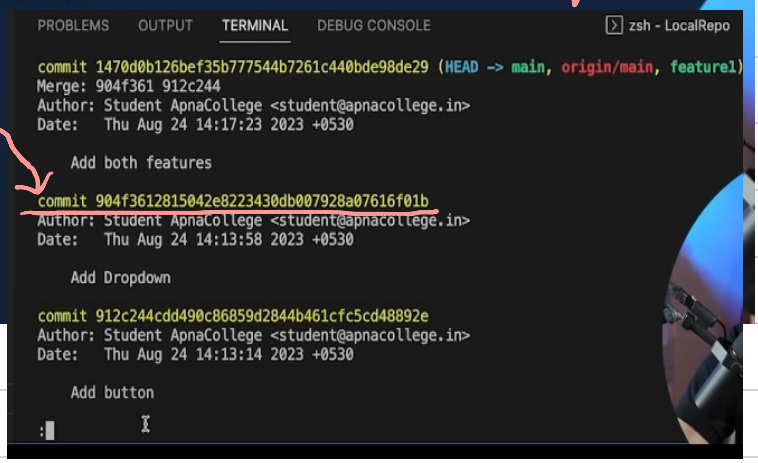
current head থেকে 1 step পিছনে ⇒ git log → prev commits check

Case 3 : commited changes (for many commits)

git reset <- commit hash ->

git reset --hard <- commit hash ->

vs থেকেও মুছে যাবে



Fork

A fork is a new repository that shares code and visibility settings with the original "upstream" repository.

Fork is a rough copy.