GIT

(Use lowercase only)

The control & Tracking of code changes over time

1. It's a version control management or source code management
2. Its used to track the changes with in the file
3. It's a platform independent
4. It can handle large projects efficiently
5. It's a free and open source

* Initially install the GIT application or the IDE or bash.
* Git workflow consists of three regions (working directory, staging area , Repository )
* Repository in git are (Local , Central , Remote )
* Local (with in computer) remote (we can send data from 1-4 by skipping 2,3) central (where all users can accs)
* git Init – to initialize the git
* Git Branch - to find the current working branch.
* Git status – to check the status of the git and shows the number of files without being tracked
* Git add filename – to add the files into the git
* Git add . – to add all files with in the folder (. indicates all files )
* Git commit -m “write your message ” filename — to commit some message into the file and use to track the files
* Git log – to check the recent activities or the logs
* Git config –global core.editor”Editorname.exe -multiinst -nosession” – to make any editor as default editor
* Git config -global -e – to open the default editor
* Types of status Untracked & Tracked (unmodified , Modified , staged)
* If we add the files from untracked it moves into staged state upon that if we do commit then it goes to modified state.
* Unmodified files can't be seen by doing git status
* Git config –global alias.aliasname “original full command”

**How to push Git files to the GIT HUB**

* Initially create a GIT HUB account & sign up
* Create a repository in it ( public , private )
* Git remote add origin URL – add the repository url upon doing this the after pushing all files that can be seen under the github repository.
* Git fetch (to fetch all files )  
  Git merge origin/branchname (which mergers the files in that branch)
* Git pull origin repo name –to pull the files into your local repository
* Git push origin reponame –to push the files into the git hub repository
* Git push -u origin branchname – upon entering the command it will ask you to enter the credentials like username, password or the tokens. After entering all the files will get pushed to the github.

**GIT Branches**

**Its a series of commits along with the files, dir, workprofiles**

* It allows you to create, list, rename,and delete branches
* The default branch name is MASTER
* Git branch – to check the current working branch
* Git log –oneline (to check the all content with that branch)
* Git branch -a –to display all the branches \
* Git commit -am “to add & commit in 1 command” (existing local repo files or dir )
* Git commit –amend -m “content added and file modified” filename (this how we can modify the existing file and can add and commit in a single command)
* Git branch branchname – to create a new branch
* Git checkout branchname – to switch between the branches or to switch to the new branch name
* Git checkout -b newbranchname – to create a new branch and also to check in as well
* Git checkout -m newbranchname oldbranchname – to rename the branches or to move
* Git clone -b branch url – to clone a branch
* Git clone url newreponame – to clone the repo with a newname
* Git branch -d branchname – to delete a branch
* Git branch -D branchname - is used to force delete a branch
* Git push origin -d branchname : to delete the branch on the remote (github)
* To merge the branches initially we have to checkout to the destination branch then
* Git merge branchname – to merge the changes with the destination branch
* Git branch –merge —to get the details of the branches that are merged and some are not even merged
* Git branch –no-merge –to get the details of the branches that havent been merged yet.

**Basics**

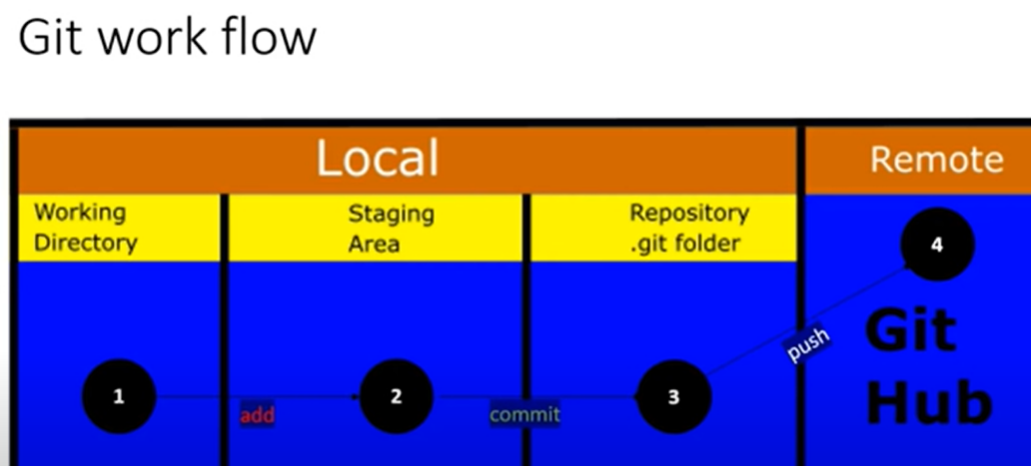
* Inorder to ignore few files we need to create a **.gitignore** initially blank lines and #comments are excluded.
* We can specify the file name in the .**gitignore** file
* Git add -f filename (to add a file forcefully into staging area)
* Git diff : it shows the differences with in the file
* Git diff commit-id..commit-id – to get the difference in the files
* Git commit -v – here -v indicates verbose which gives the full details about the last commit and what we have committed as well (data)
* Git show commit-id (to see the content of the committed file or the directory)
* Git log –since=YYYY-MM-DD ( to search the older dates committed files)
* Got log –author=”ravi” (to seach all files whos author is ravi)
* Git log – to get the logs of the repo to get the differences use -p
* Git log -2 —to get the last two commits
* Git log —oneline –to get the committed logs in one line
* git log --oneline --all —to get the all committed logs in oneline
* Git lig –oneline –graph –all —-to get the committed logs in graphical manner
* Rm filename (to remove files)
* Git checkout filename (to restore files)
* Git rm filename (to delete permanently from all repo’s )
* Git rm –cached filename ( it makes the file to untracked from tracked)
* Git mv oldname newname (to rename a file permanently)

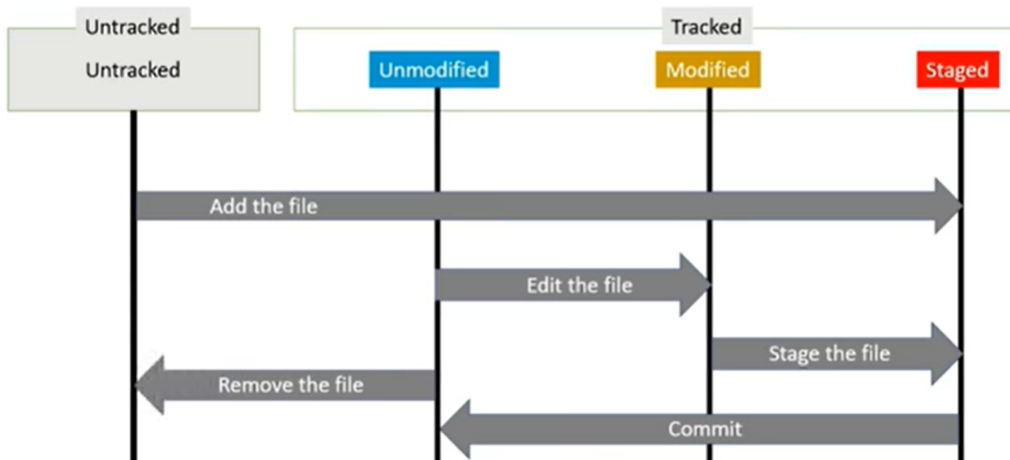
GIT STASH

* Git stash – its a recycle bin it stashes the staged files
* Git stash list –to check the list of deleted items by indexing
* Git stash apply –to apply & get the latest changed stash
* Git stash drop –to remove from stash
* Git stash apply stash@{3} –to apply the 3rd stash
* Git stash pop –it is used to apply and drop it
* Git stash -u –it stashes the untracked files
* Git clean –it delete the untracked files
* Git clean -f –to delete the untracked files forcefully.
* Git clean -f -d –its to delete the untracked files & directories forcefully
* Git clean -f -n –it list all the untracked files that we want to delete.
* Git tag tagname –its used for naming convention
* Git commit -am “write the comment” –to add and commit at a time Express commit
* Git tag -a tagname –here a is used for annotated tag
* Git tag -a tagname -f commitid –to update or to modify the tags

GIT DELETE

* Git reset –soft commit-id (which delete from the local repo)
* Git reset –mixed commit-id (which deletes from staging area)
* Git reset –hard commit-id (which will gives us the older version as per the commit id)





**Interview Questions**

* How to create or initilize a git repository = git init
* From which branch does the releases takes place = release branch
* Which is the file where git referes for tracking and everything = .git file
* How can we prevent sensitive and password to push into git = using hooks/pre-commit
* How can you able to see the commits performed by git = git log
* What is the git workflow that you use in your org = add,commit -m , push, pull, clone
* How can you see the reference for a branch = Navigate to folder & git remote -v
* Primary diff btw clone and fork= fork is used to copy of a repo and clone is to download specific repo
* Best merging technique if they are under 5 commits = git cherry-pick commit\_ID
* How can you make tracked file to untracked = git rm –cached filename