**GIT(version control system)**

**Git** is a distributed version control system used for tracking changes in source code during software development. It enables multiple developers to work on a project simultaneously without overwriting each other's changes, ensuring a full history of every change made to the codebase.

**KEY features of Git :-**

* **Distributed Version Control**: Each developer has a local copy of the entire repository, including its history.
* **Branching and Merging**: Git allows easy creation of branches for developing features or experimenting and then merging them back into the main branch.
* **Tracking Changes**: Git tracks changes to files, allowing users to see who made changes and when.
* **Commit History**: Each change is stored as a "commit" with a unique identifier, allowing users to navigate through the history of the project.

**GIT Commands :-**

**git config :-**

Usage: git config –global user.name “[name]”

Usage: git config –global user.email “[email address]”

This command sets the author name and email address respectively to be used with your commits.

Git Config Command - Git Commands - Edureka

### git init :-

Usage: git init [repository name]

 This command is used to start a new repository.

GitInit Command - Git Commands - Edureka

### git clone:-

Usage: git clone [url]

This command is used to obtain a repository from an existing URL.



### git add:-

U sage: git add [file]

This command adds a file to the staging area.

Git Add Command - Git Commands - Edureka

Usage: git add \*

This command adds one or more to the staging area.

Git Add Command - Git Commands - Edureka

### git commit:-

Usage: git commit -m “[ Type in the commit message]”

This command records or snapshots the file permanently in the version history.



Usage: git commit -a

This command commits any files you’ve added with the git add command and also commits any files you’ve changed since then.

Git Commit Command - Git Commands - Edureka

### git diff :-

Usage: git diff

This command shows the file differences which are not yet staged.



 Usage: git diff –staged :-

This command shows the differences between the files in the staging area and the latest version present.



Usage: git diff [first branch] [second branch]

This command shows the differences between the two branches mentioned.



**git reset:-**

Usage: git reset [file]

This command unstages the file, but it preserves the file contents.



Usage: **git reset [commit]**

This command undoes all the commits after the specified commit and preserves the changes locally.

Git Reset Command - Git Commands - Edureka

### git status:-

Usage: git status

This command lists all the files that have to be committed.



### git rm:-

Usage: git rm [file]

This command deletes the file from your working directory and stages the deletion.

Git Rm Command - Git Commands - Edureka

### git log:-

Usage: git log

This command is used to list the version history for the current branch.



Usage: git log –follow[file]

This command lists version history for a file, including the renaming of files also.



### git show:-

Usage: git show [commit]

This command shows the metadata and content changes of the specified commit.



### git tag:-

Usage: git tag [commitID]

This command is used to give tags to the specified commit.



### git branch:-

Usage: git branch

This command lists all the local branches in the current repository.

Git Branch Command - Git Commands - Edureka

Usage: git branch [branch name]

This command creates a new branch.

Git Branch Command - Git Commands - Edureka

Usage: git branch -d [branch name]

This command deletes the feature branch.

Git Branch Command - Git Commands - Edureka

### git checkout:-

Usage: git checkout [branch name]

This command is used to switch from one branch to another.

Git Checkout Command - Git Commands - Edureka

Usage: git checkout -b [branch name]

This command creates a new branch and also switches to it.

Git Checkout Command - Git Commands - Edureka

### git merge:-

Usage: git merge [branch name]

This command merges the specified branch’s history into the current branch.

Git Merge Command - Git Commands - Edureka

### git remote:-

Usage: git remote add [variable name] [Remote Server Link]

This command is used to connect your local repository to the remote server.

Git Remote Command - Git Commands - Edureka

### git push:-

Usage: git push [variable name] master

This command sends the committed changes of master branch to your remote repository.



Usage: git push [variable name] [branch]

This command sends the branch commits to your remote repository.



Usage: git push –all [variable name]

This command pushes all branches to your remote repository.



Usage: git push [variable name] :[branch name]

This command deletes a branch on your remote repository.



### git pull:-

Usage: git pull [Repository Link]

This command fetches and merges changes on the remote server to your working directory.



### git stash:-

Usage: git stash save

This command temporarily stores all the modified tracked files.

Git Stash Command - Git Commands - Edureka

Usage: git stash pop

This command restores the most recently stashed files.



Usage: git stash list

This command lists all stashed changesets.

Git Stash Command - Git Commands - Edureka

Usage: git stash drop

This command discards the most recently stashed changeset.

Git Stash Command - Git Commands - Edureka

**GIT Advance commands** :-

**Git rebase :-**

**Git rebase** is a command that integrates changes from one branch into another. It allows you to move or combine a sequence of commits to a new base commit. This process can help create a cleaner, more linear project history compared to merging.

### When to Use Rebase:

1. **Clean History**: To create a linear project history without merge commits.
2. **Feature Branch Integration**: To incorporate changes from the main branch into a feature branch before merging it back.

Usage:- **git checkout (Branch name )**

**git rebase (main repo)**