**GIT**

* Git is a DevOps tool used for source code management. It is a free open-source version control systems used to handle small to very large projects efficiently. It is used to tracking the changes in source code, enable multiple developers to work together or non-linear development.
* Git is a version control system used for tracking changes in computer files. It is generally used for source management in software development.
* Git is used to tracking changes in the source code.
* The distributed version control tool is used for source code management.
* It allows multiple developers to work together.
* It supports non-linear development through its thousands of parallel branches.
* **Features of Git:**
* Tracks history
* Free and open source
* Supports non-linear development.
* Create backups.
* Scalable
* Supports Collaboration
* Branching is easier.
* Distributed development
* **Working flow is divided into 3 stages.**
* Working directory- Modify files in your working directory.(create or edit the files) but not committed.
* Staging area(index): stage the files and add snapshots of them to your staging area.
* Before sending the files to staging area we must provide git add to the file.
* Get directory (Repository)-perform a commit that stores the snapshots permanently to your- Git directory. Checkout any existing version, make changes, stage them and commit.
* Repository has all the project related data.
* It contains the collection of files and also history of changes made to those files.

1.Local repository

2.Central Repository

3.Remote Repo.

* **Branch in Git:**

Branch in Git is used to keep your changes until they are ready. You can do your work on a branch while the main branch(master) remains stable. After you are done with your work, you can merge it with the main office.

**Commands:**

* **Git Config:**

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

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

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

* **Git init:**

Usage: git init [repository name]

This command is used to start a new repository.

* **Git Clone:**

Usage: git clone [URL]

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

* **Git adds.**

Usage: git add [file]

This command adds a file to the staging area.

* **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 have added with the git add command and commits any files you have changed since then.

* **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.

* **Git reset:**

Usage: git reset [file]

This command un stages 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.

* **Usage: git reset -hard [commit]**

This command discards all history and goes back to the specified commit.

* **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 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 remaining 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.

* **Usage: git branch [branch name]**

This command creates a new branch.

* **Usage: git branch -d[branch name]**

This command deletes the feature branch.

* **Git checkout**

Usage: git checkout [branch name]

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

* **Usage: git checkout -b [branch name]**

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

* **Git merge:**

Usage: git merge [branch name]

This command merges the specified branches history into the current branch.

* **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 Push**

Usage: git push [variable name] master

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

* **Usage: git push [varaiable 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.

* **Usage: git stash pop**

This command restores the most recently stashed files.

* **Usage: git stash list**

This command lists all stashed changesets.

* **Usage: git stash drop**

This command discards the most recently stashed changeset.