**GIT COMMANDS**

Git commands are instructions that interact with the Git version control system. These commands help developers manage source code, track changes, and collaborate effectively on projects. Git operates by recording snapshots of files (commits) and enabling teams to work simultaneously on code without overwriting each other’s changes.

**Git Commands and Their Purpose**

**Git Basic Commands**

* **git init** - Initializes a new Git repository in the current directory.
* **git add <file>** - Stages a specific file for the next commit.
* **git add .** - Stages all changes in the current directory for the next commit.
* **git commit -m** "<message>"- Commits staged changes with a descriptive message.
* **git status**- Displays the state of the working directory and staging area.
* **git config** - Sets user configurations like name and email for Git.

**Git Branching Commands**

* **git branch** - Lists all branches in the repository.
* **git branch <branch\_name>** - Creates a new branch.
* **git checkout <branch\_name>**- Switches to the specified branch.
* **git merge <branch\_name>** - Merges the specified branch into the current branch.
* **git pull** - Fetches changes from the remote repository and merges them into the current branch.
* **git push** - Uploads local commits to a remote repository.
* **git log** - Displays a history of commits in the repository.
* **git remote add <name> <url>** - Adds a remote repository reference.

**Git Advanced Commands**

* **git stash** - Temporarily saves changes not ready for commit.
* **git cherry-pick <commit\_hash>** - Applies a specific commit from another branch into the current branch.
* **git rebase <branch\_name>** - Reapplies commits on top of another base branch.
* **git diff** - Shows changes between the working directory and the staging area.