Here are most common scenarios and commonly used Git commands for DevOps and Cloud Engineers to manage their source code in Git repositories:

1. Cloning a Repository:
    - Command: **git clone <repository URL>**
2. Creating a New Branch:
    - Command: **git branch <branch name>**
    - Command (create and switch to the new branch): **git checkout -b <branch name>**
3. Switching Between Branches:
    - Command: **git checkout <branch name>**
4. Checking the Status of the Repository:
    - Command: **git status**
5. Adding Changes to the Staging Area:
    - Command (add a specific file): **git add <file name>**
    - Command (add all files): **git add.**
6. Committing Changes:
    - Command: **git commit -m "Commit message"**
7. Pushing Changes to a Remote Repository:
    - Command (push to the current branch): **git push**
    - Command (push to a specific branch): **git push origin <branch name>**
8. Pulling Changes from a Remote Repository:
    - Command: **git pull**
9. Merging Branches:
    - Command (switch to the branch you want to merge into): **git checkout <branch name>**
    - Command (merge a specific branch): **git merge <branch to merge>**
10. Resolving Merge Conflicts:
    - When conflicts occur during a merge, manually edit the conflicting files to resolve the conflicts. Then, add and commit the changes.
11. Fetching Changes from a Remote Repository:
    - **Command: git fetch**

12. Rebasing Changes:

    - Command (switch to the branch you want to rebase onto): **git checkout <branch name>**
    - Command (rebase a specific branch): **git rebase <branch to rebase>**

13. Tagging Releases:

    - Command: **git tag <tag name>**
    - Command (annotated tag with a message): **git tag -a <tag name> -m "Tag message"**

14. Viewing Commit History:

    - Command (show the commit history): **git log**
    - Command (show a summarized view of the commit history): **git log --oneline**

15. Undoing Changes:

    - Command (discard local changes to a file): **git checkout -- <file name>**
    - Command (undo the last commit and keep the changes locally): **git reset HEAD~ •**

      Command (undo the last commit and discard the changes): **git reset --hard HEAD~**

16. Viewing Differences:

    - Command (show changes between commits, branches, etc.): **git diff**
    - Command (show changes in staged files): **git diff --staged**

17. Stashing Changes:

    - Command (stash changes in a temporary area): **git stash**
    - Command (apply stashed changes): **git stash apply**
    - Command (list stashes): **git stash list**

18. Renaming or Moving Files:

    - Command (rename/move a file): **git mv <old file name> <new file name>**

19. Viewing Remote Repositories:

    - Command (list remote repositories): **git remote -v**

20. Adding Remote Repositories:

    - Command (add a remote repository): **git remote add <remote name> <repository URL>**

21. Removing Remote Repositories:

    - Command (remove a remote repository): **git remote remove <remote name>**

22. Viewing Branches:

- Command (list branches): **git branch**
- Command (list all branches, including remote branches): **git branch -a**

23. Deleting Branches:

- Command (delete a branch): **git branch -d <branch name>** •

  Command (forcefully delete a branch): **git branch -D <branch name>**

24. Checking Out a Specific Commit:

- Command (check out a specific commit): **git checkout <commit hash>**

25. Cherry-picking Commits:

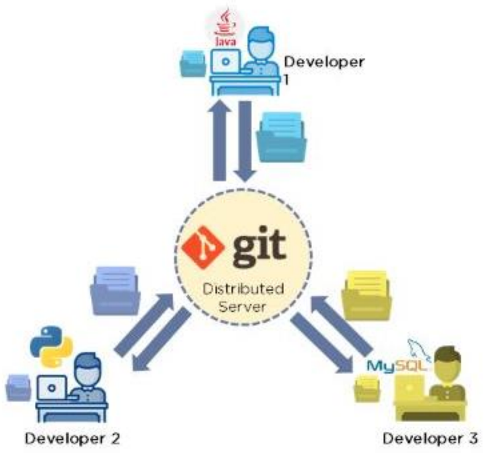
- Command (apply a commit from another branch): **git cherry-pick <commit hash>**

What is GIT & GITHUB?

➔Git is a distributed version control system **(DVCS)** that allows multiple people to collaborate on projects and track changes to their code. It was created by Linus Torvalds in 2005 and is widely used in software development for managing and tracking changes in source code during the development process. Git allows developers to work on different aspects of a project simultaneously and merge their changes without conflicts.

Here are some key concepts of Git:

- **Repository (Repo):** A repository is a collection of files and version history managed by Git. It can be local (on your computer) or remote (on a server).
- **Commit:** A commit is a snapshot of the changes made to the files in a repository at a specific point in time. Each commit has a unique identifier.
- **Branch:** A branch is a separate line of development within a repository. It allows developers to work on different features or fixes without affecting the main codebase.
- **Merge:** Merging is the process of combining changes from one branch into another. This is typically done to incorporate new features or bug fixes into the main branch.

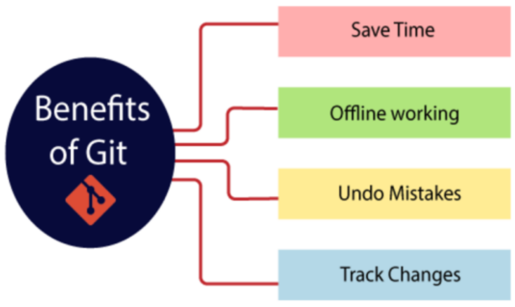


what is GitHub?

➔**GitHub,** on the other hand, is a web-based platform that provides hosting for Git repositories. It adds a web-based graphical interface, as well as features like issue tracking, collaboration tools, and pull requests. GitHub allows developers to store and manage their Git repositories online, making it easier to collaborate with others. It has become one of the most popular platforms for open-source development and collaborative software projects.

Key features of GitHub include:

- **Remote Repositories**: GitHub allows you to host your Git repositories on its servers, providing a centralized location for collaboration.
- **Pull Requests:** A pull request is a way to propose changes to a repository. It allows others to review the proposed changes and merge them into the main branch.
- **Pull Requests:** A pull request is a way to propose changes to a repository. It allows others to review the proposed changes and merge them into the main branch.
- **Issues:** GitHub provides a built-in issue tracking system, allowing developers to report and discuss bugs, feature requests, and other topics.
- **Wikis:** GitHub repositories can include wikis for documentation and project-related information.
- **Actions:** GitHub Actions enable you to automate workflows, such as building, testing, and deploying your code directly from the repository.



Author: **Jagadeesh Reddy**

Junior DevOps Engineer