1.Git Stash

Git stash is a feature that allows you to store changes that are not ready to be committed temporarily. You can use git stash save to stash changes and git stash apply to apply them back to the working directory.

2. git stash pop

Whenever we want files for commit from stash we should use this command.

3. git stash clear

By doing this, all files from stash area is been deleted.

**Git user configuration (First Step)**

git --version (to check git version)

git config --global user.name "your name here"

git config --global user.email "your email here"

git init

git clone http\_url

**# Connecting to the remote** **repository**

git remote add origin http\_url/ssh\_url

git pull origin master

git push origin master -> pushes your files to

github master branch

git push origin anyOtherBranch -> pushes any

other branch to github.

git log ; to see all your commits

## git diff

The ‘git diff’ command compares the staging area with the working directory and tells us the changes made. It compares the earlier information as well as the current modified information.

--=

#To remove only from the staging area

git rm –cached “ file name”

**Branching in Git**

create branch ->

git branch myBranch

or

git checkout -b myBranch -> make and switch to the

branch myBranch

Do the work in your branch. Then,

git checkout master ; to switch back to master branch

Now,  merge contents with your myBranch By:

git merge myBranch (writing in master branch)

This merger makes a new commit.

**Another way**

git rebase myBranch

This merges the branch with the master in a serial fashion. Now,

git push origin master

**To remove  or delete a file**

To remove.  a file from the  Git repository we use

git rm “file name”

# To remove only from the staging area

git rm –cached “ file name”

12) What is a 'conflict' in git?

A 'conflict' appears when the commit that has to be combined has some change in one place, and the current act also has a change at the same place. Git will not be easy to predict which change should take precedence.

### What is the difference between `git merge` and `git rebase`?

### Git merge integrates changes from one branch into another, creating a merge commit. Git rebase, on the other hand, moves the entire branch to begin from the tip of another branch, resulting in a linear history without merge commits.

### What is Git cherry-pick?

Git cherry-pick is a command used to apply a specific commit from one branch to another. It allows you to pick individual commits and apply them to the current branch.

### What is the purpose of the .gitignore file?

The .gitignore file specifies intentionally untracked files that Git should ignore. This is useful for excluding build artifacts, temporary files, and other files that shouldn’t be committed to the repository.

### 6. What is the difference between a Git clone and a Git fork?

Git clone creates a copy of an existing repository, including all branches and commit history, on the local machine. Git fork creates a copy of a repository on a remote server (like GitHub), allowing for independent development. You need to know the difference as a DevOps engineer because interviewers often ask this Git interview question.

### What is `git checkout`?

`git checkout` allows navigating between different branches

**How do you change the last commit?**

Use `git commit --amend` to modify the most recent commit. This can change the commit's message or include new changes.

### There are eight commits in the ‘****develop****’ branch, and one of those commits needs to be pushed into the ‘****release****’ branch. How would you approach this?

The best way to approach this is by using git cherry-pick. This is the act of picking a commit from a branch and applying it to another.

# Git Advanced Commands

### Branching & Rebasing & squashing

| **Command** | **Description** |
| --- | --- |
| git checkout [branch name] --orphan | Creates a new branch without the long repository's history. |
| git rebase [source branch] [target branch] | Fetch the current commits timeline from the remote branch and apply the new commits from the local branch on top of it |
| git rebase -i [commit hash]^ | Interactively rebase starting from the parent commit, where [commit hash]^ specifies the parent commit |
| git pull origin [source branch] --rebase | Pull changes from a remote repository using the rebase strategy. Default is merge. |
| git pull origin [source branch] --rebase --autostash | The same as above but performing automatically stash and pop your uncommitted changes when the branch is in a dirty state. |
| git cherry-pick [commit hash] | Apply the changes from any branche's commit to the target branch. |
| git config branch.[branch name].rebase true | Set a specific branch to always use the rebase strategy. |

### Fixing mistakes & destructive

| **Command** | **Description** |
| --- | --- |
| git reset --hard origin/[target branch] | Reset the local branch to the origin branch's state. **Caution:** your local changes will be lost. |
| git reset --soft HEAD~1 | Undo the last local commit. The --soft flag keeps the changes, if you don't need to keep them, you can use the --hard instead. **Caution**, the latter is a descructive command. |
| git commit --amend | Modify the most recent commit. Can be used to edit message or add more files to that commit. **Caution**, this changes the commit hash, so do this only while it was not yet pushed to a shared branch. |
| git revert HEAD~1 | Create a new commit undoing the last commit's changes. |
| git reflog | List all the git actions that were executed before. Useful to find a lost commit, then you can try to recover it. |
| git bisect start | Useful to find the buggy commit (that possible introduces some bug). After this, you use `git bisect [good |
| git branch --merged | egrep -v "(\*|master|dev)" | xargs git branch -d | Delete already merged branches but master and dev in this example. |
| git clean -df | Remove untracked files and directories from the working tree. Add n to the flags to list all files that would be removed. |