upGrad

Interview Questions for Version Control System

Q1.	What is the difference between Git and Github
Answer	Git is a version control system, a tool to manage your source code history. GitHub is a hosting service for Git repositories.

Q2.	What are the benefits of using version control system.
Reference	https://www.atlassian.com/git/tutorials/what-is-version-control
Answer	Version control systems are a category of software tools that help a software team manage changes to source code over time. Version control software keeps track of every modification to the code in a special kind of database. If a mistake is made, developers can turn back the clock and compare earlier versions of the code to help fix the mistake while minimizing disruption to all team members. Also, Version control systems allow you to compare files, identify differences, and merge the changes if needed prior to committing any code

Q3.	Why is it desirable to create an additional commit rather than amending an existing commit?
Reference	https://www.pcds.co.in/iqa/why-is-it-advisable-to-create-an-additional-commit-rather-than-amending-an-existing-commit.php
Answer	Git amend internally creates a new commit and replaces the old commit. If commits have already been pushed to central repository, it should not be used to modify the previous commits. It should be generally used for only amending the git comment. The amend operation destroys the state that was previously saved in a commit. If there is just the commit message being changed then that's not a problem. But if the contents are being amended then chances of eliminating something important remains more. Abusing "git commit- amend" can result in the growth of a small commit and acquire unrelated changes

Q4.	What is the purpose of branching in GIT.
Reference	https://www.atlassian.com/git/tutorials/using-branches#:~:text=In%20Git%2C%20branches%20are%20a,branch%20to%20encapsulate%20your%20changes.
Answer	In Git, branches are a part of your everyday development process. Git branches are effectively a pointer to a snapshot of your changes. When you want to add a new feature or fix a bug—no matter how big or how small—you spawn a new branch to encapsulate your change

Q5.	What is the upper limit on the heads in the Gits?
Answer	There is no upper limits on the numbers of heads

Q6.	How to resolve a conflict in git.
Reference	https://www.atlassian.com/git/tutorials/using-branches/merge-conflicts#:~:text=Git%20commands%20that%20can%20help%20resolve%20merge%20conflicts&text=Passing%20the%20%2D%2Dmerge%20argument,conflict%20between%20the%20merging%20branches.&text=diff%20helps%20find%20differences%20between,predicting%20and%20preventing%20merge%20conflicts.
Answer	Merge conflicts can be an intimidating experience. Luckily, Git offers powerful tools to help navigate and resolve conflicts. Git can handle most merges on its own with automatic merging features. A conflict arises when two separate branches have made edits to the same line in a file, or when a file has been deleted in one branch but edited in the other. Conflicts will most likely happen when working in a team environment. There are many tools to help resolve merge conflicts. Git has plenty of command line tools. git log, git reset, git status, git checkout, and git reset. In addition to the Git, many third-party tools offer streamlined merge conflict support features

Q7.	How will you know in Git if a branch has already been merged into master?
Reference	https://stackoverflow.com/questions/226976/how-can-i-know-if-a-branch-has-been-already-merged-into-master
Answer	git branch --merged master : lists branches merged into master

	git branch --merged : lists branches merged into HEAD (i.e. tip of current branch) git branch --no-merged : lists branches that have not been merged By default this applies to only the local branches. The -a flag will show both local and remote branches, and the -r flag shows only the remote branches.

Q8.	What is the difference between the ‘git diff ’and ‘git status’.
References	https://git-scm.com/docs/git-status https://git-scm.com/docs/git-diff
Answer	git-diff - Show changes between commits, commit and working tree, etc Show changes between the working tree and the index or a tree, changes between the index and a tree, changes between two trees, changes resulting from a merge, or changes between two files on disk. git-status - Show the working tree status Displays paths that have differences between the index file and the current HEAD commit, paths that have differences between the working tree and the index file, and paths in the working tree that are not tracked by Git.

Q9.	What is the difference between ‘git remote’ and ‘git clone’.
Reference	https://stackoverflow.com/questions/4855561/difference-between-git-remote-add-and-git-clone
Answer	git remote add just creates an entry in your git config that specifies a name for a particular URL. You must have an existing git repo to use this. git clone creates a new git repository by copying an existing one located at the URI you specify

Q10.	Explain the advantages of forking workflow.
Reference	https://www.atlassian.com/git/tutorials/comparing-workflows/forking-workflow#:~:text=The%20main%20advantage%20of%20the,push%20to%20the%20official%20repository.
Answer	The main advantage of the Forking Workflow is that contributions can be integrated without the need for everybody to push to a single central repository. Developers push to their own server-side repositories, and only the project maintainer can push to the official repository. This allows the maintainer to accept commits from any developer without giving them write access to the official codebase.

Q11.	What do you understand by git collaboration. Explain any one workflow.
Reference	https://www.atlassian.com/git/tutorials/comparing-workflows/forking-workflow
Answer	Collaboration helps you work with the people around you and produce something even better than before. You can add features or improvise some of the features in someone else's project who is sitting miles away from you. Explanation of Forking Workflow The Forking Workflow typically follows a branching model based on the Gitflow Workflow. This means that complete feature branches will be purposed for merge into the original project maintainer's repository. The result is a distributed workflow that provides a flexible way for large, organic teams (including untrusted third-parties) to collaborate securely. This also makes it an ideal workflow for open source projects. 1. To contribute to someone's project we need to follow the steps below- 2. To contribute to someone's project we first need to **fork** it. Forking will get you that project on your github account. **3.** Now to bring that project to your local system you need to **clone** it. 4. Next you will make modifications and push the changes back to your Github account. **5.** Now to inform the owner of the project about new changes that you made you will create a **pull request** 6. If your changes are good, the owner will merge your changes.

Q12.	What is the difference between fork, and clone
Answer	Forking is a concept while cloning is a process. Forking is just containing a separate copy of the repository and there is no command involved. Cloning is done through the command ‘git clone‘ and it is a process of receiving all the code files to the local machine.