AWS & DevOps

Interview Questions

ACHYUTH REDDY S

Follow For More Update

www.linkedin.com/in/achyuth-reddy-s-245632246

GIT COMMANDS

GIT BASICS

git init:

Create empty Git repo in specified directory. Run with no arguments to initialize the current directory as a git repository.

git clone <repo url>:

Clone repo located at *<repo>* onto local machine. Original repo can be located on the local filesystem or on a remote machine via *HTTP* or *SSH*.

git config user.name <name>:

Define author name to be used for all commits in current repo. Devs commonly use --*global* flag to set config options for current user.

git add <directory>:

Stage all changes in *<directory>* for the next commit. Replace *<directory>* with a *<file>* to change a specific file.

git commit -m "<message>":

Commit the staged snapshot, but instead of launching a text editor, use *<message>* as the commit message.

git status:

List which files are staged, unstaged, and untracked.

git log:

Display the entire commit history using the default format. For customization see additional options.

git diff:

Show unstaged changes between your index and working directory.

UNDOING CHANGES

git revert <commit>:

Create new commit that undoes all of the changes made in *<commit>*, then apply it to the current branch.

git reset <file>:

Remove *<file>* from the staging area, but leave the working directory unchanged. This unstages a file without overwriting any changes.

git clean -n:

Shows which files would be removed from working directory. Use the *-f* flag in place of the *-n* flag to execute the clean.

REWRITING GIT HISTORY

git commit –amend:

Replace the last commit with the staged changes and last commit combined. Use with nothing staged to edit the last commit's message.

git rebase <base>:

Rebase the current branch onto *<base>*. *<base>* can be a commit ID, branch name, a tag, or a relative reference to *HEAD.*

git reflog:

Show a log of changes to the local repository's *HEAD*. Add --*relative-date* flag to show date info or --*all* to show all refs.

GIT BRANCHES

git branch:

List all of the branches in your repo. Add a *<branch>* argument to create a new branch with the name *<branch>*.

git checkout -b <branch>:

Create and check out a new branch named *<branch>*. Drop the *-b* flag to checkout an existing branch.

git merge <branch>:

Merge *<branch>* into the current branch.

REMOTE REPOSITORIES

git remote add <name> <url>:

Create a new connection to a remote repo. After adding a remote, you can use *<name>* as a shortcut for *<url>* in other commands.

git fetch <remote> <branch>:

Fetches a specific *<branch>*, from the repo. Leave off *<branch>* to fetch all remote refs.

git pull <remote>:

Fetch the specified remote's copy of current branch and immediately merge it into the local copy.

git push <remote> <branch>:

Push the branch to *<remote>*, along with necessary commits and objects. Creates named branch in the remote repo if it doesn't exist.

GIT CONFIG

git config --global user.name <name>:

Define the author name to be used for all commits by the current user.

git config --global user.email <email>:

Define the author email to be used for all commits by the current user.

git config --global alias. <alias-name> <git-command>:

Create shortcut for a Git command. E.g. *alias.glog "log --graph --oneline"* will set *"git glog"* equivalent to *"git log --graph --oneline.*

git config --system core.editor <editor>:

Set text editor used by commands for all users on the machine. *<editor>* arg should be the command that launches the desired editor (e.g., vi).

git config --global –edit:

Open the global configuration file in a text editor for manual editing.

GIT LOG

git log -<limit>:

Limit number of commits by *<limit>*. E.g. *"git log -5"* will limit to 5 commits.

git log –oneline:

Condense each commit to a single line.

git log -p:

Display the full diff of each commit.

git log –stat:

Include which files were altered and the relative number of lines that were added or deleted from each of them.

git log --author= ”<pattern>”:

Search for commits by a particular author.

git log --grep=”<pattern>”:

Search for commits with a commit message that matches *<pattern>*

git log <since>..<until>:

Show commits that occur between *<since>* and *<until>*. Args can be a commit ID, branch name, *HEAD*, or any other kind of revision reference.

git log -- <file>:

Only display commits that have the specified file.

git log --graph –decorate:

--graph flag draws a text based graph of commits on left side of commit msgs. *--decorate* adds names of branches or tags of commits shown.

GIT DIFF

git diff HEAD:

Show difference between working directory and last commit.

git diff –cached:

Show difference between staged changes and last commit.

GIT RESET

git reset:

Reset staging area to match most recent commit, but leave the working directory unchanged.

git reset –hard:

Reset staging area and working directory to match most recent commit and **overwrites all changes** in the working directory.

git reset <commit>:

Move the current branch tip backward to *<commit>*, reset the staging area to match, but leave the working directory alone.

git reset --hard <commit>:

Same as previous, but resets both the staging area & working directory to match. **Deletes** uncommitted changes, and **all commits after** *<commit>.*

GIT REBASE

git rebase -i <base>:

Interactively rebase current branch onto *<base>*. Launches editor to enter commands for how each commit will be transferred to the new base.

GIT PULL

git pull --rebase <remote>:

Fetch the remote's copy of current branch and rebases it into the local copy. Uses git rebase instead of merge to integrate the branches.

GIT PUSH

git push <remote> --force:

Forces the *git push* even if it results in a non-fast-forward merge. Do not use the *--force* flag unless you're absolutely sure you know what you're doing.

git push <remote> --all:

Push all of your local branches to the specified remote.

git push <remote> --tags:

Tags aren't automatically pushed when you push a branch or use the *--all* flag. The *--tags* flag sends all of your local tags to the remote repo.