GITHUB

Github is a distributed version control system.

Distributed means you can access from anywhere

Version has multiple versions or copies of the same files

Github will track whatever changes will be made

You can revert back to any point in time

Other similar tools are:

Gitlab

Gitserver

Bitbucket

SVN is not a distributed version control system.

With SVN you must log in to a dashboard

You can clone your GitHub repository to any number of remote systems.

There are 2 types of repositories

Public - anyone can access. Only read only access

Private - only owner access

rpm -qa |grep git - Verify if the package git is installed or not,(has the word git anywhere in it)

rpm -qa |grep ^git – o]nly if it starts with the keyword git

Central repository - whatever you have in Github.com

Local repository - where you cloned your repository

git clone <https://github.com/ekekaws/devops.git> - clones a repository into a new directory(local repository)

Cloning using ssh

<https://www.youtube.com/watch?v=CLkl40efxuA>

ssh-keygen -t rsa -f "name for key"

git add . - send an update to a staging area- intermediate area between local repository and central repository

git commit - you’re committing the message - describing your change as to what you have done

git push origin master - uploading your changes to the central repository

git log show the log - the log produced a code (7286e588186fd23addb6eea54e1e33649439c042)

git show - reveals the code for (d366cc38c7d290897f199220a265af77b6da7610)

git --help -> shows you all commands

git add --help -> it's just like a MAN command

git log -> will show you git log commit

git log --stat -> will show who created or updated the files

Git branch branchname >> to create a branch

Git checkout branchname >> to switch to new branch

Git token – Setting/DeveloperSettings/generate classic token ( THIS CAN BE USED AS PASSWORD TO PUSH in to repo when from EC2 instance

Change this in the git config –-global --edit to view changes made by a particular user instead of ec2-user

# Please adapt and uncomment the following lines:

# name = Dheeraj-Nagireddy

# email = nagiredn@oregonstate.edu

In Linux, if the file starts with '.' is a hidden file or hidden folder

How to avoid username and password prompt from anywhere

-> create a file called .netrc under the user home directory with below content

machine github.com

login "github username"

Password "github password"

-> save it

Pull remote branch git switch “branch name” or git pull origin “branch name”

Switch to remote branch git switch “branch name”

Pull from remote branch

Branching Strategy:

-> Repository name is devops [Any project name]

-> Already we have branches called Master, development, QA & production

-> First developer will get a request [ New functionality or bug fixing] through a ticket [Jira, Remedy, servicenow..]

-> Now developer will create a future branch from development branch

-> Developer will start working on future branch only and he will complete the code

-> Once all the test cases are successful, then it will be merged into development branch

-> On development server, the development code will be checked out and the test cases will be tested here

-> Once everything is successful, the code will be merged into QA branch

-> On QA server, the code will be checked out, test cases will be run and it is successful it will be merged into production

-> On Sandbox or UAT server, the code will be checked out from UAT branch, test cases will be checked here

-> Finally, the management will review the application and create a release tag and deploy the code into production server

1. please provide only amar user commits

--> git log --author=amar

----------------------------------------------------------

2. what is the max size of the file, i can push it into git hub

100MB

GitHub blocks pushes that exceed 100 MB.

GitHub limits the size of files allowed in repositories. To track files beyond this limit, you can use

Git Large File Storage.

Using Git LFS, you can store files up to:

Product Maximum file size

GitHub Free 2 GB

GitHub Pro 2 GB

GitHub Team 4 GB

GitHub Enterprise Cloud 5 GB

------------------------------------------------------------------

3. how to push empty directory into git hub

Create an empty folder with the mkdir command

--> mkdir empty-directory

---> cd empty-directory

Create the .gitkeep file

--> touch .gitkeep

--> git add .

--> git commit -m "Commit empty folder in Git with gitkeep"

--> git push

-------------------------------------------------------------------------

4. how to ignore a particular file while commiting

create two files

--> touch file2 file3

create .gitignore file

--> touch .gitignore

here we are ignoring file3 while commiting. so add particular file name into .gitignore file

--> vi .gitignore

file3

save it and do

--> git status

the output shows only file2 and .gitignore to commit

and after commiting u can check in git hub gobal repository that file3 is not added

--------------------------------------------------------------------------------

5. how to get only last two days commit

--> git log --raw --since=2.days

or

--> git log --name-status --since="2 days ago"

or if u want to display only files/folders names

--> git log --pretty=format: --name-only --since="2 days ago"

if some files duplicate in multiple commits, we can use pipe to filter it

--> git log --pretty=format: --name-only --since="2 days ago" | sort | uniq

-----------------------------------------------------------------------------------

6. how to get a particular date commit say 05/01/2022

--> git log --after="2022-01-05 00:00" --before="2022-01-05 23:59"

-----------------------------------------------------------------------------------

7. how to get commit in b/w days (4 jan to 6 jan)

--> git log --after="2022-01-05" --until="2022-01-06"

this excludes commits on 4 jan and gives commits ong 5th and 6th jan

--> git log --after="2022-01-04" --until="2022-01-06"

this gives commits of 4,5,6 jan

-------------------------------------------------------------------------------------

8. i have a file, find how many commits done on that file

--> git log --follow -- <filename\_here>

if u want to find all commits

--> git rev-list --all

------------------------------------------------------------------------------------------

9. find in particulat commit, how many files are modified

--> git log --stat

this gives in all commits how many are modified

or

--> git log

u will get commit id

git diff-tree -r <commit id>

u will see output only in that particular commit id

---------------------------------------------------------------------------------------

10. how to update commit messsage

Rewriting the most recent commit message:

------------------------------------------

--> git commit --amend -m "New commit message."

and then check

--> git log

the latest commit gets "New commit message."

or

--> git commit --amend

press enter and then it opens vi file, there u can see old commit message u can edit it there

and check git log

The new commit and message will appear on GitHub.com the next time you push.

we have to force push

--> git push --force

Amending older or multiple commit messages:

-----------------------------------------

for example if u want to amend last 2 commits

--> git rebase -i HEAD~2

it opens a file and the file looks like this

pick c2b2bd2 old commit message

pick 902e8d0 new

# Rebase f225acf..902e8d0 onto f225acf (2 commands)

#

# Commands:

# p, pick <commit> = use commit

# r, reword <commit> = use commit, but edit the commit message

# e, edit <commit> = use commit, but stop for amending

# s, squash <commit> = use commit, but meld into previous commit

# f, fixup [-C | -c] <commit> = like "squash" but keep only the previous

# commit's log message, unless -C is used, in which case

# keep only this commit's message; -c is same as -C but

# opens the editor

# x, exec <command> = run command (the rest of the line) using shell

# b, break = stop here (continue rebase later with 'git rebase --continue')

# d, drop <commit> = remove commit

# l, label <label> = label current HEAD with a name

# t, reset <label> = reset HEAD to a label

# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]

Replace pick with reword before each commit message you want to change.

reword c2b2bd2 old commit message

reword 902e8d0 new

Save and close the commit list file.

In each resulting commit file, type the new commit message, save the file, and close it.

now it automatically opens the vi file for each of it one after the other

first u get

old commit message (vi editor file change the message here and save it)

next u get

new (vi editor file change message and save it)

When you're ready to push your changes to GitHub, use the push --force command or

( push -f )to force push over the old commit.

---------------------------------------------------------------------------

11. can we update the new/modifed file with same/earlier commit

firstly i checked how many commits i have

--> git rev-list --all |wc -l

output: 14

i already have file4 and i opened and written some data in it

also created new file

--> touch file6

--> git add .

--> git commit --amend --no-edit ( i used no edit not to change commit message)

--> git push -f

and again i checked how many commits i have

--> git rev-list --all |wc -l

i got output 14

-------------------------------------------------------

12. how to revert the file from staging area

suppose i created two files

--> touch file7 file8

--> git add .

--> git status

o/p:

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: file7

new file: file8

but now i don't want file8 to commit

to revert it

--> git reset HEAD -- file8

--> git status

o/p:

Changes to be committed:

(use "git restore --staged <file>..." to unstage)

new file: file7

Untracked files:

(use "git add <file>..." to include in what will be committed)

file8

and now when we do git commit the file8 will not upload into central git hub repository

-------------------------------------------------------------------

13. how to revert the file after git commit command

--> touch file8

--> git add .

--> git commit -m "new file"

now to revert the file after commit command

--> git log

copy the latest commit id ( or commit id of "new file")

--> git revert <commitid>

opens a file and save it

when u do git push now the file8 will not commit

-------------------------------------------------------------------------

14. what is the equivalent service of git in aws

AWS code commit

Azure DevOps Server

Helix Core

Subversion

Rational ClearCase

Mercurial

Micro Focus AccuRev

CVS

---------------------------------------------------------------------------

15. how to revert a particular commit id say as commit 5

--> git revert <commitd of 5>

--------------------------------------------------------------------

16. how to revert upto 5th commit

--> git revert --no-commit HEAD~5..

--> git commit --no-edit

--> git push

------------------------------------------------------------------------------

17. how to make push changes to github with auto login from client

we can generate ssh-keygen and copy idrsa\_pub into github>settings>ssh and cpg keys> new ssh key

18.Delete branch and push to repo

// delete branch locally

git branch -d localBranchName

// delete branch remotely

git push origin --delete remoteBranchName

PUSHING INTO WRONG BRANCH

git checkout wrong\_branch

git revert commitsha1

git revert commitsha2

git checkout right\_branch

git cherry-pick commitsha1

git cherry-pick commitsha2

Graphical user interface, text, application, email

Description automatically generated

TO only get particular commit changes in another branch

>>Switch to branch you want the changes on

git cherry-pick “commit id”

Text

Description automatically generatedMAIN BRANCH

DEV BRANCH

Now d4 is available in main after the cherry-pick