git configuration on Linux
1. create ssh key with your mail id
   ex. ssh-keygen -t rsa -C "sagar.jagtap@calsoftinc.com"
2. git config --global user.name "sagar.jagtap"
   git config --global user.email "sagar.jagtap@calsoftinc.com"
3. cat /.ssh / id_rsa.pub
paste this key in ur git repository

Git Commands
1. Git diff

New create repo	1.git clone https://git.druvwa.org/sagarj/phoeniwx.git 2. Follow below commands Git> git remote add main https://git.druvwa.org/phoeniwxdev/phoeniwx.git Git> git fetch main Git> git checkout -b <UR_branch_name> main/<branch_name> git checkout -b <branch_name> main/f_AHV Git> git push -u origin <UR_branc_name> branch bame =>f_AHV 3.
Git revert file	git checkout -- ../roboCloud/vmware/vmware_api.py
Remove file forcefully	git clean -fdx
Git log with patch	Git log -p
Add to staging	Git add -p
Show in stagging diff	Git add --staged
Git --amend	The command git commit --amend will overwrite the previous commit with what is already in the **staging** area.
Git cherry-pick –n	-n option , doesn't commit to ur branch, it kept in staging areas. https://www.youtube.com/watch?v=wIY824wWpu4 For Basic understanding => https://www.youtube.com/watch?v=yw-qkJs4py0
Git stash	From current branch ,Keep ur current changes in temporary area of git , like buffer.
Git stash apply	From git temporary area to your current branch .
Git revert	Let's say we've made a mistake in our latest commit to a public branch. Which of the following commands is the best option for fixing our mistake? if u did wrong commit in branch, used revert to rollback.
use the commit ID at the end of the git revert command	If we want to rollback a commit on a public branch that wasn't the most recent on2qe using the revert command, what must we do?

What does the command git commit --amend do?	Awesome! The command git commit --amend will overwrite the previous commit with what is already in the staging area.

Command	**Explanation & Link**
git commit -a	Stages files automatically
git log -p	Produces patch text
git show	Shows various objects
git diff	Is similar to the Linux `diff` command, and can show the differences in various commits
git diff --staged	An alias to --cached, this will show all staged files compared to the named commit
git add -p	Allows a user to interactively review patches to add to the current commit
git mv	Similar to the Linux `mv` command, this moves a file
git rm	Similar to the Linux `rm` command, this deletes, or removes a file

There are many useful git cheatsheets online as well. Please take some time to research and study a few, such as this one.

.gitignore files https://gist.github.com/octocat/9257657

.gitignore files are used to tell the git tool to intentionally ignore some files in a given Git repository. For example, this can be useful for configuration files or metadata files that a user may not want to check into the master branch. Check out more at: https://git-scm.com/docs/gitignore.

A few common examples of file patterns to exclude can be found here.

git checkout is effectively used to switch branches.

git reset basically resets the repo, throwing away some changes. It's somewhat difficult to understand, so reading the examples in the documentation may be a bit more useful.

There are some other useful articles online, which discuss more aggressive approaches to resetting the repo.

git commit --amend is used to make changes to commits after-the-fact, which can be useful for making notes about a given commit.

git revert makes a new commit which effectively rolls back a previous commit. It's a bit like an undo command.

There are a few ways you can rollback commits in Git.

There are some interesting considerations about how git object data is stored, such as the usage of sha-1. Feel free to read more here:

- https://en.wikipedia.org/wiki/SHA-1
- https://github.blog/2017-03-20-sha-1-collision-detection-on-github-com/

//release specific build

git-lfs clone -b release-11.0.5.0 git@github.ibm.com:Voldemort/nwps.git

//ips-helm
git clone -b develop git@github.ibm.com:privatecloud-ap/ipws-helm

For reference (if needed):
1.   git clone -b dev_develop git@github.ibm.com:Voldemort/npws   : This will clone git branch into pwd.
2.   git checkout -b <your_branch_name>   : format I use is ibmid_<name>
e.g. abhijog_goclient_backend
3.   git push origin <your_branch_name>   : publish your branch to remote repo. This also sets remote branch for â€œgit pushâ€
4.   make your changes.
5.   git status   : this should show your changed files. You can check diff using git diff
6.   git add
6.   git commit -m â€œ<some comment>â€   : commit staged changes to local repo
7.   git push   : git push origin sjagtap3_ACCESS_SECRET_KEYS_update
8.   from web, compare dev_go_connector and your_branch_name
9.   generate PR, mention JIRA epic link in the comment. I have added you in all epics

git push origin sjagtap3_contents<ur branch>
    1.   git commit -am "review changes added"

Delete file	`git rm 'file name'`
	`git commit -m'message'`
	`git push -u origin <ur branch>`

Revert file	Git log <filename> `git checkout f08a63ff4fa7b8479f8c698e5998ee1afcac3a4e` <filename>
Checkout file from from another branch	`git checkout <branch_name> -- <paths>`
Git with whitspaceshown	git diff --color | less -R
Git update	Git pull origin <branch name>
git branch -d delete	Delete from local branch
git-lfs clone -b **pod_connector** git@github.ibm.com:Voldemort/nps.git	Clone code from another branch.
git branch JSagar_B1798	
git checkout <branch name>	Switch from master branch to your branch
git checkout **-b** Bug-2376-GXT-Jsagar	is create new branch
git branch JSagar_B1798	
git checkout <branch name>	Switch from master branch to your branch
