**Git Commands**

1. **git init** it will create a folder called **.git**

**2. git add .** it will add files to staging area

3. **git commit -m”commit message”** file will move from staging area to commit area and it

can assign commit id

4. **git config --global user.email"kattasandeep6191@gmail.com"**

**git config --global user.name"sanny"** it will add user and username to git projects

5. **git remote add origin https://github.com/sandeepkatta01/**deepu it will add remote repo to our

Local

6. **git push origin master** it will push our files to remote repo

7. **git branch feature** it will create a feature branch, branch which is used to fix the bugs in code and

later we push to master

8. **git checkout master** it will switch to master from feature branch

9. **git branch -l** it will show all the branches

10.**git pull origin master** it will pull repo from remote

11. **git clone “url”** it will download entire repo from remote

12. **git branch -d feature** it will delete the feature branch in our local

13. **git push origin -d** it will delete the feature branch in remote repo

14. **git tag -a v1.0 - m “this is my new release”** it will used to put the version for new release of

code

15. **git tag -d v1.0** it will delete the version in our local repo

16. **git push origin :refs/tags/v1.0** it will delete the tag in central repo

17. **git show v1.0** it will show the entire details

18. **git checkout sample.txt** it will undo the local changes in sample .txt file when it is not committed

file

19. **git reset --soft HEAD~1** file comes from history to commit area and last commit id get deleted

20. **git reset --hard HEAD~1** it will delete the last changes in file and also delete the commit things

21. **git stash** it will save our file local without committing when we move from proj1 to proj2

22. **git stash pop** again we want to work on proj1 where we left in middle we can continue.

23. **git commit --amend** to modify the latest commit messages

24. **git rebase -i HEAD~2** to modify the commit messages for previous commits.

25. **git cherry-pick commit id** if we keep mistakenly commit message applied to one branch

Then we get that commit message to original branch we use above command

26. **git rebase feature** to get history of the feature branch to master branch.

27. **git merge --squash** feature to avoid merge conflicts, but commit messages don’t get to master