*Git Commands*

# git status :

This command is used to see status of file. If any new file is added it will show here. If any file is modified then that file will also be seen here.

# git clone [url]

This command is used to download a copy of source code from the repository.

# git log

This command is used to list the version history of the current branch.

# git log --follow <file or folder name>

This command is used to history of a specific or a folder.

# git commit –m “commit message”

This command is used to commit changes we have made into the local repository. It is mandatary to give –m, in order to give the commit message without which we have to do in vi editor but its compulsory.

This command records snapshot of the version permanently.

# git commit –a

This command will commit all changes added via git add to staging area and also commits the changed file as well. In short it commits modified files as well as newly staged files.

# git add –<filename>

This command is used to add a file (single file) to the staging area.

# git add\*

This command adds one or more file to the staging area

# git add –A

This command will add all the files in our workspace to the staging area or we can say it will index the files. After this only the files can be committed. –A has to be added that too capital A so that all the files which are to be indexed can be added to the staging area.

# git push –u origin master

This command is used to push the contents of local repository to the remote repository. But please note that only the changes will be stored.

# git push --all origin

This command pushes all your branches to the remote repository

# git pull

This command will pull all the latest changes from the remote directory and merges into the local directory you are working.

# git config --global user.name “vikas1265”

This command is used to set the username to be used with your comments.

# git config --global user.email gautamv725@gmail.com

This command is sued to set the email to be used with your comments.

# git init

This command is used to start a new repository.

# git checkout –b “BranchName”

This command will create a new branch in the local and also switch to it.

# git branch branch\_name

This command will create a new branch in the local.

# git branch

This command will show the current branch name in use.

# git branch –d branchName

This command is used to delete the branch.

# git checkout “BranchName”

This command is used to switch to the new branch.

# git merge testingBranch

This command will merge all changes done in testingBranch into the master branch.

# git remote add origin [url]

This command is used to connect the local repository the remote repository. Here we are basically specifying the origin variable as the url of the remote repository and then later use that url.

# git stash save

This command will temporarily remove all the local changes we have done to our machine and save somewhere. It will put our local machine into the last pull stage. We can later on pop these changes.

# git stash pop

This command will bring back those removed changes in stash back.

# git stash list

This command will show the stashed list items.

# git stash drop

This command drops the recently stashed changes.

# git show

This command will show the changes made in the latest commit and metadata.

$ git show

commit c3537c81ae6ee1793e9b4d5a29f7f1a6905a90b6 (HEAD -> master, origin/testingBranch, origin/master, origin/HEAD)

Author: vikas1265 <gautamv725@gmail.com>

Date: Sun Sep 1 19:14:14 2019 +0530

test data

**diff --git a/testFile.txt b/testFile.txt**

**new file mode 100644**

**index 0000000..016e97f**

**--- /dev/null**

**+++ b/testFile.txt**

@@ -0,0 +1 @@

+this is test data

# git diff

This command will show all the difference between files which are not yet staged and current version.

VIKAS GAUTAM@Vikas MINGW64 /c/Git/CodingData (master)

$ git diff

**diff --git a/sampleFile.txt b/sampleFile.txt**

**index 13cd291..f833e5b 100644**

**--- a/sampleFile.txt**

**+++ b/sampleFile.txt**

@@ -1 +1 @@

-this is sample data

\ No newline at end of file

+this is sample data, which has been modified

\ No newline at end of file

# git diff –staged

This command will show all the difference between staged files and current version of the file.

# git diff <branch1> <branch 2>

This command is used to see the difference between the two branches.

VIKAS GAUTAM@Vikas MINGW64 /c/Git/CodingData (master)

$ git diff --staged

**diff --git a/sampleFile.txt b/sampleFile.txt**

**index 13cd291..f833e5b 100644**

**--- a/sampleFile.txt**

**+++ b/sampleFile.txt**

@@ -1 +1 @@

-this is sample data

\ No newline at end of file

+this is sample data, which has been modified

\ No newline at end of file

**diff --git a/testFile.txt b/testFile.txt**

**deleted file mode 100644**

**index 016e97f..0000000**

**--- a/testFile.txt**

**+++ /dev/null**

@@ -1 +0,0 @@

-this is test data

# git diff <filename>

This command will show the difference in the file that is current commit and unstaged file.

# git rm-<file or folder name>

This command will remove a file or folder from local and stages it for deletion. Once committed and pushed it will be removed from the remote repository also.

# git reset

This command will unstage a file but preserve its contents.

# git reset [commit name]

This command will undo all the commits after the specified commit.

# git remote show origin

This command will show all the branches in remote repository along with current HEAD that is the current branch or master branch.

VIKAS GAUTAM@Vikas MINGW64 /c/Git/CodingData (master)

$ git remote show origin

\* remote origin

Fetch URL: https://vikas1265@github.com/vikas1265/CodingData.git

Push URL: https://vikas1265@github.com/vikas1265/CodingData.git

HEAD branch: master

Remote branches:

master tracked

testingBranch tracked

Local branch configured for 'git pull':

master merges with remote master

Local ref configured for 'git push':

master pushes to master (fast-forwardable)

# git push origin --delete branchTest

This command will delete the branch from the remote repository.

# How to resolve git conflicts

* Sometimes in our branch we could have done same change and committed and in same branch someone else could have also done an update and pushed.
* In this case when we are trying to do a git pull then we will get conflict.
* This conflict can be auto merged by by the git. If its unable to auto merge then we need to explicitly remove the conflict.
* We need to delete conflict markers <<<<<<<, =======, >>>>>>> and make the changes you want in the final merge.
* After this do a **git add -A** and then commit the changes. In this way the conflict will get resolved.

# Git Conflict Scenario-Branch updated and trying to push our change

* Consider a scenario where someone has updated a file in the remote repository.
* Now the change done in the remote repository does not exist. In that case if we are trying to push our change from the local into the remote repository then we are going to get error.
* In order to resolve this issue we need to make sure that we are doing a git pull and thereafter we are doing the git push then the error will not come.
* Please consider the below screenshot for the reference

