To make your local repository to point to the git repository, use command

**git remote set-url origin "<URL here>"**

Use **git config -l** :- to see if the remote url has been changed to point to the new repository

If the local repository is empty, add a file using :

**touch README**

**vi README**

enter your content

save your file

**git add README**

**git commit -m '<commit comment here>'**

**git push origin "<branch name here>"**

by default, the branch is "**master**"

If the local repository is not empty and containing files or folders,

then we can either select the folder or file particulary or if we want to commit the entire folder comments, then

**git add -all**

**-a**

**--all**

Tell the command to automatically stage files that have been modified and deleted, but new files you have not told Git about are not affected.

When recording your own work, the contents of modified files in your working tree are temporarily stored to a staging area called the "index" with git add.

And similarly, all the indexed files will be committed using -

**git commit -m "<commit message here>"**

It may happen that there is no new changes in your file and you have added an already committed file using git add.

In that case, when you will do a git commit, it will show you a message as:

Everything upto date, nothing new to add.

This shows that the changes have already been committed and git makes sure that only new changes gets committed to the repository to prevent unnecessary commits to the repository.

To see if you have made some changes or git commit is actually committing something to the repository,

use:

**git status.**

It will show you what files you have changed and staged for commit.

After you have committed your staged files, you need to push those files to the repository else the changes wont be reflected in you git repository.

**git push origin "<branch name>"**

By default, git pushes all the committed files to the branch “**master**”

However, if you want to create a new branch, you can use command:

**git checkout -b "<new branch name>"**

- git will acknowledge that a new branch has been created with the message that - switched to a new branch 'branch name';

All the files are pushed to this branch in the same manner.

But, in order for the git repository to keep track of the newly created branch and files you have committed inside it, it is necessary to put the name of the repository after a commit.

**git add "file name"**

**git commit -m "<commit message>"**

**git push origin "<branch name>"**

If you want to switch to another repository than the one in which you are currently in,

use command:

**git checkout "<branch name>"**

if the branch already exists.