**Assignment No. \_1\_**

**Student Name :** Ashlesha Vikas Joshi **MIS:** 111903022

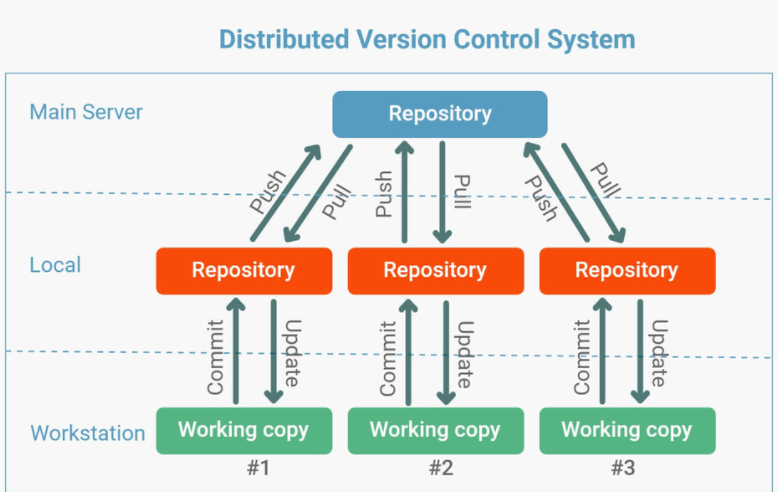
**Course Teacher :** Rohini Sarode Ma’am

**Course Coordinator :** Rohini Sarode Ma’am

**Aim: Introduction to Git and various Git commands**.

INTRODUCTION TO GIT AS A VERSION CONTROL SYSTEM

1. Version control is a practice by which one can track, manage and record changes to a project code or software code over time.
2. Version control is also known as source control.
3. There are basically three types of Version control systems like Local Version control system, Centralized Version control system and Distributed Version Control System.
4. Version control systems are helpful when many people are working on a particular project like when it is a team project. It is easier to track changes made to the source code of the project by the members of the team.
5. **Git** is a Version control system which keeps track/record of all the files and various projects as they are modified by various contributors present.
6. **Git** is a **Distributed Version Control system** which means that every collaborator will have the entire repository on his/her local machine. All the information about commits, tags, branches, files, revision history etc will be present on every collaborator’s local machine.



In the above picture, #1, #2 and #3 are the three collaborators and working copy is the same copy of entire repository on the respective local machines of each collaborator. This is the way a distributed control system like Git works.

INSTALLATION OF GITHUB

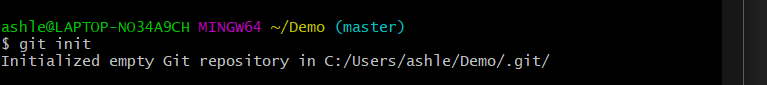
1. You can install GitHub Desktop from the link <https://desktop.github.com/> and set-up your Github account using the steps provided on <https://docs.github.com/en/desktop/installing-and-configuring-github-desktop/installing-and-authenticating-to-github-desktop/setting-up-github-desktop>
2. If you don’t wish to install Github Desktop, you can visit the link <https://github.com/join>

And create your GitHub account by entering the required information.

INTRODUCTION TO VARIOUS GIT COMMANDS

1. **git init**

The git init command is used for creating a new/empty git repository. For starting a project, this is basically a first command you should run. It will create .git directory in the current working directory which contains the the Git metadata (data about data) for the new repository you created.



This command can be used with various options like-

**--bare** – It will create a bare repository.

--**quiet** – It will display some part of output like errors or warnings, not all.

--**template**=  **-** It is used to indicate directory from which templates will be used.

**2. git add**

The git add command is used to add changes that have been made in the current working directory to the Git Staging area. The git Staging area is like an area which will record everything in that area before committing using the git commit command.

This command has various options like-

**git add <file>** - It will stage all changes in the file specified for the next upcoming commit.

**git add <directory>** - It will stage all changes in the directory for the next upcoming commit.

1. **git commit**

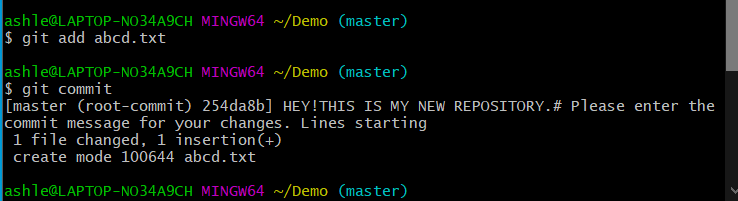
The git commit command is used to keep record of your project’s currently staged changes. The **git add** command must be executed prior to it for adding changes in the staging area.

This command has various options like-

**git commit -m “commit message” –** Used for creating a commit with an commit message

**git commit –amend –** This option will modify the commit that was last created rather than creating a new commit.

Below picture shows execution of **git add** with a file and git commit.



1. **git config**

The git config command is used to set the configuration values for Git. We can set these values on a local level of project or global level of project. It will modify the configuration text file (.gitconfig text files).

There are --local, --global and --system configuration levels.

We can also configure the text editor to be used, merge tools and can have coloured outputs.

For example, we can disable git’s coloured terminal output using(false)

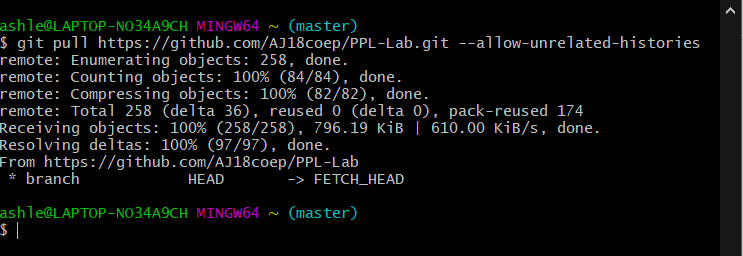
**git config –global color.ui false**

1. **git pull**

The git pull command is used to fetch and download the content/data from a repository

whose URL is specified(like a remote repository) in the command to your working directory.

This command comes with some common options like- --**verbose**(will give an verbose output), --**rebase,** --**no-commit**(does not create new merge commit)



1. **git push**

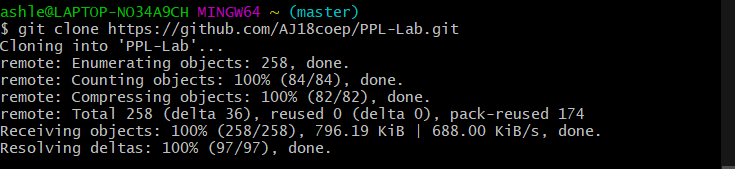
The git push command is used to send/upload the content from local repository to the remote repository, whose URL is specified in the command. You must take care that this command does not overwrite any changes. Some options for this command are-

--**all**(pushes all local branches)

--**tags**(sends local tags).

1. **git clone**

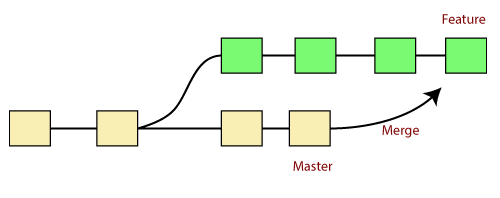
This command is used to create a local copy of any target repository in your current working directory, whose URL is specified in the **git clone** command. You can create the copy in any folder you wish by specifying it in the command.



1. **git merge**

The git merge command is generally used for combining two branches. It will join two or more histories of development.

The **git merge** command will take the content that is created by git branch and will make them into one single branch. The argument passed to git merge is that of the particular commit you wish to merge on an active branch.



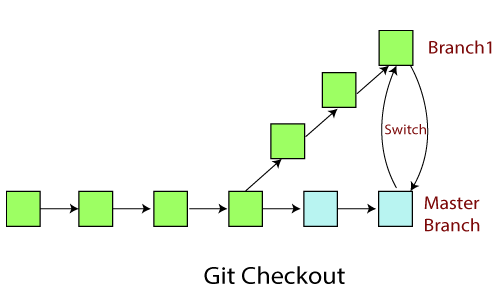
1. **git branch**

The git branch command is basically used for creating, listing, renaming and removing branches. We can perform various operations on Git branches. It does not let us switch in between the branches. It has options like- -**d**(delete a branch), -**m**(rename a branch),

-**a**(listing all the remote branches).

**10. git checkout**

The git checkout command is used for switching between different branches in a repository. This command should be used carefully as it is difficult to revert the changes made.

If one wishes to explore github more, refer the following links-

<https://www.youtube.com/playlist?list=PL4cUxeGkcC9goXbgTDQ0n_4TBzOO0ocPR>

<https://github.com/joshnh/Git-Commands>

The overall flow of various different commands has been indicated in the diagram below-

