**GIT PRACTISE AND ITS NOTES**

**What you Get from Version Control Sys**

Some of the major benefits of using a Version Control( provide to access different source code by roll back to get old version ,updates version etc System are listed below:

* Saves you from creating multiple backups of your files
* Allows multiple people to work on same file
* Tracks changes and also who have made those changes
* Easy to switch back to older versions as and when required
* Makes you more productive

**GIT is Version Control System**

Version Control is the management of changes to documents, computer programs, large websites and other collection of information.

* **Centralized Version Control System (CVCS):**

CVCS uses a central server to store all files. It works on a single repository to which users can directly access a central server. Central server could be local or remote machine directly connected to each of the programmer's workstation.

**Drawbacks of Centralized VCS**

Centralized repository is not locally available.

Since everything is centralized, in any case of the central server getting crashed or corrupted will result in losing the entire data of the project.

* **Distributed Version Control System (DVCS):**

In Distributed VCS, every contributor has a local copy or "clone" of the main repository.

Developer can update their local repositories with new data from the central server by an operation called "pull" and affect changes to the main repository by an operation called "push" from their local repository.

**Advantage of Distributed VCS**

All Operations are very fast because tools need to access the Hard Drive only.

Committing new change-sets can be done locally without manipulating the data on the main repository

If the central server gets crashed at any point of time, the lost data can be easily recovered from any one of the contributor’s local repositories.

**What is GIT**

Initially developed by Linus Torvalds, Git is a distributed version control system

Design Philosophy

* Free and Open source
* Blazingly Fast
* Distributed
* Data Assurance

**GIT Configuration:**

After installing GIT in our system.

The First thing we need to do is set our username and the email address.

This is because when we use git to change the files in the project. GIT uses this information to identify who has made the changes to the file.

The command to add the username and email address is "git config"

GIT has multiple level of configuration-

1- Repository/Project Level (Local)

2- User Account (Global Level)

GIT Config Locations:

Local (Repository/Project Level) repository/.git/config

Global (User Level) users/user-name/.gitconfig

1- Show user name & email ID

git config user.email

git config user.name

2- Set GIT Config for Global GIT repositoty(For All Project)

git config – global user.email dhruv@microsoft.com

git config – global user.name Dhruv

3- Remove a specific setting for a specific level of configuration:

git config --global --unset user.name

git config --global --unset user.emai

4- Set GIT Config for Global Local repositoty(For Project) :

git config --local user.name

git config --local user.email

5- Remove a specific setting for a specific level of configuration:

git config --unset user.name

git config --unset user.email

**How GIT Works:**

* Compared to other VCS like subversion. Most Operation in GIT requires local files and local resources to operate.
* If you are used to a Central GIT where most operations have that network latency overhead.
* For Example, to browse the history of the Project, GIT does not need to go out to the server to get the history and display it for you.
* It simply reads it directly from your local database.
* GIT can look up the file a month ago and do a local difference calculation
* There is no need to ask a remote server to do it or pull an older version of the file from the remote server to do it locally.
* This means you can do your work happily until you get to a network connection to upload.
* in Subversion and CVS, you can edit files, but you can not commit to your database (because your database is offline).

**Three States in GIT**

GIT has three main states that your files can reside in: modify, Staged, and committed:

Modify means that you have changed the files but have not committed it to your database yet.

Staged means that you have marked a modified file in its current version to go into your next commit snapshot.

Committed means that the data is safely stored in your local database.

Working Directory

The Working tree is a single checkout of one version of the project. These files are pulled out out from GIT directory and placed on disk for you to use or modify.

Staging Area

The staging area is a file generally contained in your GIT directory that stores information about what will go into your next commit. its technical name in GIT is the "index".

Repository Area

The GIT directory is where it stores the metadata and object database for your pre important part of GIT, and it is what is copied when you clone a repository from computer.

hdfs

STAGING AREA

(Git start tracking)

(This area initiates as we do git add <filename> / git add\*

red colour filer

cccc

BY passing (directly working area to commit area) this concept is applied only on tracked files it means it is once added in git.

git commit -a -m “Any message”

WORKING DIRECTORY

(Git is not tracking)

Green colour file

Repository Area/Commit Area

(Git will make its backup from here)

Tree clean