**Assignment 1 (GitHub)**

Problem statement:

ABC Organization would like to opt for the distributed version control system to upgrade their environment, where Git has been selected as the solution. You been assigned as a consultant to educate the migration process to move their Source Code from Centralized to Distributed systems. As a phase one, you would like to go ahead with a workshop to demonstrate below operation to make the ABC team comfortable.

i) Create a Repository - Done

ii) Add Two Directory and some raw code files to the repository

iii) Move Code from One directory to Another Directory

iv) Update one source code file and display the difference

v) Create a Branch

vi) Add some raw code to the branch

vii) Merge the Branch with Main line

And at the end provide the Summary of advantages of moving from Centralized Source Code to Distributed Version Control.

**Objective:**

The aim of this assignment to install GIT and also be able to perform basic GIT operations such as Creating Project or Repository, add files to repo and Commit the change.

I am using my Laptop for installation or not required if already installed in system.

Installing git Open the MacBook terminal and use Sudo apt install git command in order to install Git. Once installation is done, you will be able to see below message in the last

**In my case, git version 2.32.0 (Apple Git-132)**



Configure Git with your name, email id (use BITS Email id) and editor of your choice

Create a project/repository with **Git create** a folder for your project and use command git init to creates an empty Git repository

Add a file to the project. The **git status** command displays the state of the working directory and the staging area. Create a file and use command **git add<filename>** to add the file to git repository

Commit a file to the project Command to be used **git commit –a –m “message”**

1.

git init <directory>

git log

2.

mkdir directory1  -----to create directory

mkdir directory2

ls ----to view the files

ls -a  ----to show all file including hidden files

git add newFile1.java

git commit -m "added new file1"

echo "content added here" >> Newfile1.java

git add newFile2.java

git commit -m "added new file2"

echo "Put any content here" >> Newfile2.java

3.

$ git mv directory1 directory2

git mv Directory1/test2.txt new/   ----move file from dir 1 to dir 2

4.

vi <filename>               -- to open a file

i                           -- to insert into the file

ESC button followed by :wq   -- to save and close the file

echo "content added here" >> Newfile1.java

5. Create a Branch

git branch

git branch new\_branch

ls ----to view the files

6.

ls ----to view the files

git add newFile3.java

ls ----to view the files

7.Merge the Branch with Main line

Move to main branch

git checkout main

git branch

ls ----to view the files

git merge new\_branch

Unlike a centralized version control system (VCS), a distributed version control system enables every user to have a local copy of the running

history on their machine, so if there's an outage, every local copy becomes a backup copy and team members can

continue to development offline.

A distributed version control doesn't have a single point of failure, because developers clone repositories on their distributed version control

workstations, creating multiple backup copies.

DVCS has the biggest advantage in that it allows you to work offline and gives flexibility. You have the entire

history of the code in your own hard drive, so all the changes you will be making in your own server or to

your own repository which doesn’t require an internet connection, but this is not in the case of CVCS.

DVCS is faster than CVCS because you don’t need to communicate with the remote server for each and every command.

You do everything locally which gives you the benefit to work faster than CVCS.

Working on branches is easy in DVCS. Every developer has an entire history of the code in DVCS,

so developers can share their changes before merging all the ‘sets of changes to the remote server.

In CVCS it’s difficult and time-consuming to work on branches because it requires to communicate with the server directly.

Merge conflicts with other developer’s code are less in DVCS.

Because every developer work on their own piece of code. Merge conflicts are more in CVCS in comparison to DVCS.

DVCS has more advantages and it’s more popular than CVCS