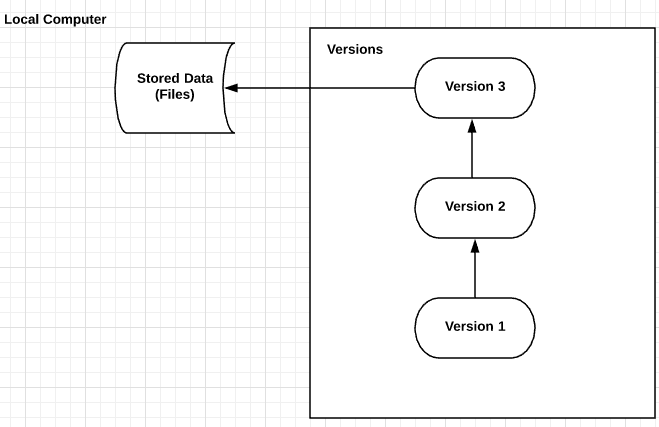
**Experiment No. 1**

# Aim-: To study and implement Version control using GIT Theory:

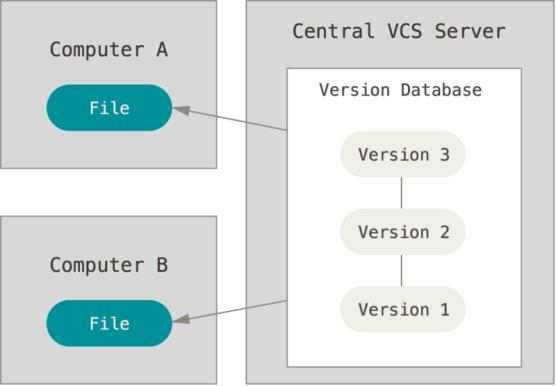
Version control allows you to keep track of your work and helps you to easily explore the changes you have made, be it data, coding scripts, notes, etc. Version control systems are also called as revision control systems. Revision control systems work as independent standalone applications. Applications like spreadsheets and word processors have control mechanisms. The unique features of version control system/ revision control system are as follows: Up to date history is available for the document and file types. It does not require any other repository systems. The repositories can be cloned as per the need and availability. This is extremely helpful in case of failure and accidental deletions. VCS includes tag system which helps in differentiating between alpha, beta or various release versions for different documents.

The various types of the version control systems are:

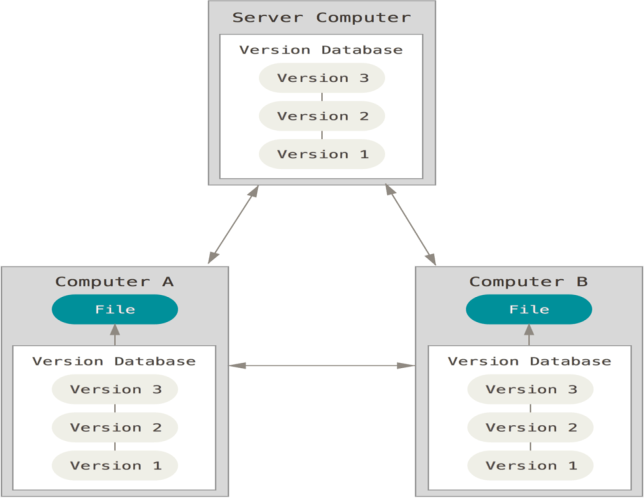
1. Local Version Control System
2. Centralized Version Control System
3. Distributed Version Control System
4. **Local version control system**: Local version control system maintains track of files within the local system. This approach is very common and simple. This type is also error prone which means the chances of accidentally writing to the wrong file is higher.



1. **Centralized Version Control System:** In this approach, all the changes in the files are tracked under the centralized server. The centralized server includes all the information of versioned files, and list of clients that check out files from that central place.

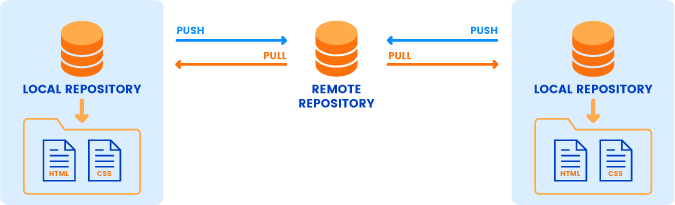


1. **Distributed Version Control System:** Distributed version control systems come into picture to overcome the drawback of centralized version control system. The clients completely clone the repository including its full history. If any server dies, any of the client repositories can be copied on to the server which help restore the server.



Git is a free and open-source distributed version control system designed to handle everything from small to very large projects with speed and efficiency. Git is easy to learn and has a tiny footprint with lightning-fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows. Git is a distributed version control system (DVCS). "Distributed" means that all developers within a team have a complete version of the project. A version control system is simply software that lets you effectively manage application versions. Thanks to Git, you'll be able to do the following:

* + Keep track of all files in a project
  + Record any changes to project files
  + Restore previous versions of files
  + Compare and analyze code
  + Merge code from different computers and different team members.



The commonly used git commands are listed as follows

15L@203-02 MINGW64 ~ (main)

$ mkdir git-dvcs

15L@203-02 MINGW64 ~ (main)

$ cd git-dvcs

15L@203-02 MINGW64 ~/git-dvcs (main)

$ git config -global

error: did you mean `--global` (with two dashes)?

15L@203-02 MINGW64 ~/git-dvcs (main)

$ git config --global user.name "Divyansh"

15L@203-02 MINGW64 ~/git-dvcs (main)

$ git config --global user.email "divyanshsancheti13@gmail.com

> "

15L@203-02 MINGW64 ~/git-dvcs (main)

$ git config --global --list

user.name=Divyansh

user.email=divyanshsancheti13@gmail.com

color.ui=true

15L@203-02 MINGW64 ~/git-dvcs (main)

$ mkdir git-demo-project

15L@203-02 MINGW64 ~/git-dvcs (main)

$ cd git-demo-project/

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (main)

$ git init

Initialized empty Git repository in C:/Users/15L/git-dvcs/git-demo-project/.git/

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ rm -rf .git/

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (main)

$ git init

Initialized empty Git repository in C:/Users/15L/git-dvcs/git-demo-project/.git/

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ git add

Nothing specified, nothing added.

hint: Maybe you wanted to say 'git add .'?

hint: Turn this message off by running

hint: "git config advice.addEmptyPathspec false"

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ git status

On branch master

No commits yet

Untracked files:

(use "git add <file>..." to include in what will be committed)

div.txt

index.html

style.css

nothing added to commit but untracked files present (use "git add" to track)

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ git commit -m "first commit"

On branch master

Initial commit

Untracked files:

(use "git add <file>..." to include in what will be committed)

div.txt

index.html

style.css

nothing added to commit but untracked files present (use "git add" to track)

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ git add

Nothing specified, nothing added.

hint: Maybe you wanted to say 'git add .'?

hint: Turn this message off by running

hint: "git config advice.addEmptyPathspec false"

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ git add .

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ git commit -am "express Commit"

[master (root-commit) 2948bb0] express Commit

3 files changed, 0 insertions(+), 0 deletions(-)

create mode 100644 div.txt

create mode 100644 index.html

create mode 100644 style.css

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ nano index.html

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ touch teststatus

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ git remote add origin https://github.com/DivyanshSEPM/sepm-lab-1.git

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (master)

$ git branch -M main

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (main)

$ git push -u origin main

Enumerating objects: 3, done.

Counting objects: 100% (3/3), done.

Delta compression using up to 20 threads

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 242 bytes | 242.00 KiB/s, done.

Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

To https://github.com/DivyanshSEPM/sepm-lab-1.git

\* [new branch] main -> main

branch 'main' set up to track 'origin/main'.

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (main)

$ git pull

remote: Enumerating objects: 5, done.

remote: Counting objects: 100% (5/5), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 680 bytes | 35.00 KiB/s, done.

From https://github.com/DivyanshSEPM/sepm-lab-1

2948bb0..b951a11 main -> origin/main

Updating 2948bb0..b951a11

Fast-forward

div.txt | 1 +

1 file changed, 1 insertion(+)

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (main)

$ git fetch

remote: Enumerating objects: 5, done.

remote: Counting objects: 100% (5/5), done.

remote: Compressing objects: 100% (2/2), done.

remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (3/3), 688 bytes | 36.00 KiB/s, done.

From https://github.com/DivyanshSEPM/sepm-lab-1

b951a11..30958ae main -> origin/main

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (main)

$ git merge

Updating b951a11..30958ae

Fast-forward

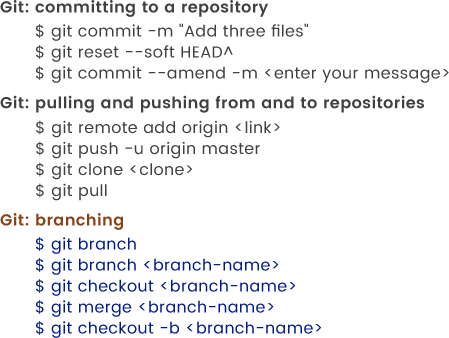
div.txt | 2 +-

1 file changed, 1 insertion(+), 1 deletion(-)

15L@203-02 MINGW64 ~/git-dvcs/git-demo-project (main)

$





# Conclusion: Thus, we have successfully studied and implemented version control using GIT