Experiment No. 1

Aim-: To study and implement Version control using GIT

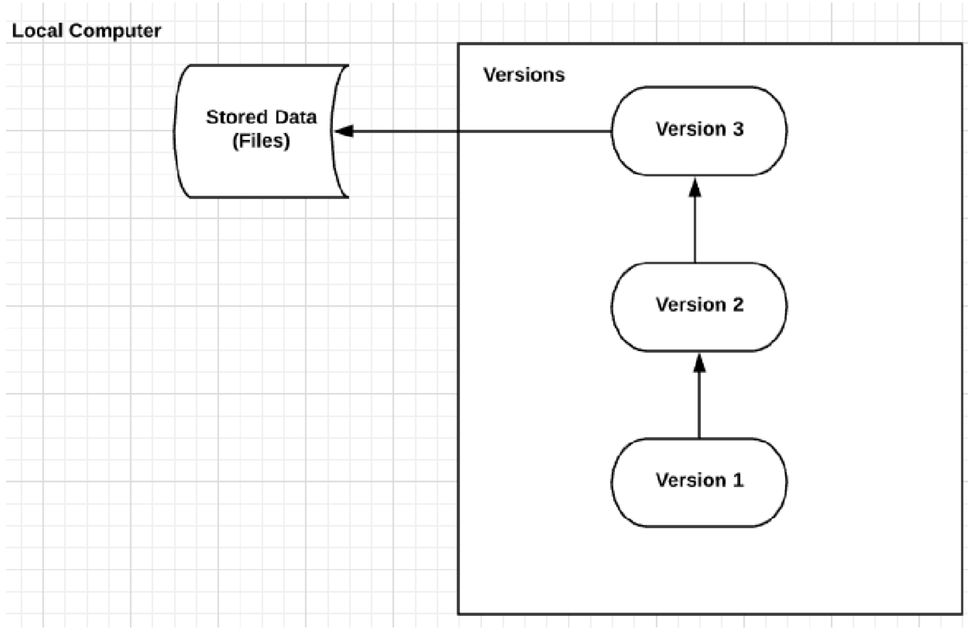
Theory:

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.

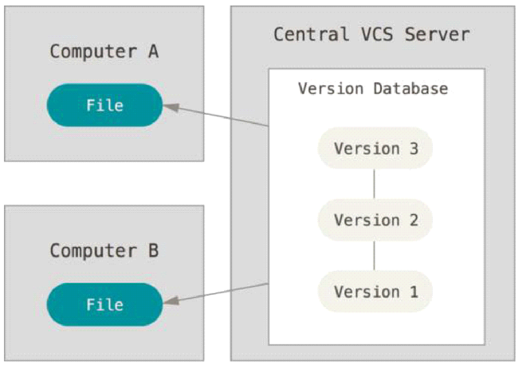
The various types of the version control systems are:

- Local Version Control System
- Centralized Version Control System
- Distributed Version Control System

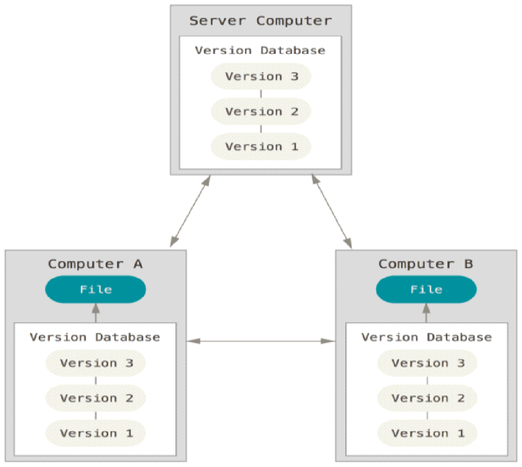
- **Local version control system**: Local version control system maintains track of files within the local system. This approach is very common and simple. This type is also error prone which means the chances of accidentally writing to the wrong file is higher.



- **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.



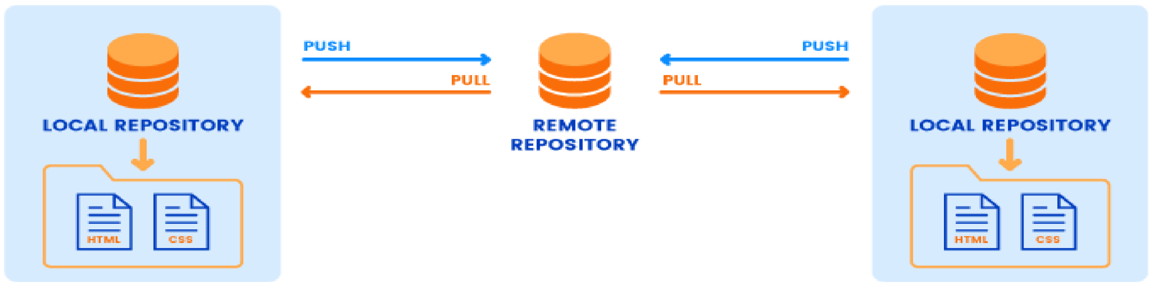
- **Distributed Version Control System:** Distributed version control systems come into picture to overcome the drawback of centralized version control system. The clients completely clone the repository including its full history. If any server dies, any of the client repositories can be copied on to 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@203-01 MINGW64 ~
$ mkdir git-dvcs2

15L@203-01 MINGW64 ~
$ cd git-dvcs2/

15L@203-01 MINGW64 ~/git-dvcs2
$ git config -global
error: did you mean `--global` (with two dashes)?

15L@203-01 MINGW64 ~/git-dvcs2
$ git config --global user.name "aditya"

15L@203-01 MINGW64 ~/git-dvcs2
$ git config --global user.email "adityasavant13@gmail.com"

15L@203-01 MINGW64 ~/git-dvcs2
$ git config --global --list
user.email=adityasavant13@gmail.com
user.name=aditya

15L@203-01 MINGW64 ~/git-dvcs2
$ cd git-demo-project/
bash: cd: git-demo-project/: No such file or directory

15L@203-01 MINGW64 ~/git-dvcs2
$ mkdir git-demo-project

15L@203-01 MINGW64 ~/git-dvcs2
$ cd git-demo-project/

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project
$ git init
```

```
Initialized empty Git repository in
C:/Users/15L/git-dvcs2/git-demo-project/.git/

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git add .

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to
track)

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git commit -m "First Commit"
On branch master

Initial commit

nothing to commit (create/copy files and use "git add" to
track)

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to
track)

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git add .

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
```

```
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   adityaexp1.html


15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git commit -m "First Commit"
[master (root-commit) fee4a61] First Commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 adityaexp1.html

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git status
On branch master
nothing to commit, working tree clean

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git add .

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git commit -m "express Commit"
On branch master
nothing to commit, working tree clean

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ nano adityaexp1.html

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
$ git remote add origin
https://github.com/AdityaSEPM/Lab1.git

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (master)
```

```
$ git branch -M main

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (main)
$ git push -u origin main
remote: Permission to AdityaSEPM/Lab1.git denied to
fahad-charolia.
fatal: unable to access
'https://github.com/AdityaSEPM/Lab1.git/': The requested URL
returned error: 403

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (main)
$ git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 219 bytes | 219.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/AdityaSEPM/Lab1.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (main)
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 649 bytes | 108.00 KiB/s,
done.
From https://github.com/AdityaSEPM/Lab1
   fee4a61..fd73b21  main       -> origin/main
error: Your local changes to the following files would be
overwritten by merge:
        adityaexp1.html
Please commit your changes or stash them before you merge.
Aborting
Updating fee4a61..fd73b21

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (main)
```

```
$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 659 bytes | 164.00 KiB/s,
done.
From https://github.com/AdityaSEPM/Lab1
   fd73b21..67c946a  main       -> origin/main

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (main)
$ git merge
error: Your local changes to the following files would be
overwritten by merge:
        adityaexp1.html
Please commit your changes or stash them before you merge.
Aborting
Updating fee4a61..67c946a

15L@203-01 MINGW64 ~/git-dvcs2/git-demo-project (main)
$
```

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