Experiment No :- 1

Vedant Jayesh Oza
TY_IT_T4-60

Aim :- To study & use Distributed Version Control System.

Objective:- To understand Version Control & Its use.

Theory:-

Version control is a system that records changes to a file or set of files over time, allowing users to recall specific versions later. It is commonly used in software development and other collaborative projects to keep track of changes made by different team members. Popular version control systems include Git, Mercurial, and Subversion.

There are several types of version control systems, including:

1. Local Version Control Systems: These systems store the version history on a single developer's local machine. Examples include RCS, SCCS, and VSS.
2. Centralized Version Control Systems: These systems store the version history on a single, central server. Examples include Subversion, CVS, and Perforce.
3. Distributed Version Control Systems: These systems store the version history across multiple repositories, allowing for distributed development. Examples include Git, Mercurial, and Bazaar.
4. Cloud-based Version Control Systems: These are the version control systems that are hosted on cloud and allow users to access their repos from anywhere. Example include GitHub, GitLab, Bitbucket.
5. Hybrid Version Control System: These are the version control systems that are based on combination of above two or more types, such as Git-SVN, Git-TFS etc.

Distributed Version Control System- Multiple repositories make up distributed version control systems. Each user gets access to his or her own repository and working copy. Simply committing your changes does not grant anyone access to them. This is due to the fact that commit will reflect these changes in your local repository, but push is required to make them accessible in the central repository. Likewise, when you update, you do not receive others' modifications unless you have previously merged them into your repository.

To make your changes visible to others, 4 things are required:
- You commit
- You push
- They pull
- They update

The most popular distributed version control systems are Git, and Mercurial. They help us overcome the problem of a single point of failure. Git is a distributed, open-source version control system. It is meant to manage small to large jobs quickly and effectively. It is created to coordinate the developers' efforts. The version control enables us to monitor and collaborate with team members in the same workspace.

Performance :-

Git init

Git Status & Git add

```
Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        names.txt

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

Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git add .

Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   names.txt
```

Git commit, Git Config

```
Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git config --global user.email "vedantoza1313@gmail.com"

Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git config --global user.name "Vedant-Jayesh-Oza"

Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git commit -m "names.txt file added"
[master (root-commit) ed8cb82] names.txt file added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 names.txt
```

Git Restore

```
Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   names.txt


Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git restore --staged names.txt

Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   names.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Git Log

```
Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git log
commit 6428732db60335179645f72c841253fc4481e61f (HEAD -> master)
Author: Vedant-Jayesh-Oza <vedantoza1313@gmail.com>
Date:   Mon Aug 2 11:13:30 2021 +0530

    names.txt file modified

commit ed8cb825e940bd35e261eaa2e1089e32fc2814d0
Author: Vedant-Jayesh-Oza <vedantoza1313@gmail.com>
Date:   Mon Aug 2 11:00:06 2021 +0530

    names.txt file added
```

Git Reset

```
Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git reset ed8cb825e940bd35e261eaa2e1089e32fc2814d0
Unstaged changes after reset:
D       names.txt

Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git log
commit ed8cb825e940bd35e261eaa2e1089e32fc2814d0 (HEAD -> master)
Author: Vedant-Jayesh-Oza <vedantoza1313@gmail.com>
Date:   Mon Aug 2 11:00:06 2021 +0530

    names.txt file added
```

Git Stash

```
Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git stash
Saved working directory and index state WIP on master: ed8cb82 names.txt file added
```

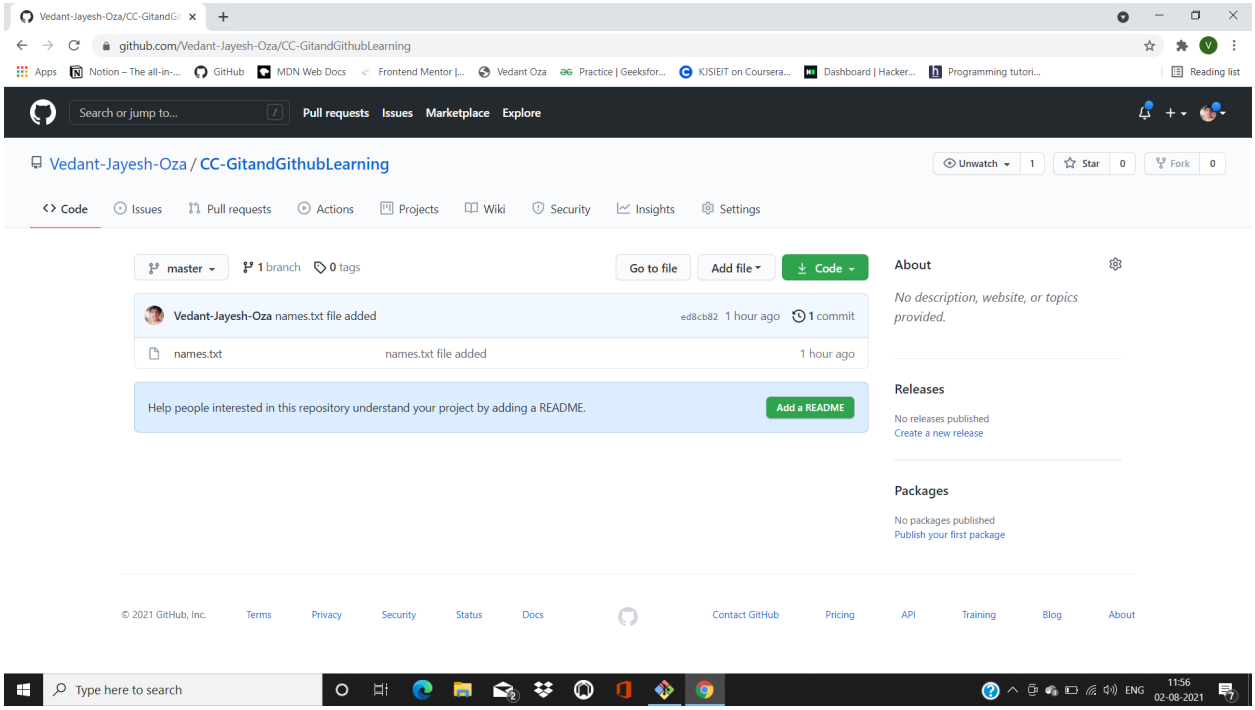
Git Remote

```
Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git remote add origin https://github.com/Vedant-Jayesh-Oza/CC-GitandGithubLearning

Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git remote -v
origin  https://github.com/Vedant-Jayesh-Oza/CC-GitandGithubLearning (fetch)
origin  https://github.com/Vedant-Jayesh-Oza/CC-GitandGithubLearning (push)
```

Git Push

```
Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git push origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 228 bytes | 228.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/Vedant-Jayesh-Oza/CC-GitandGithubLearning
 * [new branch]      master -> master
```

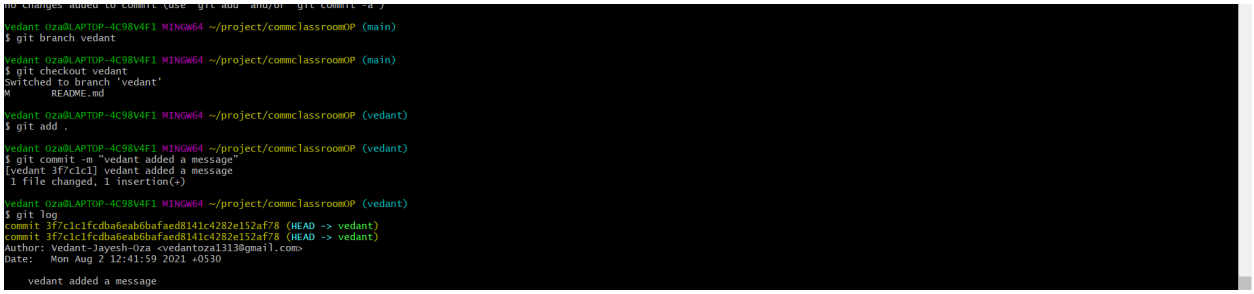


Git Clone

```
Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
$ git clone https://github.com/Vedant-Jayesh-Oza/commclassroomOP.git
Cloning into 'commclassroomOP'...
remote: Enumerating objects: 36, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 36 (delta 4), reused 0 (delta 0), pack-reused 28
Receiving objects: 100% (36/36), 9.36 KiB | 2.34 MiB/s, done.
Resolving deltas: 100% (7/7), done.

Vedant Oza@LAPTOP-4C98V4F1 MINGW64 ~/project (master)
```

Git branch



Outcome:- Able to use version Control system & its use.

Conclusion:-

By tracking all code revisions, the version control system helps the software team manage the source code. Additionally, it protects the source code from unanticipated human error and its consequences. All code versions are saved, so this enables engineers to compare the more recent code versions at any time to assist in fixing the minimise team disruption mistakes. As a result, we comprehend the fundamentals of version control and its advantages