

Experiment – 2

Git Version Control

Objective: The objective of this experiment is to understand and implement version control using local repositories.

Software Used/ Tools Required: Git, GitHub Account, Visual Studio Code.

Theory Overview:

Version control is the process of tracking and managing changes made to software code, documents, or files over time. It allows developers to maintain a complete history of changes, collaborate effectively without conflicts, and restore previous versions if required. This is essential in software development, where multiple team members may be working on the same project simultaneously.

Git is a widely used **distributed version control system**. Unlike centralized systems (where all data is stored on a single server), Git stores the entire project history on every contributor's local machine. This means you can work offline, commit changes locally, and later sync them with others via a remote repository.

A **repository** (or “repo”) is a storage location where Git tracks all files, commits, and project metadata. Repositories can be:

- **Local Repository** – Stored on your personal computer. It contains:
 1. **Working Directory** – The actual project files you edit.
 2. **Staging Area (Index)** – A place where changes are collected before committing.
 3. **Git Directory (.git folder)** – Stores the complete history of commits, branches, and configurations.

When you make changes, you first save them in the **staging area**, then create a **commit** (a snapshot of the project at that point). These commits stay in your **local repository** until you push them to a **remote repository** like GitHub or GitLab, making them available to other collaborators.

Git also supports **branching**, which allows you to work on different versions of a project simultaneously. For example, you can create a feature branch to develop a new feature without affecting the main branch. Once complete, you can merge it back into the main branch.

Procedure:

1. **Create a New Folder** (Open PowerShell and create a new directory):
 - `mkdir text_file`
2. **Navigate to the Directory**(Change directory to the newly created folder)

- Cd text_file

3. Initialize a Local Git Repository

- git init

4. Create Sample File

- echo "This is a sample file" > sample.txt

5. Commit changes

- git commit -m "First commit: Adding sample.txt"

6. Connect to Remote Repository

- git remote add origin <remote_repo_URL>

7. Push Changes to Remote

- git push origin main

8. Check Commit History

- git log

Code:

```
PS C:\Users\arnav\OneDrive\Desktop\git\Capital> git commit -m "New file"
On branch main
nothing to commit, working tree clean
PS C:\Users\arnav\OneDrive\Desktop\git\Capital> git remote add origin https://github.com/shristi13star/Capital.git
PS C:\Users\arnav\OneDrive\Desktop\git\Capital> git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 240 bytes | 120.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
```

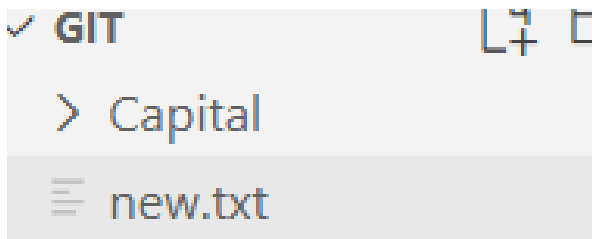
- In this code, a new folder is made using **mkdir** command, then in the folder root a file is made using **echo** command, then it was pushed to GitHub.

```
PS C:\Users\arnav\OneDrive\Desktop\git> git log
commit 2b31be20ef4c453f45c90c4d11126db762f2c9ff (HEAD -> main)
Author: Shruti Singh <arnavsingh14star@gmail.com>
Date:   Wed Aug 13 12:38:48 2025 +0530
```

added new test file

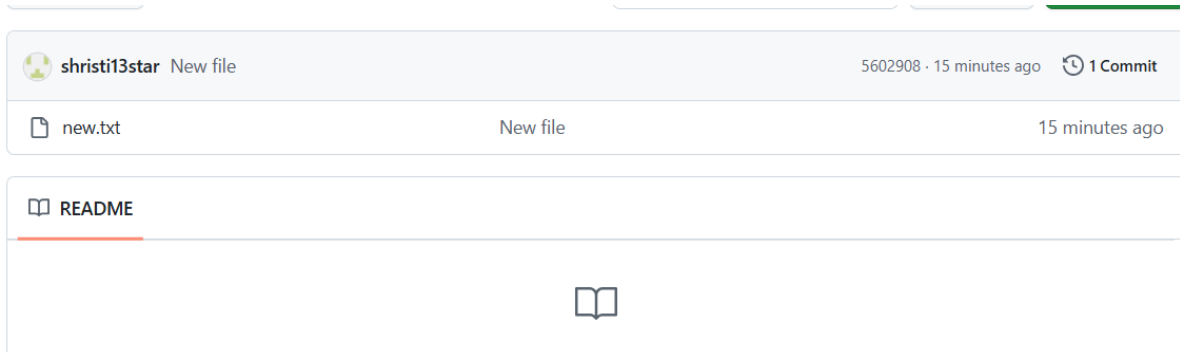
```
commit 8bf401f1913dffa50bf3e511abdd8c8f3c450693
Author: Shruti Singh <arnavsingh14star@gmail.com>
Date:   Wed Aug 6 14:19:01 2025 +0530
```

- In this code, we have used **git log** command to see the history of changes made.



Output:

- Output showing a sample.txt file was made using echo command in terminal.



- Output showing that new file has been successfully pushed to GitHub.

Experiment – 3

Remote Repository and branch Mangement

Objective:

To understand how to create, manage, and synchronize remote repositories, and to learn the process of branch creation, switching, merging, and deletion in Git for collaborative development.

Software Used/ Tools Required: Git, GitHub Account, Visual Studio Code.

Theory:

Remote Repository

A remote repository is a version of your project that is hosted on the internet or a network, typically using platforms like GitHub, GitLab, or Bitbucket. It allows multiple collaborators to contribute to the same project from different locations. The remote repository stores the complete history of the project and is linked to your local repository via a URL.

- Common Remote Commands in Git:
 - `git remote add origin <URL>` → Links local repo to a remote one.
 - `git push origin <branch>` → Sends local commits to the remote repository.
 - `git pull origin <branch>` → Fetches and merges changes from remote to local.
 - `git fetch` → Downloads remote changes without merging them.

Branch Management in Git

A branch in Git is an independent line of development. Branches allow you to work on new features, bug fixes, or experiments without affecting the main (usually main or master) branch.

- Common Branch Commands in Git:
 - `git branch` → Lists all branches.
 - `git branch <branch-name>` → Creates a new branch.
 - `git checkout <branch-name>` → Switches to the specified branch.
 - `git merge <branch-name>` → Merges changes from the specified branch into the current branch.
 - `git branch -d <branch-name>` → Deletes a branch.

Procedure:

- 1. Create a new branch named TestBranch:**
 - `git branch TestBranch`
- 2. Switch to the new branch:**
 - `git checkout TestBranch`
- 3. Create a new text file named hello.txt:**
 - `echo "Hello from TestBranch" > hello.txt`
- 4. commit the file:**
 - `git add hello.txt`
 - `git commit -m "Added hello.txt in TestBranch"`
- 5. Push the branch to GitHub:**
 - `git push origin TestBranch`
- 6. On GitHub, create a Pull Request from TestBranch to main.**
- 7. Compare the changes and accept the merge.**
- 8. Pull the latest merged changes to your local main branch:**
 - `git checkout main`
 - `git pull origin main`

Code:

- PS C:\Users\arnav\OneDrive\Desktop\git> git branch
 - * f2
 - f3
 - main
- PS C:\Users\arnav\OneDrive\Desktop\git> git checkout f2
 - Already on 'f2'
 - error: Unable to create 'C:/Users/arnav/OneDrive/Desktop/git/.git/packed-refs.lock': File exists.

Another git process seems to be running in this repository, e.g.
an editor opened by 'git commit'. Please make sure all processes

- In this code, we made a new branch using **git branch TestBranch** and switched from main to it using **git checkout TestBranch**, then we made a new file in this branch using **echo** command and then pushed it to GitHub in testbranch.










The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar shows the file structure with 'text3.txt' selected. The main editor area shows 'text3.txt' with two lines of code: '1 file' and '2'. Below the editor, the TERMINAL panel is open, displaying the following output:

```
Another git process seems to be running in this repository, e.g.  
an editor opened by 'git commit'. Please make sure all processes  
are terminated then try again. If it still fails, a git process  
may have crashed in this repository earlier:  
remove the file manually to continue.  
error: Unable to create 'C:/Users/arnav/OneDrive/Desktop/git/.git/packed-refs.lock': File exists.  
  
Another git process seems to be running in this repository, e.g.  
an editor opened by 'git commit'. Please make sure all processes  
are terminated then try again. If it still fails, a git process  
may have crashed in this repository earlier:  
remove the file manually to continue.  
[f2 d97ae8e] added  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 new.txt  
PS C:\Users\arnav\OneDrive\Desktop\git> git checkout main  
Switched to branch 'main'  
error: Unable to create 'C:/Users/arnav/OneDrive/Desktop/git/.git/packed-refs.lock': File exists.  
  
Another git process seems to be running in this repository, e.g.
```

- In this code, We pulled changes made in testbranch to main branch in our system. Then we deleted the TestBranch.

Output:

- When we push from TestBranch, We get Compare and Pull request option in Github as shown to merge the branch. Accepting this will merge the branch.

 shristi13star added		d97ae8e · 29 minutes ago	🕒 10 Commits
 Capital	added new value	1 hour ago	
 README.md	added new test file	1 hour ago	
 README1.md	Add README1.md	last week	
 new.txt	added	29 minutes ago	
 test1.txt	added new value	1 hour ago	
 text3.txt	added	1 hour ago	
 README			

- Output showing the Pull request successfully merged and closed.