PURBANCHAL UNIVERSITY

KHWOPA ENGINEERING COLLEGE

LIBALI-08, BHAKTAPUR

LAB REPORT ON .NET

LAB NO. 01

SUBMITTED BY:

Name: Nischal Baidar

Roll No. : 770323

Group: A

SUBMITTED TO:

Department of Computer Engineering

Submission: 2081/12/08

Theory:

Git is a distributed version control system (DVCS) that allows multiple developers to work on a project simultaneously. It helps track changes, revert to previous versions, and manage collaboration efficiently.

Features of Git:

1. **Branching and Merging** – Work on different features without affecting the main codebase.
2. **History Tracking** – Maintain a log of all changes with commit messages.
3. **Collaboration** – Enables multiple developers to work on the same project without conflicts.

GitHub, on the other hand, is a cloud-based platform that hosts Git repositories, enabling collaboration. The workflow illustrated in the image represents how changes move through different stages in a Git project**.**

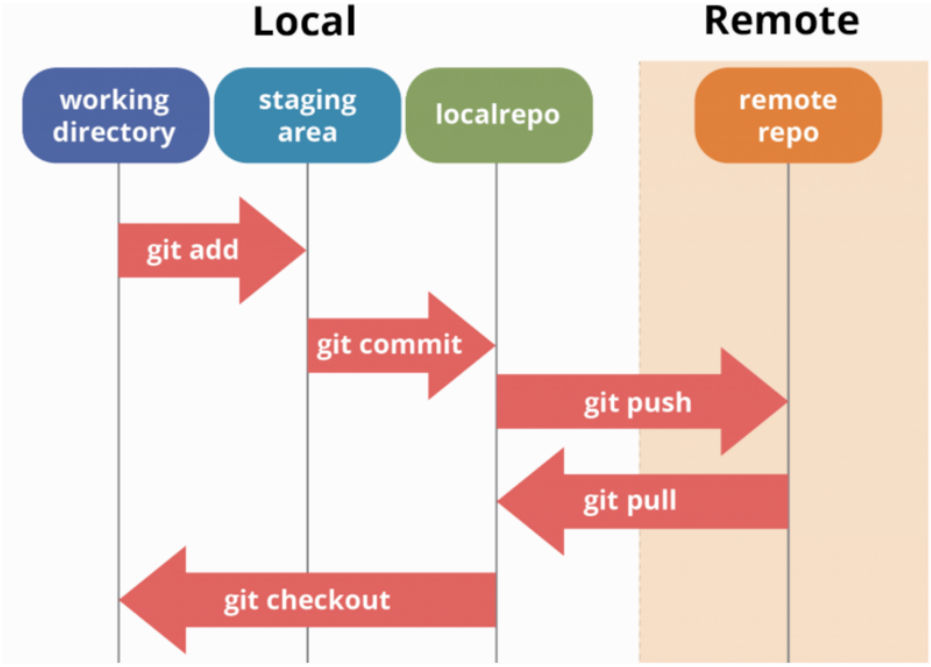


Fig: Git Workflow

Local Workflow

A Git project consists of three main areas:

1. Working Directory – This is where you modify files.
2. Staging Area – This holds changes that are marked for the next commit.
3. Local Repository – This store committed changes.

Key Commands:

- git add → Moves changes from the working directory to the staging area.

- git commit → Saves staged changes in the local repository.
- git checkout → Switches branches or restores files to a previous state.

Syncing with Remote Repository

The remote repository (hosted on GitHub) serves as a central location to share and collaborate on code.

Key Commands:

- git push → Uploads committed changes from the local repository to the remote repository.
- git pull → Fetches and integrates changes from the remote repository into the local repository.

By following this structured workflow, developers can efficiently manage code versions, collaborate with teams, and maintain a clean project history.

Git and GitHub Commands Discussed during Lab Works:

Category	Command	Description
Git Configuration	git config --global user.name “nischal”	Sets your name for commits.
	git config --global user.email “nischalbaidar@example.com”	Assigns an email to commits.
Initializing a Repository	git init	Creates a new Git repository in the current directory.
Staging and committing	git add .	Stages all changes for commit.
	git commit -m “Commit message”	Saves staged changes with a message.
Branching and Merging	git branch	Lists all branches in the repository.
	git branch <branch_name>	Creates a new branch.
	git checkout <branch_name> / git switch <branch_name>	Switches to another branch.
	git merge <branch_name>	Merges changes from one branch to another.
Syncing with GitHub	git push -u origin <branch_name>	Uploads local commits to GitHub.
	git pull origin <branch_name>	Fetches and integrates changes from GitHub.
Checking Status & History	git status	Shows the current state of the working directory.

	git log	Displays commit history
Connecting to GitHub	git remote add origin <repo_url>	Links the local repository to GitHub.

Lab Works

Configure the users username and email for commits.

```
nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (main)
$ git config --global user.name "nischal"
```

```
nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (main)
$ git config --global email "nischalbaidar@gmail.com"
```

Initializing the git repository by creating a new folder.

```
nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (main)
$ git init
Reinitialized existing Git repository in D:/Dot_Net_Git_Lab/.git/
```

Create a new branch to work on and switch to it.

```
nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (master)
$ git branch testing

nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (master)
$ git checkout testing
Switched to branch 'testing'
```

Create a new file and write content in it.

```
nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ touch nis.txt

nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ echo "hey its nischal baidar" > nis.txt
```

Now checking the status of testing branch. As testing branch is created from master branch so initially it had other files that was created during lab hours so I deleted it and added new text file nis.txt

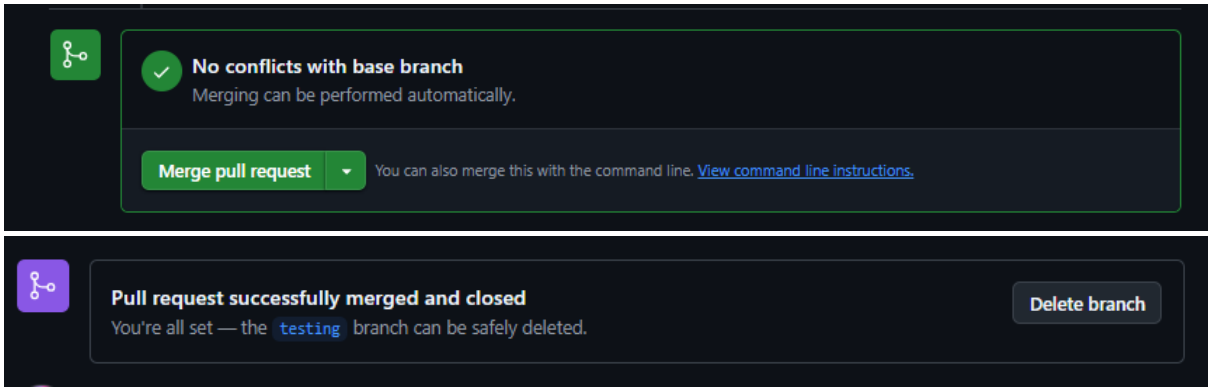
```
nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ git status
On branch testing
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    basu.py
        deleted:    bud.txt
        deleted:    niki.txt
        modified:   nis.txt

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

Now adding committing and syncing with GitHub:

```
nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ git add .

nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ git status
On branch testing
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    basu.py
        deleted:    bud.txt
        deleted:    niki.txt
        modified:   nis.txt


nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ git commit -m "done"
[testing 5f1686a] done
 4 files changed, 1 insertion(+), 4 deletions(-)
 delete mode 100644 basu.py
 delete mode 100644 bud.txt
 delete mode 100644 niki.txt

nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ git log
commit 5f1686a606ea18c2a0b597a21e14cfe840ec9e4f (HEAD -> testing)
Author: Bainash <nischalbaidar@gmail.com>
Date:   Fri Mar 21 16:17:39 2025 +0545

    done
```

```
nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ git remote add origin "https://github.com/bainash10/Dot_Net_Git_Lab.git"
error: remote origin already exists.

nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ git push origin testing
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (1/1), done.
Writing objects: 100% (3/3), 258 bytes | 129.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'testing' on GitHub by visiting:
remote:      https://github.com/bainash10/Dot_Net_Git_Lab/pull/new/testing
remote:
To https://github.com/bainash10/Dot_Net_Git_Lab.git
 * [new branch]      testing -> testing

nisch@Bainash-Laptop MINGW64 /d/Dot_Net_Git_Lab (testing)
$ git status
On branch testing
nothing to commit, working tree clean
```

Now merge the pull request by comparing master and testing branch



Conclusion:

Here we learned about how git works, why git is important and what is git.