Lab 1: Implementation of GIT

Objective

The objective of this lab is to understand and implement the fundamental concepts of version control using **GIT**. This includes learning how to set up a Git repository, perform basic Git operations such as `init`, `clone`, `add`, `commit`, `push`, `pull`, `branch`, and `merge`, and manage source code in a collaborative environment. By the end of this lab, students should be able to track changes, collaborate with teammates, and resolve simple conflicts using Git.

aadarsh baral

Materials Used

- A computer with internet access
- Operating system: Windows / macOS / Linux
- **Git** installed (version 2.x or above)
- A code editor or IDE (e.g. Visual Studio Code)

Theory

Version control systems (VCS) are tools that help manage changes to source code over time. **Git** is one of the most widely used distributed version control systems that enables multiple developers to work on the same codebase without overwriting each other's work.

Some key concepts in Git:

aadarsh baral

- **Repository**: A directory which contains your project files and a `.git` folder where Git stores the version history.
- **Commit**: A snapshot of the changes made to the files in the repository.
- **Branch**: A parallel version of the repository that allows isolated development without affecting the main codebase.
- **Merge**: The process of combining changes from different branches.
- **Remote repository**: A version of your project hosted on the internet (e.g. GitHub) that can be shared with others.

Git operations can be broadly classified as:

- **Local operations**: Initialize repo, stage files, commit changes, create branches.
- **Remote operations**: Push changes to, pull updates from, or clone a remote repository.

Implementation

1. Initialize a Git repository

```
mkdir git-lab

cd git-lab

git init
```

This creates a new Git repository locally in the `git-lab` folder.

2. Configure Git (if not done already)

```
git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"
```

This sets your identity for commits.

3. Create a file and track it

```
echo "# Git Lab" > README.md

git status

git add README.md

git commit -m "Add README file"
```

Here, we create a README file, stage it, and commit it.

4. Connect to a remote repository

Create a repository on GitHub (e.g., `git-lab`). Then link it:

```
git remote add origin https://github.com/username/git-lab.git
```

```
git push -u origin master
```

This pushes the local repository to GitHub. aadarsh baral

5. Clone a remote repository

```
git clone https://github.com/username/git-lab.git
```

This command creates a local copy of a remote repository.

6. Create and switch branches

```
git branch feature-branch

git checkout feature-branch

# OR in one command

git checkout -b feature-branch
```

This creates and switches to a new branch for feature development.

7. Merge branches

```
git checkout master

git merge feature-branch
```

Merges changes from `feature-branch` into `master`.

8. Handle conflicts

When merging, if two branches change the same line in a file, Git will report a conflict. The file will include markers like:

```
<<<<<<< HEAD
Current changes
=======
Incoming changes
>>>>>>> feature-branch
```

aadarsh baral

We must manually edit the file to resolve the conflict, then:

```
git add <file>
git commit
```

9. Pull updates from remote

```
git pull origin master
```

Fetches and merges changes from the remote repository.

Conclusion

In this lab, we explored the core concepts and operations of Git, a vital tool for modern software development. We set up a local repository, tracked changes with `add` and `commit`, connected to GitHub for collaboration, created branches, merged code, and resolved conflicts.

Git helps developers manage code history, collaborate effectively, and maintain code integrity. Its integration with platforms like GitHub provides features like pull requests and issue tracking.

By completing this lab, we gained practical experience with:

- Setting up and configuring Git
- Managing local and remote repositories
- Branching, merging, and conflict resolution

aadarsh baral