Preparation Before the Lab

Installations:

Ensure Git is installed on all lab systems.

Students should have a GitHub account ready.

Step-by-Step Lab

Setting Up Git

Objective: Configure Git on local systems.

Steps:

Open the terminal or Git Bash.

Run the following commands to set up Git user information:

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

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

Verify the setup:

git config --list

Explain: This is necessary for Git to track changes and attribute commits.

Creating a Local Repository

Objective: Learn to create and initialize a repository.

Steps:

Create a new folder for the project:

mkdir my-first-repo

cd my-first-repo

Initialize a Git repository:

git init

Create a sample file:

echo "Hello Git!" > README.md

Add the file to the staging area:

git add README.md

Commit the file:

git commit -m "Initial commit"

1. Viewing Commit History

To see the list of commits:

git log

Press `q` to exit the log view.

For a compact view:

git log --oneline

2. Undo Changes

A. Undo Last Commit (Keep Changes Locally)

If you want to undo the last commit but keep the changes in your working directory:

git reset --soft HEAD~1

`HEAD~1`: Refers to the previous commit.

B. Undo Last Commit (Remove Changes Completely)

If you want to undo the last commit and discard changes:

git reset --hard HEAD~1

3. Reverting a Commit

To create a new commit that undoes the changes made in a previous commit:

git revert <commit-hash>

Example:

git revert a1b2c3d4

This is safer for shared repositories as it doesn't rewrite history.

4. Checkout to a Previous Commit

Temporarily move to a previous commit:

git checkout <commit-hash>

Note: This puts you in a detached HEAD state. To return to the latest commit:

git checkout main

5. Reset to a Specific Commit

To reset the repository to a specific commit and keep changes:

git reset --soft <commit-hash>

To reset and remove changes:

git reset --hard <commit-hash>

6. Stashing Changes

Save uncommitted changes temporarily:

git stash

Apply stashed changes later:

git stash apply

View stashes:

git stash list

7. Discard Unstaged Changes

Discard changes to a specific file:

git checkout -- <file-name>

Discard all unstaged changes:

git checkout -- .

8. Restore Files

To restore a specific file to the last committed state:

git restore <file-name>

To restore all files:

git restore .

9. Reset Staged Files

To unstage a file:

git reset <file-name>

To unstage all files:

git reset

10. Delete Untracked Files

To view untracked files that would be deleted:

git clean -n

To delete untracked files:

git clean -f

Caution

Use `git reset --hard` and `git clean -f` with care as they permanently delete changes.

Connecting to GitHub

Objective: Push the local repository to GitHub.

Steps:

Log in to GitHub and create a new repository (e.g., `my-first-repo`).

Copy the repository URL.

Link the local repository to GitHub:

git remote add origin <repository-url>

Push the code to GitHub:

git branch -M main

git push -u origin main

Cloning a Repository

Objective: Learn to clone an existing repository.

Steps:

Choose a repository (e.g., your demo repository).

Clone it:

git clone <repository-url>

Make a small change, commit, and push.

Pulling Changes

Objective: Update the local repository with changes from GitHub.

Steps:

Modify the repository directly on GitHub.

Pull the changes locally:

git pull