Lab 7

Git & GitHub

(Branching, Merging, Cloning, and Pull Requests)

Objective

Providing hands-on experience in using Git for **version control** and GitHub for **collaborative** work. By the end of the lab, students will understand how to:

- Initialize a Git repository and track file changes.
- Work with basic and intermediate Git commands.
- Collaborate on GitHub by creating repositories, cloning, and working with branches and pull requests.

Introduction

Git

A distributed **version control** system that allows multiple people to work on a project without overwriting each other's changes.

GitHub

It is a web-based hosting service for Git repositories, making it easy to **share** code, **collaborate** with others, and manage project **versions**.

Prerequisites

- Ensure Git is installed on your machine. You can verify the installation by typing:

  git --version

- Create a GitHub account if you don’t have one.

Setting Up Git

- Configure your name and email in Git (these details are used to attribute commits).

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

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

Creating a Repository

- Initialize a local Git repository:

  git init

- Create a new file (e.g., `README.md`) and add some initial content:

  echo "OS-LAB-07" > README.md

Intermediate Git Commands

Tracking Changes:

- Add files to the staging area:

  git add os_lab_07

Note: ***os_lab_07*** *is the name of the file to be added*

- Commit staged changes to the repository:

  git commit -m "my first commit in lab_07"

Viewing History:

- View the commit history:

  git log

Branching

- Create a new branch:

  git branch branch-os

Note: A branch with name ***branch-os*** will be created.

- Switch to the new branch:
  git checkout {branch-2-os}

Note: If branch ***branch-2-os*** doesn't exist already, it will be created automatically.

Merging

- Switch back to the **main** branch and merge changes from the other branch:

  git checkout main

  git merge {branch-task-01}

GitHub

Pushing to GitHub:

- Create a new repository on GitHub.
- Link the local repository to the remote GitHub repository:

  git remote add origin {https://github.com/username/repository-name.git}

*Note****: "https://github.com/username/repository-name.git"*** refers to your's remote repository/folder you made on your GitHub account

Push changes to GitHub:

git push -u origin main/master

Cloning a Repository:

- Clone an existing repository from GitHub:

  git clone {https://github.com/username/repository-name.git}

Pull Requests:

- After making changes on a branch, push the branch to GitHub.
- Open a pull request on GitHub to merge the changes from your branch into the main branch.

Exercises

Exercise 1: Initial Commit

1. Create a new Git repository in a directory named `**OS-LAB-07**`.

2. Create a file `**threads.txt**` and add some initial content from previous labs.

3. Add the file to the staging area and commit the change.

Exercise 2: Branching and Merging

1. Create a new branch called `**feature-multi-threading**`.

2. On the ` **feature-multi-threading**` branch, create another file `**mutex.txt**` and add some content.

3. Commit the changes on this branch.

4. Switch to the `**main**` branch, merge the ` **feature-multi-threading**` branch, and resolve any conflicts.

Exercise 3: GitHub Collaboration

1. Push your repository to GitHub.

2. Invite a **classmate** as a **collaborator**.

3. Work with your collaborator to create a pull request that adds a new file.

Summary

- Version control and branching.
- Collaborating through GitHub.
- Managing pull requests and resolving merge conflicts.

Additional Resources

Git Documentation: Git Documentation

GitHub Guides: GitHub Guides