Abstract

This lab report into the fundamental concepts of Git & GitHub, aiming to address the need for efficient version control & collaborative software development. These research explores the reasons behind conducting the experiment and problems associated with traditional versioning methods. The result signify the power of git & github in providing a streamlined development process improved collaboration and a robust version control mechanism. The finding enhance the understanding of version control concepts & demonstrate the tools role in addressing collaboration challenges, organize software development practices.

Introduction to Git & GitHub :

Git and GitHub have emerged as indispansable tools in the realm of software development, revolutionizing the way teams collaborate, track changes and manage codebass Git, a distributed version control system, provides a robust framework for tracking modifications to source code, allowing for efficient collaboration among developers. On the other hand github, a web-based platform, serves as a centralizes hub fo hosting, responsibilities, fostering collaborati through features like pull requests, issue tracking and seamless integration with git work flows.

Here is the 25 command's of a Git :

① git init : Initializes a new git repository.

② git clone<repository> : Create a copy of a remote repository on your local machine.

③ git add <file> : Stage changes for the next commit.

④ git commit -m" message" : Records changes to the ~~respos~~ repository with a descriptive m

⑤ git status : Display the status of changes as untracked, modified or staged.

⑥ git diff : shows the difference betwee working directory, staging areas the last commit.

CamScanner

(7) git log: List commit history, including commit msg and SHA-1hasses.

(8) git branch: Lists all branches in repository.

(9) git branch <branch_name>: Create new branch.

(10) git merge <branch_name>: Integrates changes from one branch into another.

(11) git checkout <branch_name>: Switched to the specified branch.

(12) git remote -v: All remote repositories associated with the current repository.

(13) git pull <remote> <branch>: Fetches changes from a remote repository and merge them into the current branch.

(14) git push <remoate> <branch> : Pushes local commit to remote repository.

(15) git fetch : Retrives changes from a remote repository without merging.

(16) git reset : Unstages changes preserving modifications in the working directory.

(17) git revert <commit> : Creates a new commit that undoes changes made in a previous commit.

(18) git rm <file> : Removes a file from both the working directory and the staging area.

(19) git tag <tag_name> : Creates light weight tag to lebel specific points in history.

(20) git stash : Temporarly saves changes that are not ready to be commited.

(21) git remote add <name> <url> : Adds a new remote repository

(22) git remove <name> <url> : remove repository.

(23) git config --global user.name "Your Name" : sets the author name to be used for all commits.

(24) git config --global user.email : set author email to be used for all commits.

(25) git log --graph --online --all :

Displays a concise graphical representation of commit history.

Method and Materials:

In this lab, i have taken the help of PDF while ~~p~~ reporting this, which is provided by the course teacher, And the 20 commands helped by the git cheat sheet.

Activity 1: create Git Repo & txt script

① At first create directory which to set as my repository in my location:

```
$ mkdir Lab1-Assignment
```

** this command is should be in documents/Lab1-Assignment.

② Initialize directory as repository:

```
$ git init
$ git config --global init.default Branch main
$ git branch main
```

③ To use config add name and email:

$ git config --global user.name "trina"

$ git config --global user.email "saniarysnin28@gmail.com"

④ Create a txt script in my directory which create 2 txt script:

Home-work_1

Home-work_2

⑤ Inside the file, code to print text "Hello Guys"

Printf ("Hello Guys")

⑥ Inside the file, code to print text "Hello world"

printf ("Hello world")

⑦ All this txt script main branch and commit this script.

$ git add Home_work_1.text

$ git commit -m "Home_work_1 file added"

⑧ To add this file into GitHub main branch, which is link to GitHub account.

```
$ git remote add origin https://github.com/
trina/21040102022 59_
```

*this link from github which is you create

```
$ git branch -M main
$ git push -u origin main
```

⑨ Now add 2nd txt script into main branches (GitHub)

```
$ git add Home_work_2.text
$ git add
$ git commit -m "Home_work_2 file added"
$ git push -u origin main
```

Activity : 2 create Branches and Merge into main branch

① Create newBranch and create txt script into directory which is already created (Lab1-Assignment)

```
$ git checkout -b newBranch
$ git add
$ git commit -m "New_Home_work file added"
```

② Push this script into main Branch & Switched into main and merge branch into main Branch !

```
$ git push -u origin main
$ git push -u origin newBranch
$ git checkout main
           switched into branch 'main'
$ git merge newBranch
$ git push -u origin main
```

Discussion:

In this lab, I faced a lot of problems in the firest day. I also made spelling mistake because of that there were repeated error. Finally i overcome all these problems.

Conclusion:

The exploration of Git and github in this report illuminated their pivotal roles in modern software development. The robust version control capabilities of git, coupled with the collaboration features offered by GitHub symbiotic relationship that enhances project management and fostens efficier teamwork.