INTRODUCTION TO GIT AND GITHUB

By

Afifa Hoque Tisha
ID: 2104010202237

CSE306: Software Engineering and Information System Design Lab

Instructor:

MD. Tamim Hossain
Lecturer
Department of Computer Science and Engineering
Premier University

Signature

Department of Computer Science and Engineering
Premier University
Chattogram-4000, Bangladesh
22 November,2023

Abstract: The lab was to introduce us with Git & GitHub. where we learned fundamentals of version control using Git and collborative development using GitHub.

Introduction:

Git: Git is a version control system that tracks changes in software development. It enables collaborative work, allowing multiple developers to manage and merge code effecientlly.

GitHub: GitHub is a web based platform utilizing Git for collaborative software development, providing features like repository hosting, pull request and issue tracking. It serves as a centralized hub for managing and sharing code projects.

Materials:

- personal computer with Git installed
- Instructor's lab manual.

Useful commands for Git & GitHub:

1) Git Init: Initializes a new git repository

2) git add [file]: Adds changes to the staging area.

3) git commit -m "[message]": commits changes with a descriptive message.

4) git Status: Displays the status of changes as untracked, modified or staged.

5) git branch [branch_name]: Creates a new branch.

6) git checkout [branch_name]: Switches to a specified branch.

7) git merge [branch_name]: Merges changes from one branch into the current branch.

8) git pull origin [branch name]: Fetches changes from a remote repository and merges them.

9) git push origin [branch name]: pushes local changes to remote repository.

10) git remote -v : Lists remote repositories.

11) git log : Displays commit history.

12) git clone: clone a repository from remote file.

13) git diff: shows changes between commits, branches or files.

12) Git clone: Clone a repository from remote file.

14) git tag [tag_name]: Creates a lightweight tag for a specific commit.

15) git stash : Temporarily saves changes not ready for commit.

16) git remote add [Name] [URL]: Adds a new remote repository.

17) git fetch [remote-name]: Retrieves changes from a remote repository.

18) git reset [file]: Unstages changes in the staging area.

19) git rm [file]: removes a file from both the working directory and staging area.

20) git branch -d [branch name]: Deletes a local branch

21) git pull -- rebase origin [branch_name]: Fetches and rebase changes from a remote repository.

22) git cherry pick [commit hash]: Applies a specific commit from one branch to another.

23) git log -- graph -- oneline -- all: Display a compact, graphical representation of branch history.

24) git revert [commit hash]: create a new commit that undoes changes made in previous commit.

25. git submodule add [repository URL]: Adds a git submodule to the repository.

Activity:

Step 1: Create a git repository.

- create a directory in local disk D. as ~~labreport101~~, ~~labreport102~~. SEISD
- Initialize the directory as repository

```
>$ git init
>$ git config -- global.init.defaultBranch main.
>$ git branch -m main.
```

- use config to add name and email

```
>$ git config -- global user.name "Afifa Tapque"
```

>$git config --global user.email "afifahoque57@gmail.com.

$ • Create a new file. name "labreport1.txt".

Step 2: Create a text script.

Step 3: • add files to stagging.

> $git status
> $git add labreport101.txt
>$git status.

• commit files in staging.
> $git commit -m "commit labreport101"
>$git log (to see what commit look like)
>$git status.

step 4: add file to github
>$git remote add origin (URL)
> $ git branch -m main
> $ git push -u origin main

Step 5: creat new branch.
> $ git Checkout -b Newbranch.
> $ git push -u origin newbranch

Step 6: Merge (new branch in main branch)
> $git Checkout main
> $ git merge newbranch.
> $ git push -u origin main

Step 7: Adding a file in github: Now add a file in Github. >$ git pull.
Then "github file will come in SEED

Discussion: The introduction to Git and GitHub proved to be a pivotal learning experience for participants, providing us with a solid foundation. while doing the lab I faced a problem when a pushed before committing, but my instructor helped me to overcome it.

Conclusion: The Git and Github introduction lab successfully achieved its objectives by equipping us with practical skills in version controls and collaborations development. The positive engagement and understanding demonstrated by us underscored the importance of integrating such hands on experience into technical education