Abstraction:- This report is all about using Git and Github to work together on projects. Think of Git like a time machine for files, helping us keep track of changes. GitHub is like a teamwork hub where we share and improve our work together. The report breaks down how to use Git and GitHub in simple steps. It shows how to save our work, suggest changes, and talk about tasks using GitHub features. It also explains how these tools make it easier to deal with challenges and keep the work going smoothly. In short, this report is a guide to using Git and GitHub for teamwork, making project easy and organized.

Introduction:- Git is a popular version control system. It was created by Linus Torvalds in 2005. It is used for Tracking code changes, Tracking who made changes, Coding collaboration. Git can be accessed via a command line or through a desktop App with a graphical user interface, such as Sourcetree. GitHub is an online software development platform. It's used for storing, tracking, and collaborating on software project. It makes it easy for developers to share code files and collaborate with fellow developers on open source projects. GitHu

P.T.O

also serves as a social networking site where developers can openly network, collaborate and pitch their work.

Materials: Git, GitHub, Web CromeBrowser, Notepad.

Activity-1/. Create a git Repository

1) Create a directory in local disk D.

```
$mkdir SEIS LAB1 5C TASK
cd SEIS LAB1 5C TASK
```

2) Initialize the directory as a repository.

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

3) Use config to add name and email.

```
$git config --global user.name "Habib"
$ git config --global user.email "habibulbasenol44@gmail.com"
```

4) Create a new file. name 'New1.txt'
I used notepad and save it in my git repository

Activity-2/: Create a txt script that prints "This is New1 file".

Activity-3/ ⓐ Add file to stagging.

$ git status (get used to using this command as it helps to verify which stages the files are in)

$ git add New1.txt

$ git status

ⓑ Commit files in staging:

$ git commit -m "Commit New1 file".

$ git log (to see what our commit looks like)

$ git status

ⓒ ~~Ch~~

~~Activity-4: Create New Branch,~~

~~$ git checkout -b NewBranch1~~

~~$ git push -u origin NewBranch1~~

Activity-4/ Now add Github,

$ git remote add origin https://github.com/HBSAIKAT/2254 5C-HABIB.git

$ git branch -M main

$ git push -u origin main

Activity: 5/ Create New Branch

```
$ git checkout -b NewBranch1
$ git push -u origin NewBranch1
```

(Here New1.txt file push in NewBranch1)

Activity: 6/ Merge. (NewBranch1 ⟶ Main Branch)

```
$ git checkout main
$ git merge NewBranch
$ git push -u origin main
```

Activity-7/ Similar iterative procedure on activity can running for create another branch Activity 5 and Activity 6 can run iteratively.

Activity-8/ Now add a file in GitHub - 'github.'

From GitHub to Local pc bring file by using

```
$ git pull.
```

Then ' github file will come in SEIS LABL 5C TASK folder.

GIT 25 Command-

1) git init → Initializes a new Git repository

2) git clone <repository> → Creates a copy of a remote repository on your local machine.

3) git add <file> → Adds a file on changes to the staging area

4) git commit -m "message" → Commits the changes in the staging area with a descriptive message.

5) git status → Shows the status of changes as untracked, modified, or staged.

6) git diff → Shows the differences between the working directory and the staging area.

7) git log → Displays the commit history.

8) git branch → Lists all branches in the repository.

9) git branch <brance_name> → Creates a new branch.

0) git checkout <branch_name> → Switches to the specified branch

) git merge <branch_name> → Merges changes from the specified branch into the current branch.

12) git pull → fetches changes from a remote repository and merges them into the current branch

13) git push → Pushes changes to a remote repository

14) git remote -v → Shows the URLs of remote repositories

15) git fetch → Fetches changes from a remote repository without merging.

16) git reset <file> → Unstages the specified file.

17) git reset --hard HEAD → Discards all changes in the working directory.

18) git stash → Temporarily saves changes that are not ready to be committed.

19) git tag <tag_name> → Creates a new tag for the current commit

20) git show <tag_name> → Shows the details of a specific tag

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

22) git remote rm <name> → Removes a remote repository

23) git log --graph --oneline --all → Displays a concise and graphical representation of the commit history.

24) git rm <file> → Removes a file from both the working directory and the git repository

25) git config --global user.name "Your Name" → Sets the global username for your Git commits.

Discussion:— In the Git and GitHub lab, ~~I~~ began by creating a new directory, learning how to initialize it with Git. ~~I~~ grasped the concept of adding a text file to ~~the~~ this directory, committing changes, and pushing them to our GitHub repository. This process ensures a recorded history of our project development. The lab also delved into the significance of branching, a feature that allows us to work on different aspects of our lab work simultaneously. I practiced creating branches for various features or bug fixes. Through this hands-on experience, I have gained a practical understanding of the fundamental Git and GitHub concepts. These include the importance of version control, organized branching for efficient development, and collaborative workflows. The ability to navigate these processes is a valuable skill set for mine.

Conclusion:- In conclusion, the Git and GitHub lab has equipped us with essential skills for effective version control and collaborative codding. We've learned how to initialize a directory, add and commit changes, and push our work to a GitHub repository. The practice of branching and merging has provided insighs into managing concurrent aspects of a project seamlessly. This lab has underscored the importance of maintaining a clean project history through meanigful commits and organized branches.

Refferences:- Lab Note, W3 School, Chatgpt, Lecture PDF,