Introduction:

Git is a distributed version control system that allows multiple people to collaborate on projects simultaneously. It keeps track to changes made to files, making the it easier to manage different versions and marge changes from different contributors.

GitHub, on the other hand, is a web-based platform built around git. It provides a hosting service for git repositories and adds a collaborative layer on top of Git. Developers use github to store and share code, collaborate, Track issues and Manage Project.

Activities:

- Install and setting up Git on My pc.
- Creating my first Git repository.

once git was installed, I created my first Git repository using the 'git init' command in my project directory. This initialized a local repository where I could track changes to my project file.

- Adding and committing files:

With my repository set up, I began adding files to it using 'git add [filename]' to stage the changes. Then I commited these changes using 'git commit -m "Initial commit"' to create a snapshot.

- Connecting to GitHub;

To leverage the collaborative features to GitHub. I created an account and set up a new repository on the GitHub platform.

- Pushing files to GitHub.

Once the connection was establish, I uploaded my local repository to GitHub by pussling my code using the 'git push -u origin main' command.

- Pulling changes from GitHub.

To understand How changes made by collaborate can be integrated into my Local repository. I used 'git pull origin main' command. This pulling any neue changes from the remote repository on GitHub to my local machine.

Git Commands :-

- Repository Setup :

→ git init (initialize a new git repository)

→ git clone (clone a repository from a remote server to your local server)

- Basic Snapshotting,

→ git add [file] (Add changes in a file)

→ git commit -m "[message]" (Record changes to the repository)

- Branching and Merging:

→ git branch (List all branches in the repository)

→ git branch [branch_name] (create a new branch)

→ git ~~branch~~ checkout [branch_name] (Switch to a different branch)

→ git marge [branch-name] (Marge a specific brand into the current one)

- Comparison :

→ git status (show the current status of file)

→ git Log (Display the commit history)

→ git [file] (show changes between commits)

→ git remote -v (List remote repositories)

→ git remote add [name] [url] (Add a new remote repository)

→ git push [remote] [branch] (Push changes to a (remote repository)

→ git pull [remote] [branch] (Fetch changes from a (remote repository and merge them)

→ git reset [file] (Unstage changes in a file)

→ git checkout -- [file] (Discard changes in a file)

→ git tag (List existing tags)

→ git tag [tag_name] (Create a new tag)

→ git tag -a [tag_name] -m "[message]"

→ git stash (Stash changes in the working directory)

→ git stash apply (Apply the most recently stashed changed)

→ git stash pop (Apply the most recently stashed changed and remove them)

- git go config --global user.name "[name]" (Set your user name)

- git config --global user.email "[email]" (Set your email here).