**Git:**

Helps in maintaining the history of the project and collaborate with different people.

**GitHub:**

It is an online platform helps in hosting our git repositories.

**Steps:**

1. Open cmd in a folder(Prep)
2. *mkdir my🡪*creates a folder with name ‘my’;
3. *cd my🡪*changes the directory to the newly created folder ‘my’
4. *git init🡪*initializing empty git repository.
5. Even after initializing the git repository if we use *dir* command to list the files, we would not be able to see. Because .git file is hidden and inorder to view the hidden files use the command *dir /a.*
6. We can also find what’s inside the .git folder by using the command *dir .git* then we will find something like hooks,heads etc.
7. We can create an empty text file in this folder by using the command *type nul > your\_file.txt*
8. Now we can use the  *git status* command to know about the status of the files whether they are tracked or untracked. The untracked files or changes are not stored in the project history.
9. Now we use *git add file\_name*  command to stage the changes or we can use *git add .* command to stage all the changes at once.
10. Now we use the *git commit –m “message”*  to move the changes from staging area to local repository.
11. Now make some changes to the file (hema.txt) and use *git status* command, now we can see that hema.txt is unstaged.
12. Use *git add .* command to stage the changes.
13. We can also unstage a file without commiting by using the command *git restore -- staged hema.txt.*
14. *git log* command is used to watch the entire history.
15. *del file\_name* command can be used to delete the file.
16. We can also delete few commits and go back to the particular commit. For this first use the *git log* command and copy the hash code of the commit to which you want to go back and now use the command *git reset hash\_code* to get back to the desired state.
17. We can also put some changes on hold without saving them by storing them backstage and get them whenever we need by using the command *git stash.*
18. We can use the *git stash pop* command to get back the changes.
19. We can also use *git stash clear* command to clear all the changes in the stash without getting them back.
20. Now if we want to connect the local repository to remote repository🡪first open the github accout-create a new repository-copy the url of that repo-and now use the command *git remote add origin url* to link to the remote repository.
21. Inorder to push the changes in the remote repository to remote repository we use the command *git push origin master* , here master is the branch name.

**USEFUL COMMANDS:**

1. dir🡪lists all the files in the current directory.
2. dir /a🡪lists all the hidden files in the current directory.
3. mkdir folder\_name🡪creates a new folder.
4. cd folder\_name🡪changes the directory.
5. type nul > your\_file.txt🡪creates an empty text file.
6. git status 🡪 tells about the status of the files.
7. type file\_name🡪displays the contents of the file.
8. git log🡪displays the entire history.
9. del file\_name🡪deleting the file.
10. git reset hash\_code🡪resets back to the desired state.
11. git stash🡪temporarily store the data backstage.
12. git stash pop🡪 to get back the data or the changes.
13. git stash clear🡪to remove the changes in the stash without getting them back.
14. git remote add origin url🡪links to the remote repository.
15. git push origin branch\_name🡪push to the branch.
16. **git fetch –all –prune**
17. **git reset --hard upstream/master**
18. **16 and 17 commands can be replaced with a single command that is git pull upstream main**
19. **Merging comits: git rebase –i hashcode**
20. **Select pick and squash**
21. **Esc : x**
22. **Write the commit message**
23. **Esc : x**
24. **git push --force origin master**
25. **git reset --hard**