**Git And Github Commands**

28/10/22

Git was made for linux so all the linux commands are

ls 🡪 listing the contents in the directory

Ls –l 🡪 long listing of the contents in the directory.

Pwd 🡪 present working directory.

Whoami 🡪 System Name.

Ls –la 🡪 shows long listing along with “hidden fils”.

$ git 🡪 will show git help

$ git init 🡪 to create the repository.

A git “repo” is .git folder that contains a collection of files of various different versions of a Project.

$ git status 🡪 will show untracked files (red color) and (green color) when they are in staging area.

$ git add “filename” 🡪 add file to staging area.

$ git add . 🡪 add all the files in the staging area.

$ git commit –m “” 🡪 m stands for message.

$ git log 🡪 info(author, date, sha number (unique number)) about all the commit.

$ git log –p 🡪 To view the code along with other information of commit.

To remove the git repository just delete .git folder.

Commits are objects in the git.

Every commit has sha number of 40 digits,

Under commit there is a tree which has nodes and files attached to that nodes and commits itself are linked to each other with linked list.

The latest commit has HEAD pointer.

Branch (main/ master)

Tree

Index html (5 lines)

Tree

Head

style

Index html (6 lines)

Keep the branch name as lowercase. The default branch is main/master.

$ git resotre <filename> 🡪 to restore the file deleted which was untracked (red color).

$ git restore --staged <filename> 🡪 to unstage the file.

If we pull the remote repository to local repo.

Before pushing the new files we must first pull the latest updates

Branches

All the branches have there own files and changes done in a branch will not be reflected in another branch until and unless it is merged.

$ git branch 🡪 Branch names and current branch name in green

$ git checkout <<branch\_name>> 🡪 switch into the another branch.

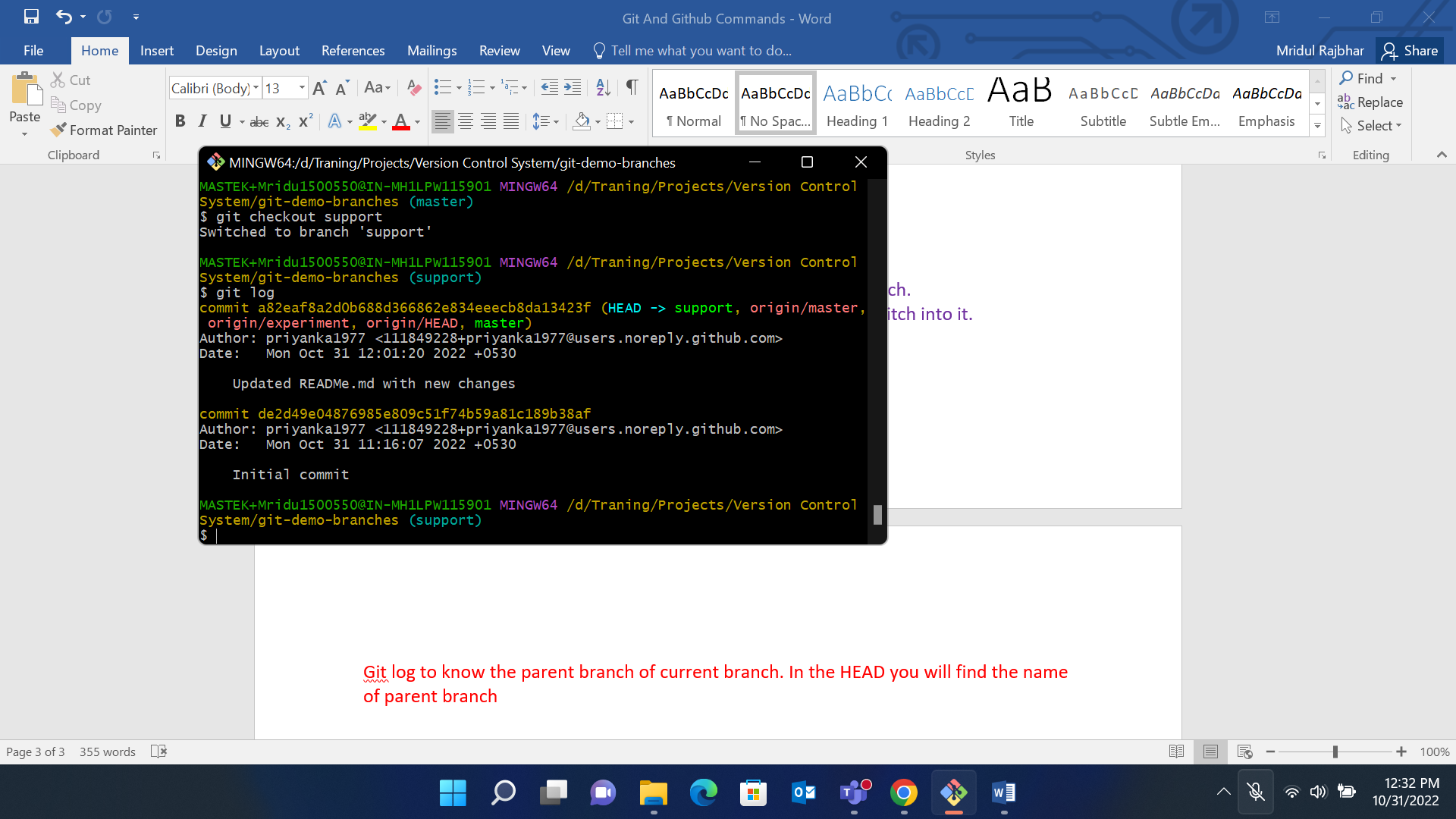
$ git checkout –b <<branch\_name>> 🡪 create the branch and switch into it.

$git branch <<branch\_name>> 🡪 create the new branch.

While we are in the branch we cannot delete it.

The commits of the parent branch are also available in child branch.

Git log to know the parent branch of current branch. In the HEAD you will find the name of parent branch



$ git log –oneline –all 🡪 to log all the commits from all the branches.

Scenario 1: Master branch is not updated with new commits, post creation of child branch

Merge support -------------🡪 master

Step 1: checkout the branch where merging has to be completed.

Git branch

Step 2: Execute the command: git checkout <<branch in which you have to merge>>

Step 3: git merge <<branch you have to merge>>

This is the fast forward merge 🡪 Copy the commit of other branch into the another branch.

The HEAD of the parent branch changes to the HEAD of child branch.

Master (Parent branch i.e the branch in which we have to merge) commits

a82eaf8 (HEAD 🡪 origin/master, origin/experiment, origin/HEAD, master, dev) Updated READMe.md with new changes

de2d49e Initial commit

Support Branch

e056066 (HEAD -> support, support1) Added file file2

addf2cd file1.txt added

a82eaf8 (origin/master, origin/experiment, origin/HEAD, master, dev) Updated READMe.md with new changes

de2d49e Initial commit

After merging support branch to the Master Branch

Go to master branch then execute command

Git merge support

e056066 (HEAD -> master, support1, support) Added file file2

addf2cd file1.txt added

a82eaf8 (origin/master, origin/experiment, origin/HEAD, dev) Updated READMe.md with new changes

de2d49e Initial commit

Scenario 2: Update Master and Support Branches with new commits

Merge with conflicts

Step 1: Checkout to the branch where the merging has to be completed.

Step 2: solve the conflicts in the file.

Step 3: Then commit the conflict.