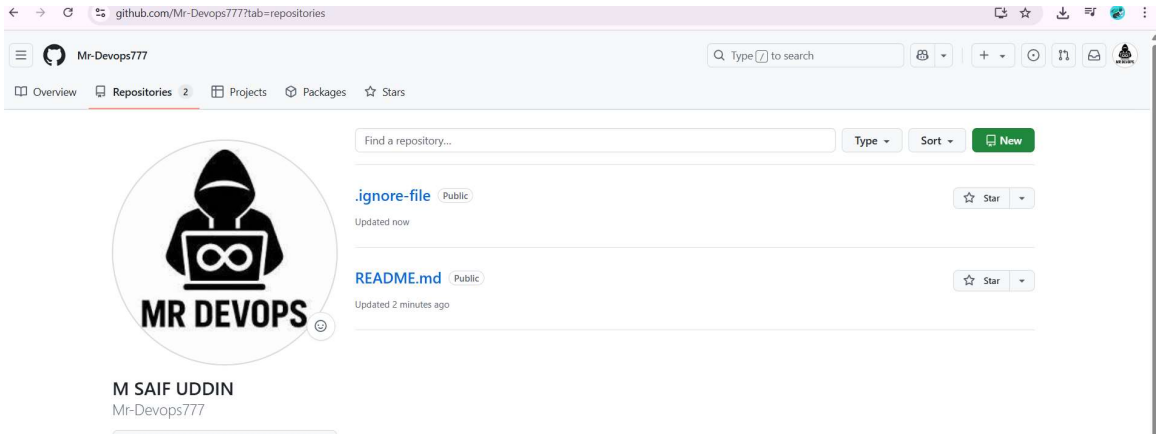
1)Install git.

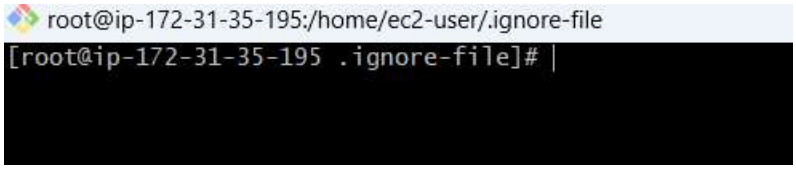
yum install git -y
yum list git

2)Create a repo in github with README.md and .ignore file.



go to github account and click on "+" button and
add repository--name of repo---public or private---create

3)Clone the created repo to local.



git clone "url"

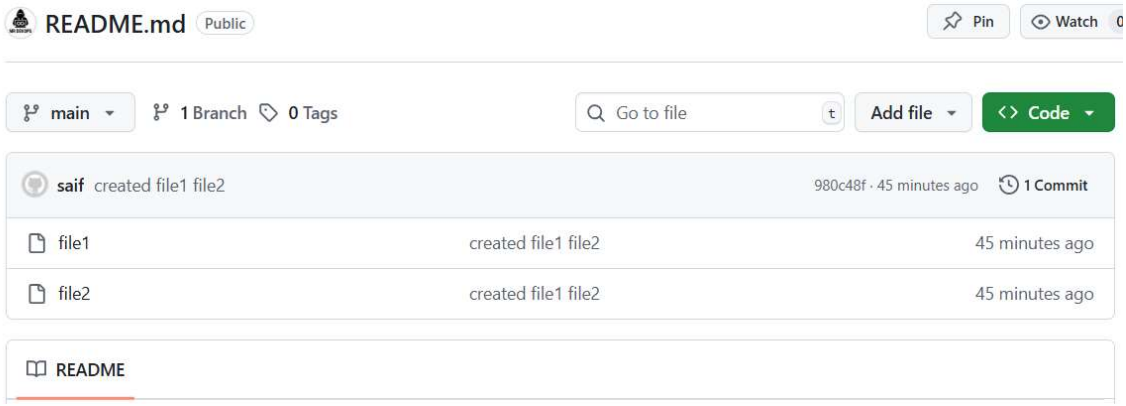
4)Create two files in local repo.

```
root@ip-172-31-42-84:/home/ec2-user/.git/README.md
[root@ip-172-31-42-84 README.md]# ls
file1  file2  file3
[root@ip-172-31-42-84 README.md]# git stauts
git: 'stauts' is not a git command. See 'git --help'.

The most similar command is
        status
[root@ip-172-31-42-84 README.md]# git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file1
        new file:   file2

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file3

[root@ip-172-31-42-84 README.md]#
```

mkdir newdir
git init
touch file1 file2
git add .

git status

5)Commit two files and push to central Repository.



ssh-keygen

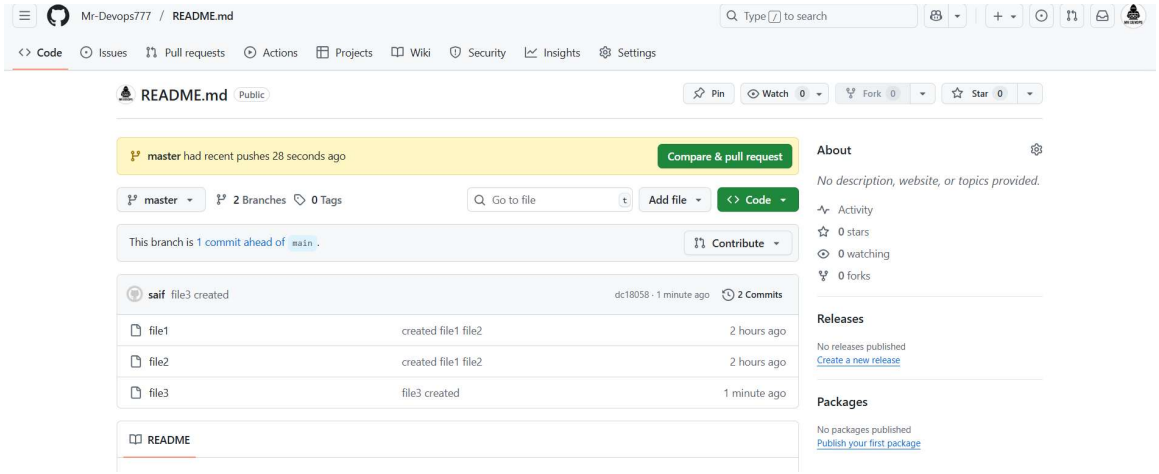
cat ~/.ssh/id_rsa.pub

copy public key and paste on github-setting-ssh&GPG-add ssh

git remote set-url origin git@github.com:Mr-Devops777/README.md.git

git push -u origin master

6)Create a branch in local and create a sample file and push to central.



git branch master

git checkout master(automaticall copied all data of previous branch)

create new file using touch

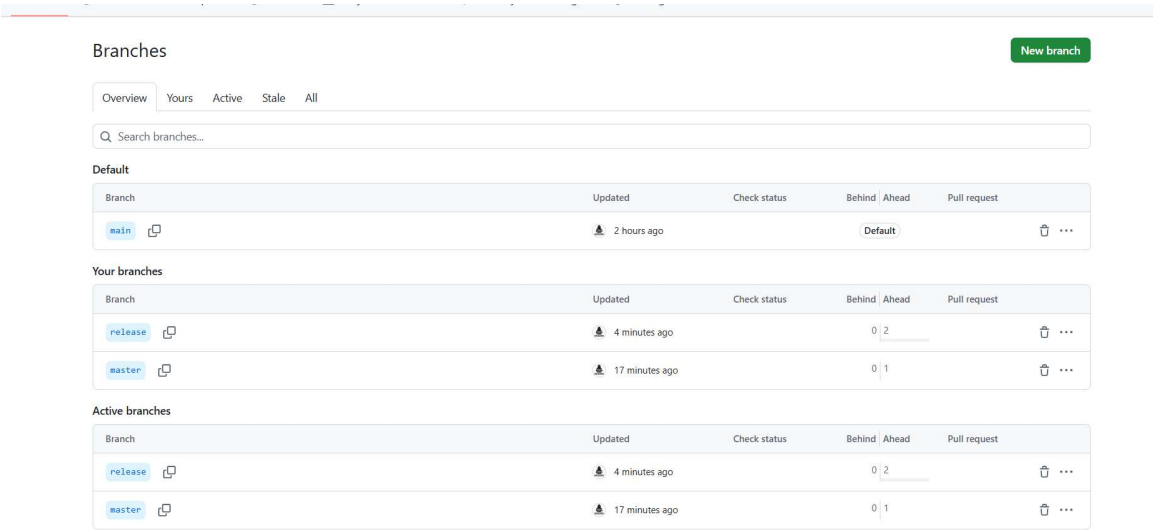
commit add

git status

git push -u origin master

7)Create a branch in github and clone that to local.

```
[root@ip-172-31-42-84 README.md]# ls
file1  file2
[root@ip-172-31-42-84 README.md]# git branch --list
* main
[root@ip-172-31-42-84 README.md]# git checkout release
branch 'release' set up to track 'origin/release'.
Switched to a new branch 'release'
[root@ip-172-31-42-84 README.md]# ls
file1  file2  file3  file4
[root@ip-172-31-42-84 README.md]#
```



created new branch "release" on github
created a file

git clone --branch <branch name> --single-branch <url>

git clone "repo url"

git checkout release

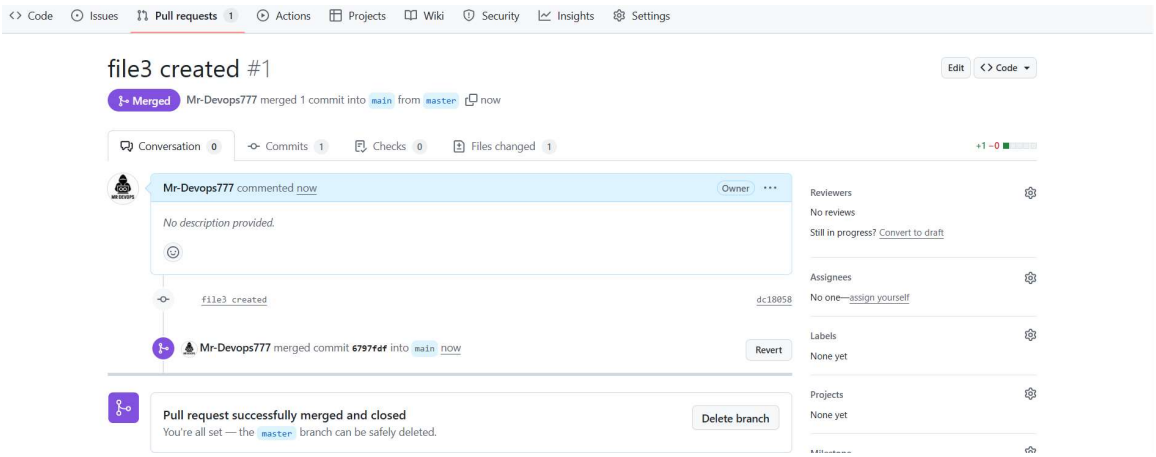
ls

8)Merge the created branch with master in git local.

git checkout master

git merger release

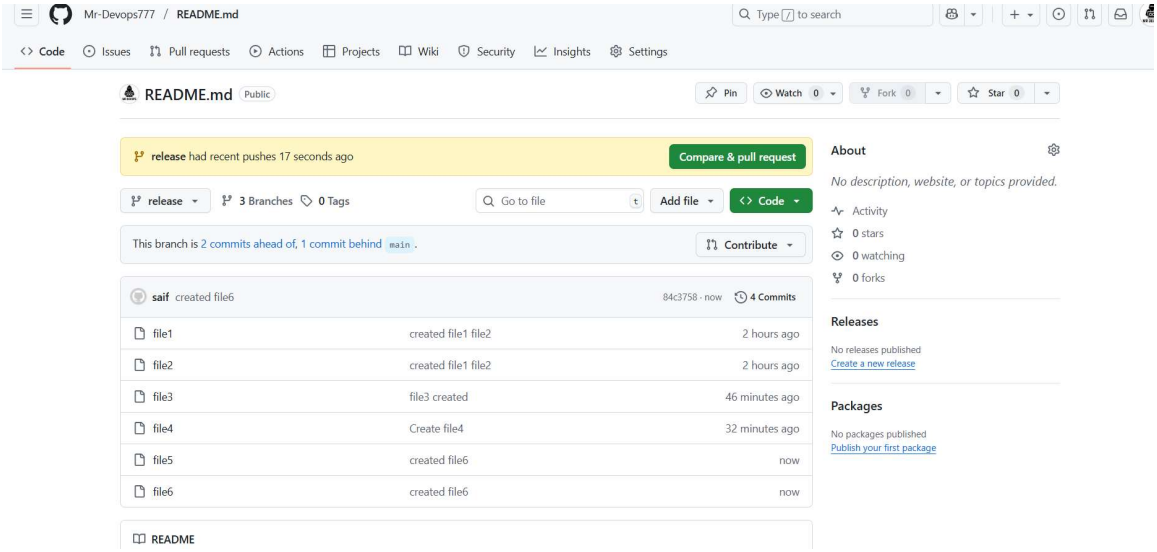
9)Merge the created branch with master in github by sending a pull request.



go to pull request under any repo
README.md--pull request--new branch--select branch to merge

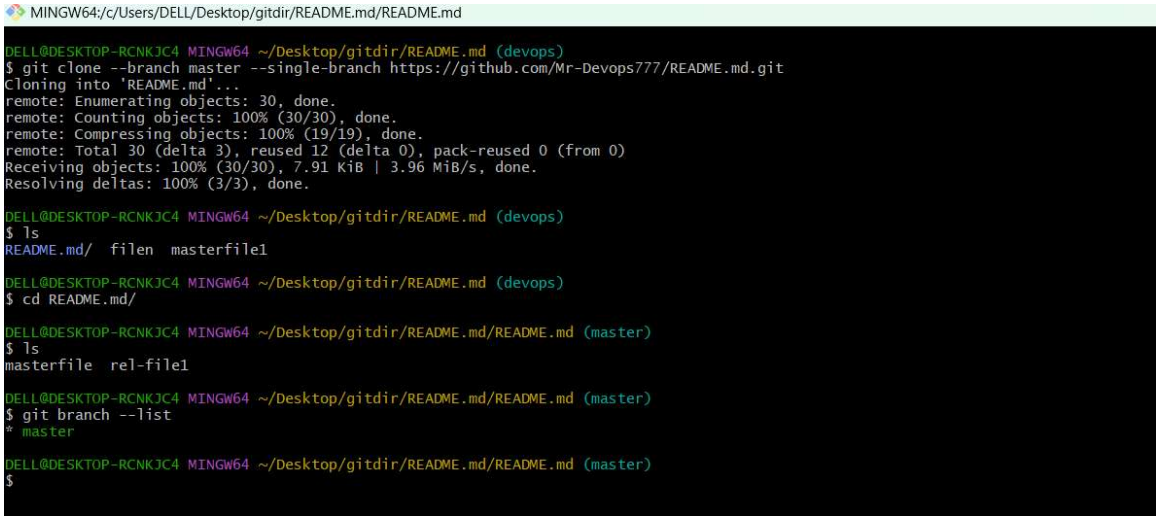
commit the message and pull

10)create a file in local and send that to branch in github.



git push

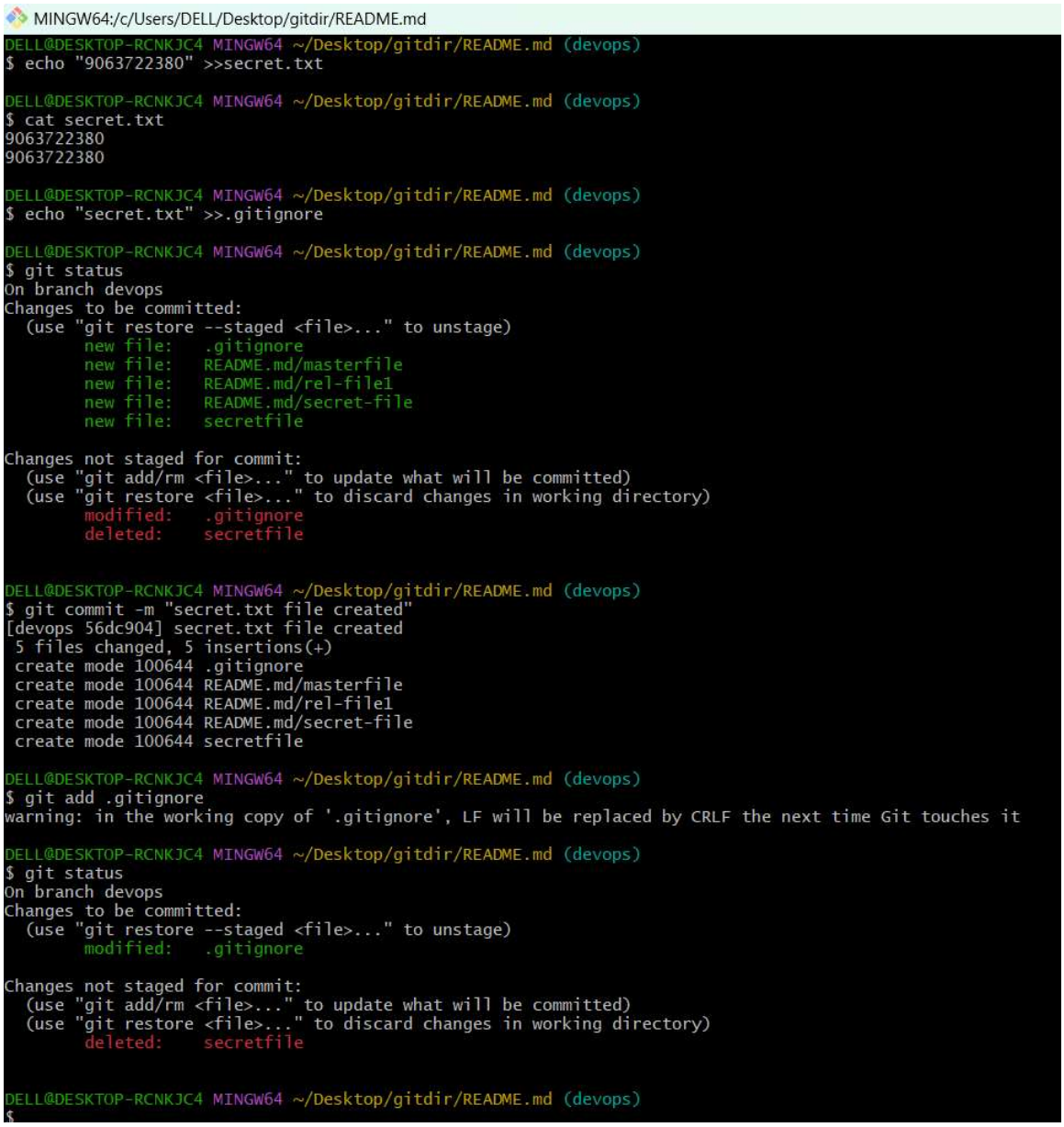
11)clone only a branch from github to local.



git clone --branch <branch name> --single-branch <url>

(to copy only a single branch we use the above command)

12)create a file with all passwords and make that untrackable with git.



echo "9063722380">>secret.txt

echo"secret.txt">>.gitignore

git commit -m "secret.txt created"

git add .gitignore

git status

13)make a commit and make that commit reset without savings changes.

```
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git status
On branch release
Your branch is ahead of 'origin/release' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ echo "first reset">reset-file

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ ls
file1-release  reset-file  revert.txt  sample-release

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git status
On branch release
Your branch is ahead of 'origin/release' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   reset-file

no changes added to commit (use "git add" and/or "git commit -a")

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git add reset-file
warning: in the working copy of 'reset-file', LF will be replaced by CRLF the next time Git touches it

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git status
On branch release
Your branch is ahead of 'origin/release' by 1 commit.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   reset-file


DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git commit -m "first reset in reset-file"
[release 5c22525] first reset in reset-file
 1 file changed, 1 insertion(+), 1 deletion(-)
```

```
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git commit -m "first reset in reset-file"
[release 5c22525] first reset in reset-file
 1 file changed, 1 insertion(+), 1 deletion(-)

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git log --oneline
5c22525 (HEAD -> release) first reset in reset-file
e56881c Revert
2c29f9d (origin/release) second added revert.txt
d3a1f7c first revert.txt
c3b1346 Revert "added reset.txt"
b92c8a6 added reset.txt
4ffbbf8 added reset file
f9c439e (origin/main, origin/HEAD, main) added sample-release file
19211b0 Create file1-release
49a2c20 Delete file2.txt
54bf593 Delete file1.txt
3480bc2 Delete README.md
cf7902f third commit updated file1.txt
64084e0 second commit updated file1.txt
54cd96b first commit added file1.txt file2.txt
603e667 Initial commit

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git reset e56881c
Unstaged changes after reset:
M       reset-file

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git status
On branch release
Your branch is ahead of 'origin/release' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   reset-file

no changes added to commit (use "git add" and/or "git commit -a")
```

first create a file

echo "first reset">reset-file
git add reset-file

git commit -m "added first reset in reset-file"

git status (it will show the changes in tracking area/staging area)

gitn log --oneline (to check the commits it will show your commit)

to move the file from tracking area to untracking area we use
git reset <one previous commit id>

git log --oneline (it will delete your latest commit)

git status (your file will be in staging area to make it track again we need to add through <git add filename>

then again we need to commit
and hence we can push to central repo

---

git reset previous commit id to latest (it will take by default git reset mixed and kepp the file in staging area)

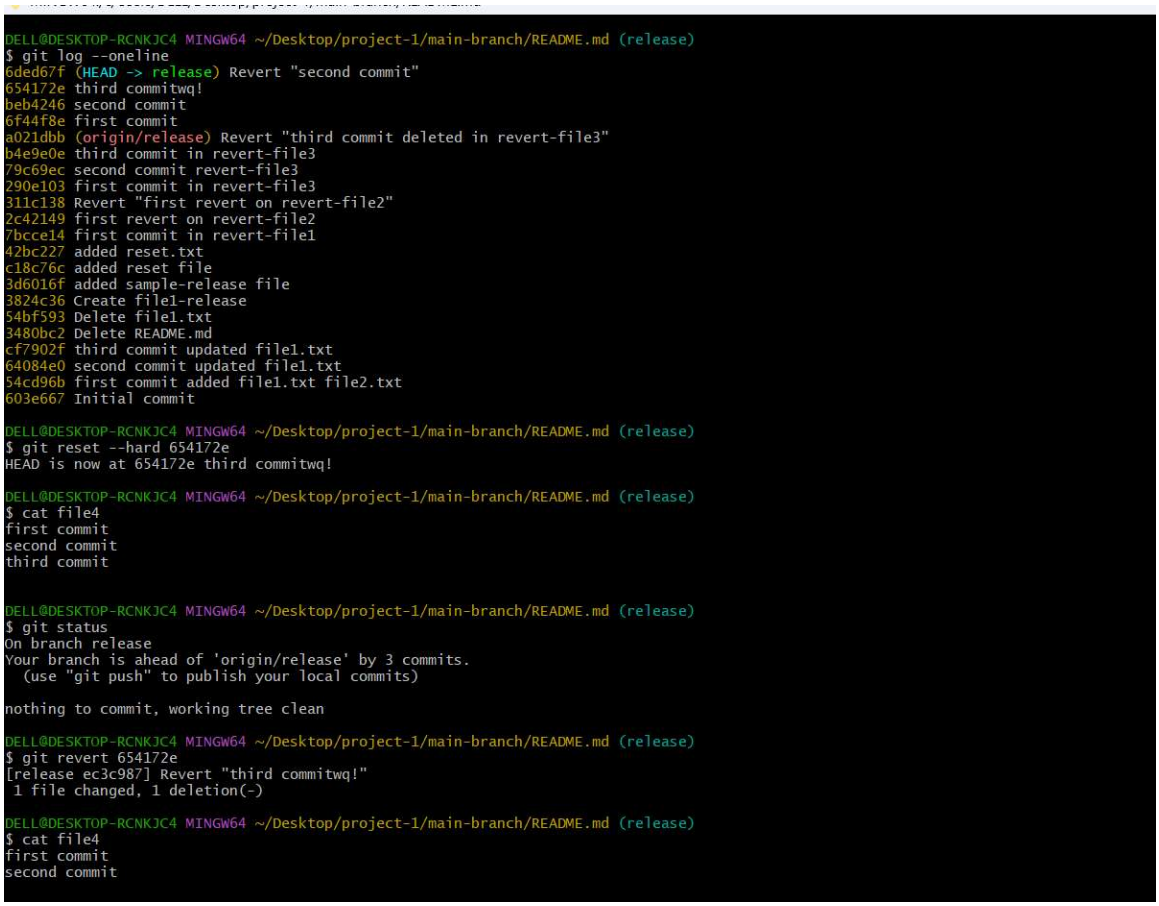
git reset --soft previous commit id (delete the commit and keeps the changes in staging area)

git reset --mixed previous commit id (delete the commit and keeps the changes in working area)

git reset --hard previous commit id (delete the changes and also deletes the changes in repo)

git reset command is used in local repo only.it enables to make changes in files and its commits,we cannot modify the commits and changes if pushed to central repo(for these we need to use git revert if pushed to central)

14)Revert a commited commit to the older version.



work flow

echo "first commit">file4

git add file4

git commit -m "first commit"

vi file4 (add second commit in it)

git add file4

git commit -m "second commit"

vi file4 (add third commit in it)

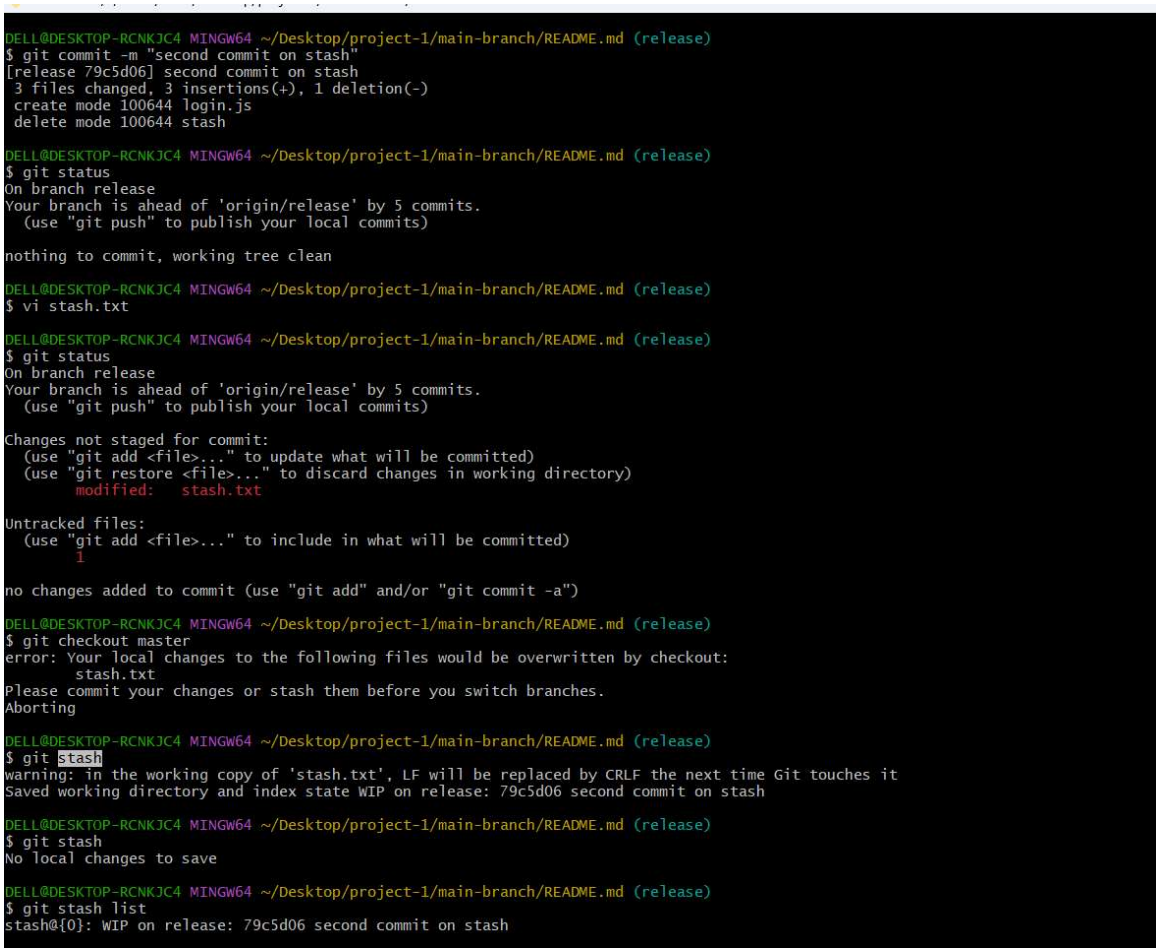
git add file4

git commit -m "third commit"

git log --oneline(to check the commits)

git revert commit_id (it will open the vi editor where you need to save the file simply without overwriting in it.It will delete the previos changes in your file and creates a new commit for these by default)

15)push a file to stash without savings the changes and work on another file.


DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git commit -m "second commit on stash"
[release 79c5d06] second commit on stash
3 files changed, 3 insertions(+), 1 deletion(-)
create mode 100644 login.js
delete mode 100644 stash
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git status
On branch release
Your branch is ahead of 'origin/release' by 5 commits.
(use "git push" to publish your local commits)
nothing to commit, working tree clean
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ vi stash.txt
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git status
On branch release
Your branch is ahead of 'origin/release' by 5 commits.
(use "git push" to publish your local commits)
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
modified: stash.txt
Untracked files:
(use "git add <file>..." to include in what will be committed)
1
no changes added to commit (use "git add" and/or "git commit -a")
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git checkout master
error: Your local changes to the following files would be overwritten by checkout:
stash.txt
Please commit your changes or stash them before you switch branches.
Aborting
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git stash
warning: in the working copy of 'stash.txt', LF will be replaced by CRLF the next time Git touches it
Saved working directory and index state WIP on release: 79c5d06 second commit on stash
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git stash
No local changes to save
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git stash list
stash@{0}: WIP on release: 79c5d06 second commit on stash

```
DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git status
On branch release
Your branch is ahead of 'origin/release' by 5 commits.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        1

nothing added to commit but untracked files present (use "git add" to track)

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ ls
1  file1-release  file2.txt  file4  login.js  reset-file  reset.txt  revert-file1  revert-file3  sample-release  stash-file  stash.txt  task-file

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ vi login.js

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git add login.js
warning: in the working copy of 'login.js', LF will be replaced by CRLF the next time Git touches it

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git add login.js

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git commit -m "added login.js"
[release 02de7e2] added login.js
 1 file changed, 1 insertion(+)

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git stash pop
On branch release
Your branch is ahead of 'origin/release' by 6 commits.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   stash.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        1

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (29b7c21ced0b74eb046dbea89c3860dd94f55777)

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ vi stash.txt
```

```
$ vi stash.txt

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git status
On branch release
Your branch is ahead of 'origin/release' by 6 commits.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   stash.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        1

no changes added to commit (use "git add" and/or "git commit -a")

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git add stash.txt

DELL@DESKTOP-RCNKJC4 MINGW64 ~/Desktop/project-1/main-branch/README.md (release)
$ git commit -m "third commit added to stash.txt"
[release dfefae7] third commit added to stash.txt
 1 file changed, 2 insertions(+)
```

echo "first stash">stash.txt

git add stash.txt

git commit -m "first commit on stash.txt"

echo "second stash">>stash.txt

git add stash.txt

git commit -m "second commit on stash.txt"
echo "third commit">>stash.txt

git status (shows stash.txt in working area)

git stash (to push all files in working area to stashing memory)

git stash push -m (here you can write a message)stash.txt {it will add only stash.txt with a message}

git stash list (it will show the list of files in stash memory)

git status (it will not show stash.txt in working area)

now you can work on any other file and add it and commit it

then switch back to stash using

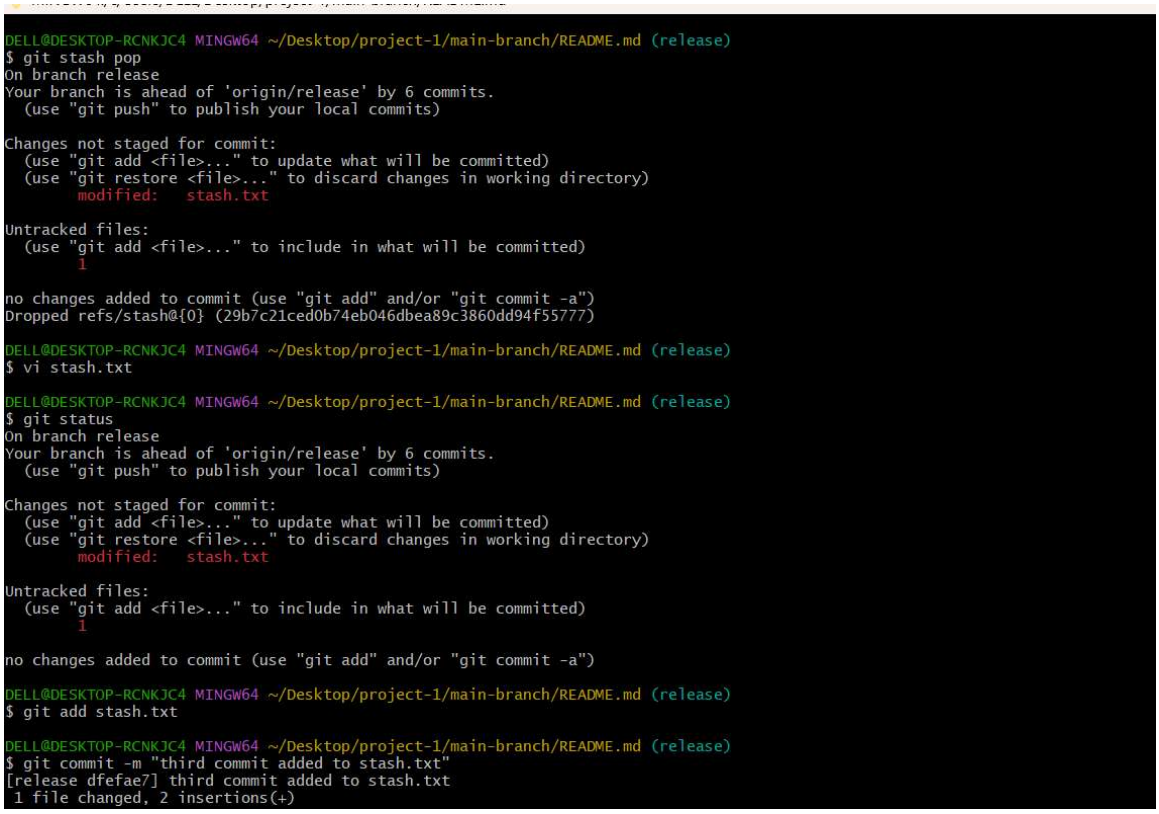
git stash pop

echo "fourth stash">>stash.txt

git add stash.txt

git commit -m "third commit on stash.txt"

16)undo the stash file and start working on that again.



git stash pop stash_id (to move a selected file from stashing to working)

git stash pop (to move all files from stashing to working)

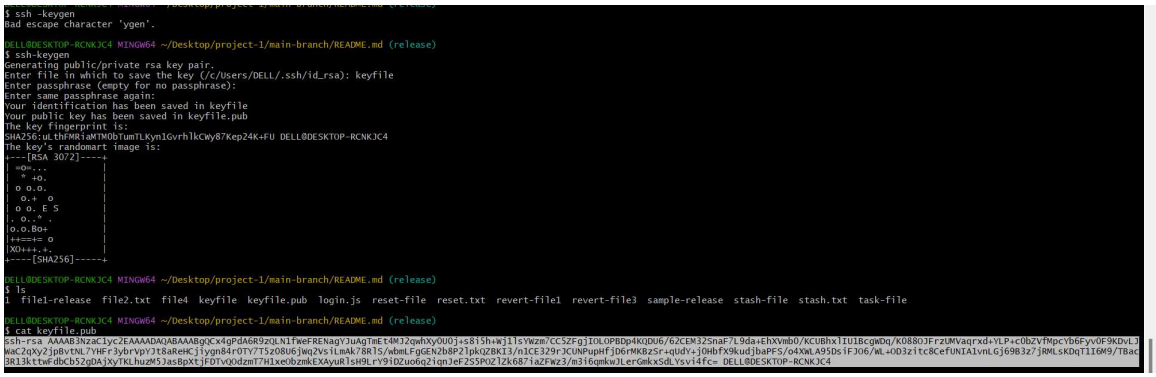
again you can continue your work on file file
then check status

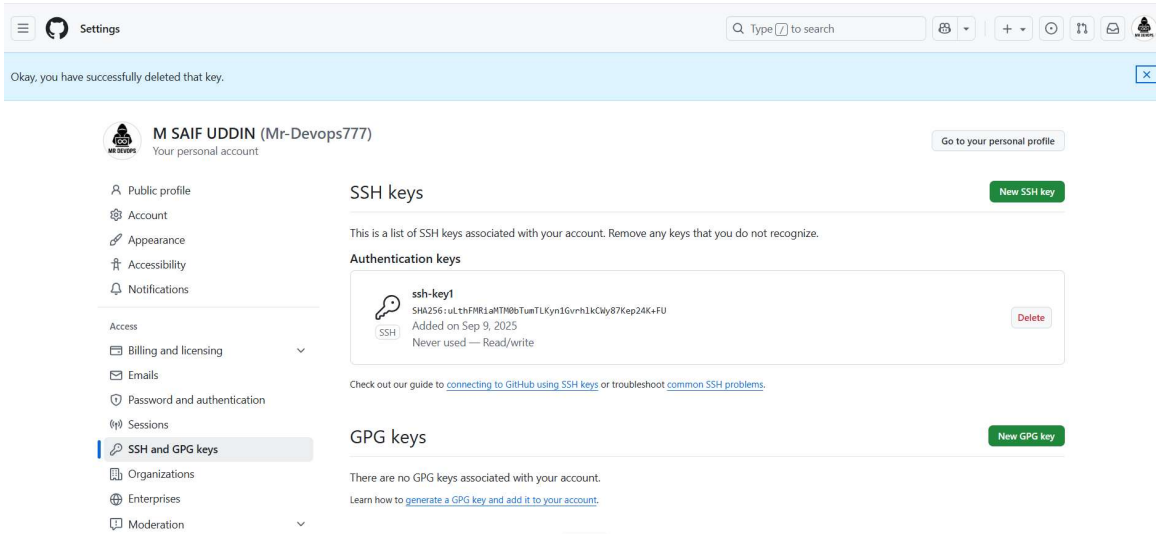
git status (it show the file in working area)

git add stash.txt (to add the file in staging area)

git commit -m "last commit" (finally your file is ready for push to central repo)

17)generate a ssh-keygen and configure into github.





ssh-keygen

define the name after ?keyfile

define the name again after ?keyfile.pub

define again

cat keyfile.pub or cat id_rsa.pub(copy the file code and paste it on)

git-hub/account/setting/ssh and keygen/add ssh key (define name of the ssh key and paste the code here like asjbdsahndsald@#$^&hgsvdhasbjhgs13135154hsdjhgsdu)

18)configure webhooks to github.

19)basic understanding of .git file.

a.(HEAD):

A file that points to the current branch or commit you're working on.

Example:ref: refs/heads/main

b. (config)

Contains repository-specific configuration (like remote URL, username, etc.).

c.(description)

A text file used by GitWeb (not usually important in daily use).

d. (hooks/)

Scripts that run on certain Git events (like before a commit or after a push).

Example: pre-commit hook can check code formatting before allowing a commit.

e . (info/)

Contains exclude file – similar to .gitignore, but applies only to this local repo.

f. (objects/)

Stores all the actual data (commits, trees, blobs, tags).

Each object is saved in a compressed format, indexed by its SHA-1 hash.

Example:

Blob → stores file contents

Tree → stores directories and file structure

Commit → stores metadata + reference to tree & parent commits

g. (refs/)

Contains references to commits.

Subdirectories:

heads/ → local branches (e.g., main, dev)

remotes/ → remote-tracking branches (e.g., origin/main)

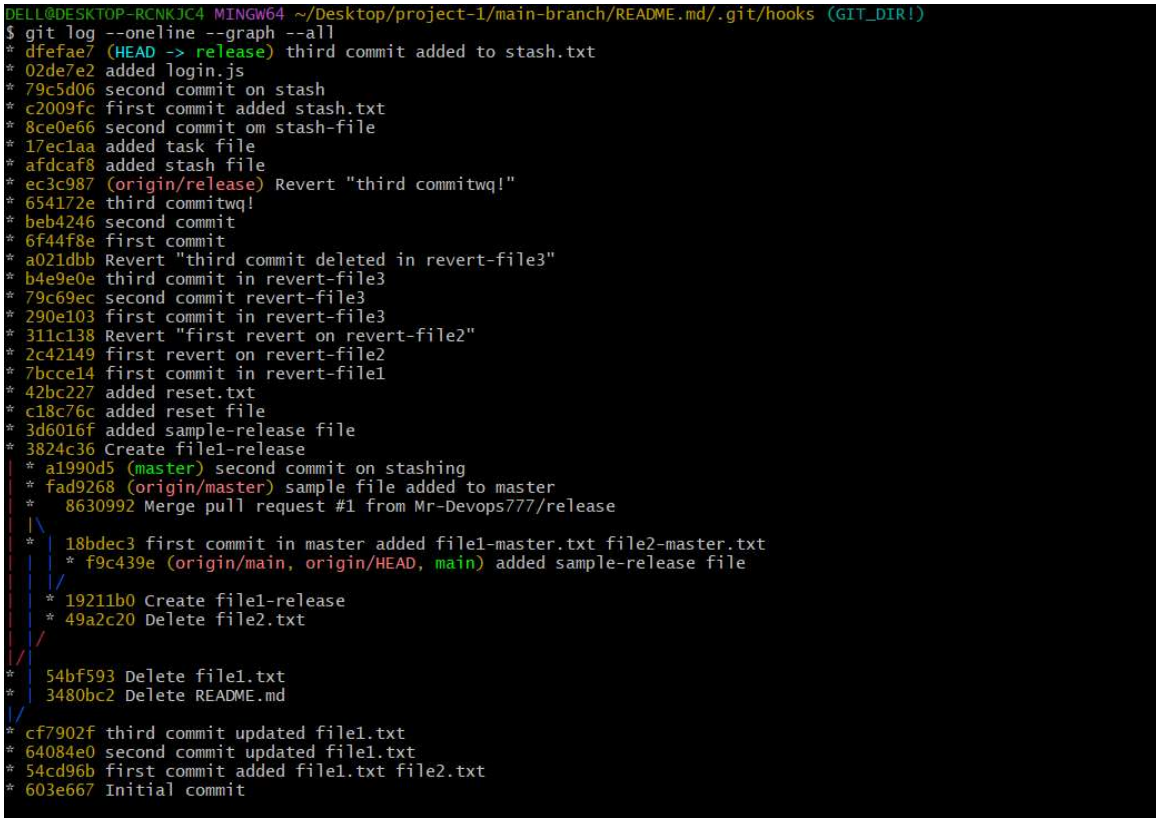
tags/ → tags pointing to specific commits

20)Check all the logs of git.

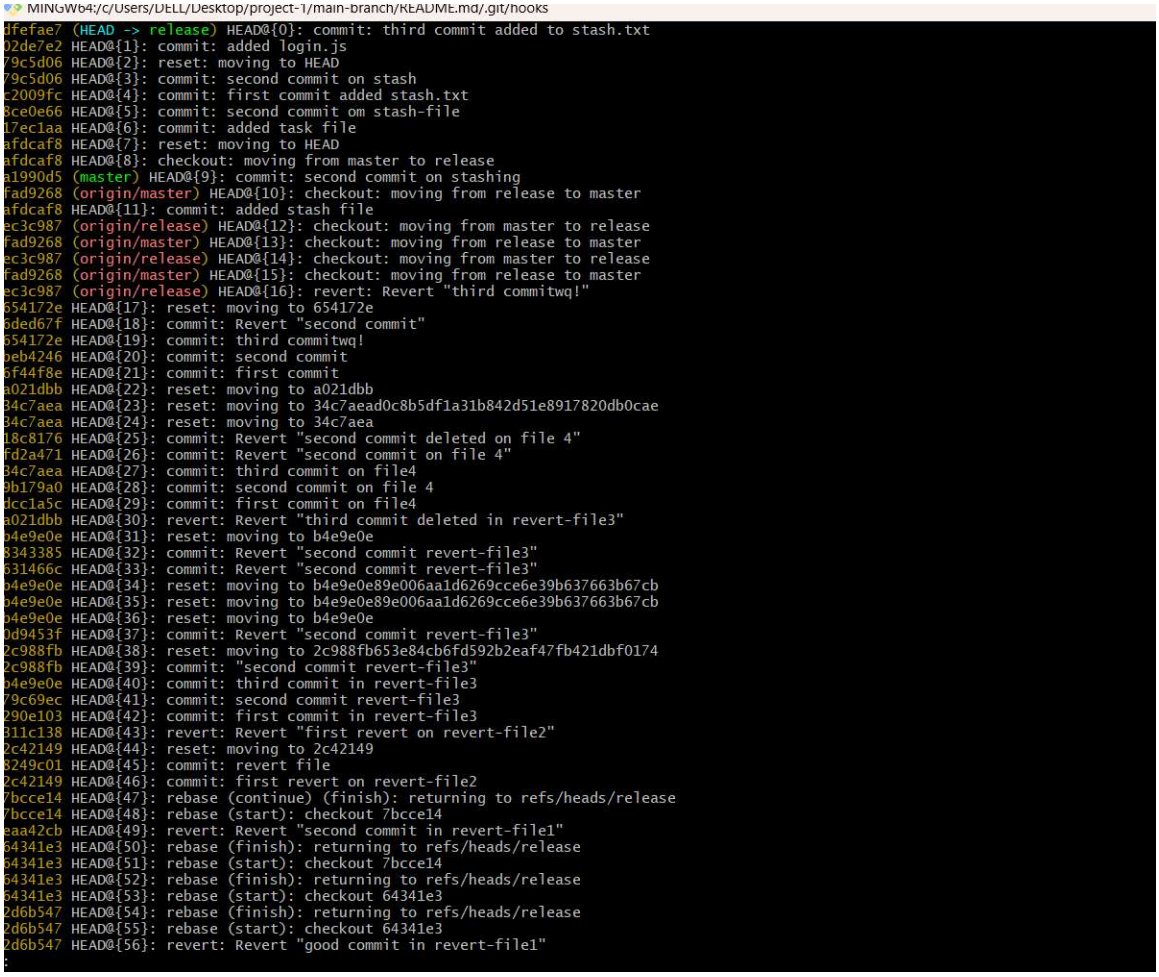
git log → commit history (normal project history)

git log --oneline

git log --oneline --graph --all → pretty tree of all branches



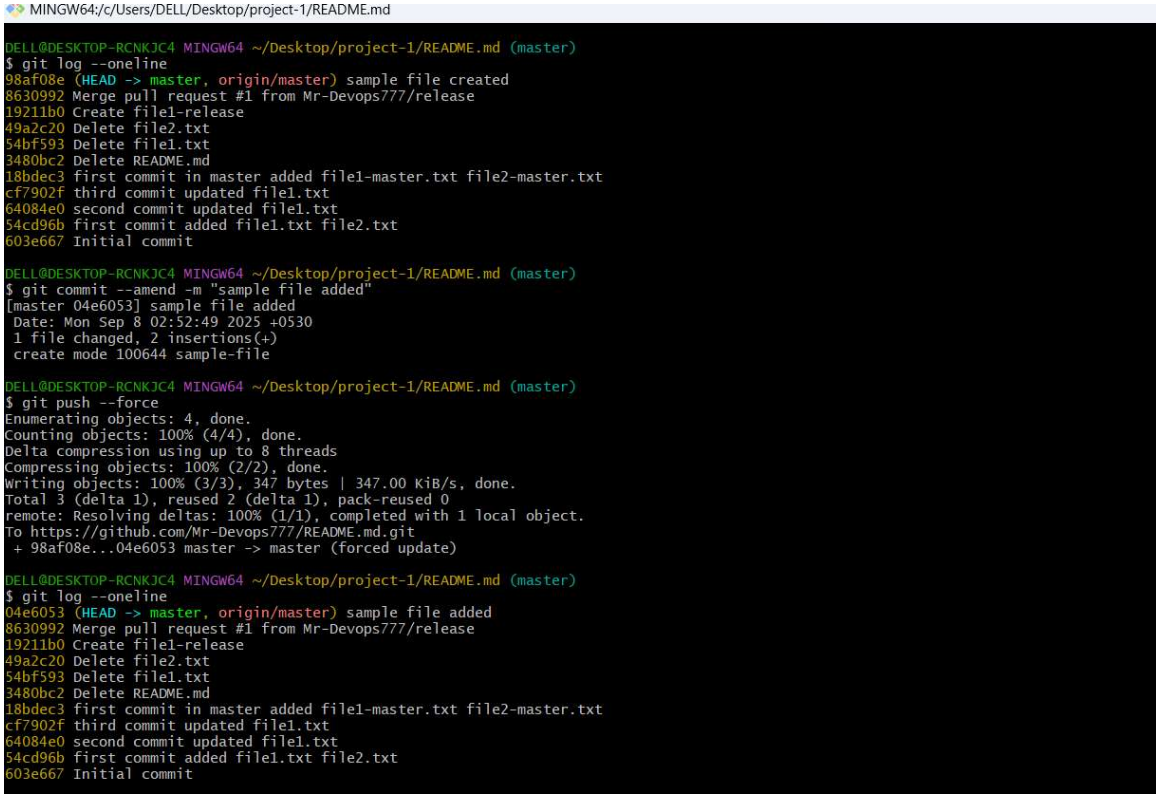
git reflog → everything HEAD has done (recover lost commits)



.git/logs/refs/ → raw logs per branch

git stash list → stash logs

21)Rename the commit message.



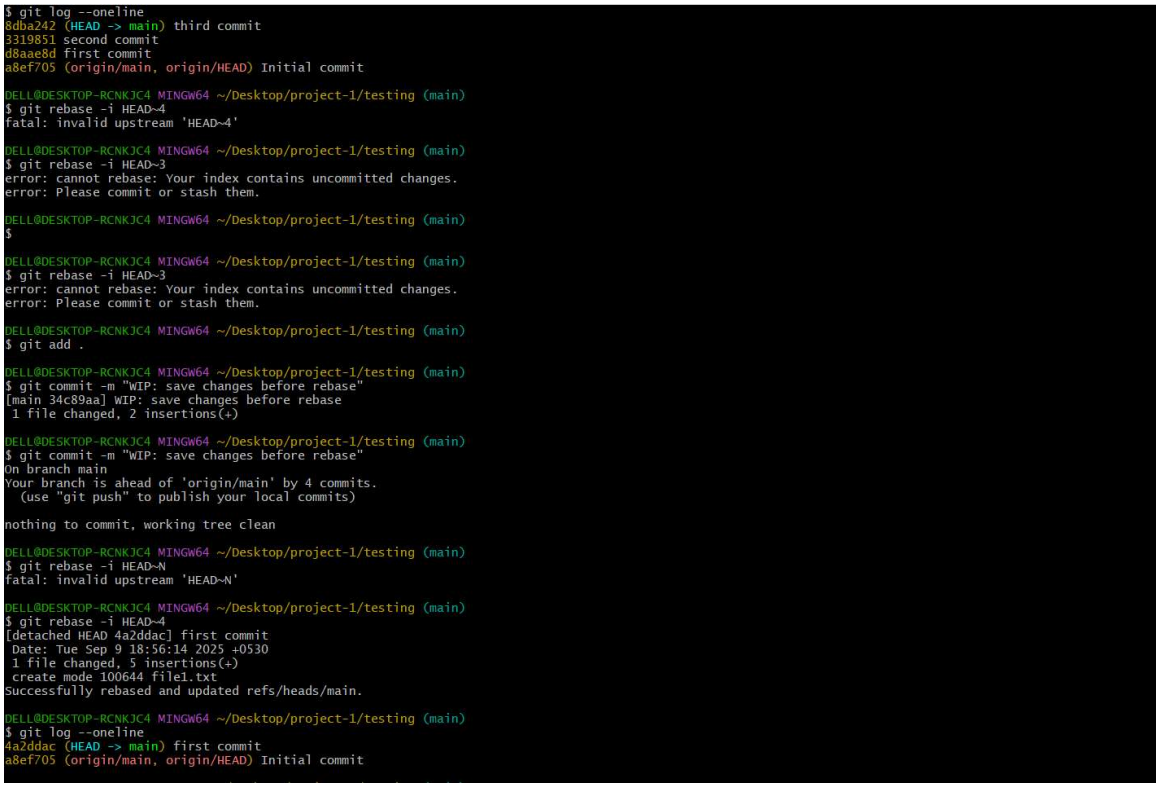
git commit --amend -m "add new commit message in it" (it will replace a new commit message)

(it will modify the latest commit only)

git push --force (to push all changes to central repo)

git log --oneline

22)Merge multiple commits into single commit..



git log --oneline

git rebase -i HEAD~4 (vi editor mode opens under change all picks with squash except the one where you want to add all commit)

git log --oneline