SAHYADRI
COLLEGE OF ENGINEERING & MANAGEMENT
An Autonomous Institution

GitHub: AI - Powered Development Platform (AEC)

Assignment - 2

Submitted by

NAME: SAMPATH KUMAR

USN: 4SF22CS175

Submitted to

Dr. Suhas A Bhyratae

Assistant professor

Department of Computer Science

Sahyadri College of Engineering & Management

Mangalore

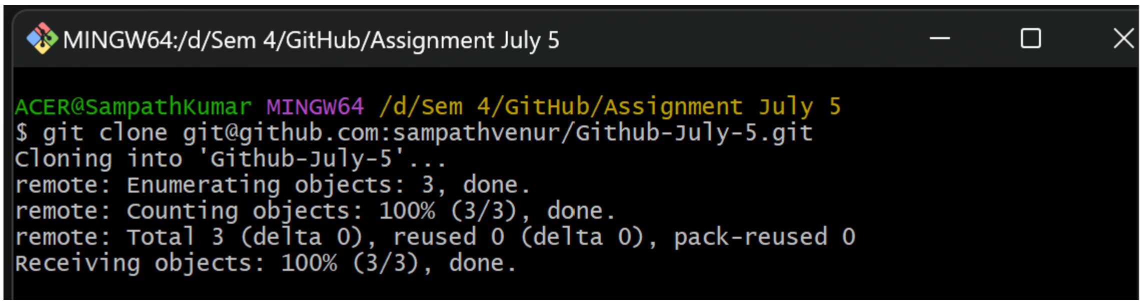
Activity-4 (Collaboration and Remote Repositories)

Q. Clone a remote Git repository to your local machine.

Step-1 To clone a remote Git repository to your local machine, you would typically use the git clone command followed by the URL of the repository you want to clone.

`> git clone <repository_url>`

Example :- `git clone https://github.com/example/repository.git`

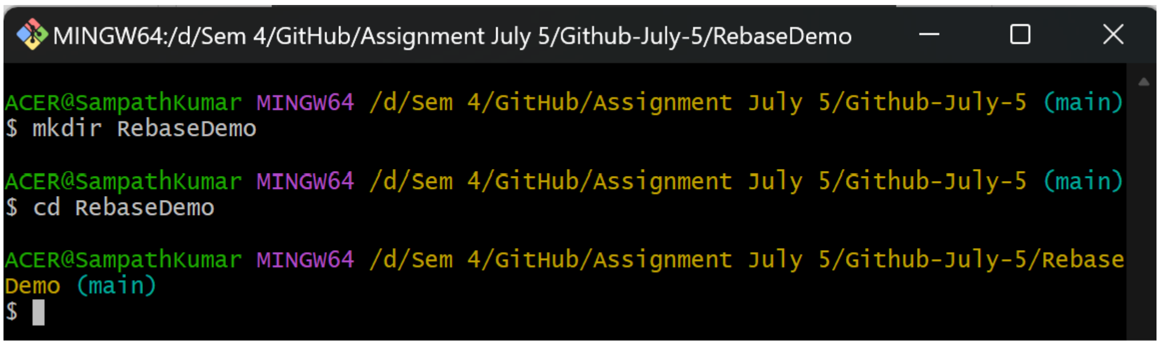


```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5
$ git clone git@github.com:sampathvenur/Github-July-5.git
Cloning into 'Github-July-5'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

Activity-5 (Collaboration and Remote Repositories)

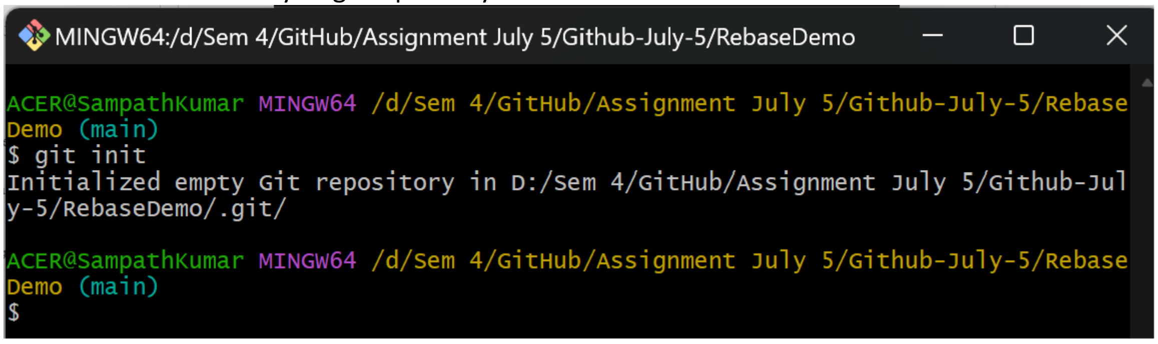
Q. Fetch the latest changes from a remote repository and rebase your local branch onto the updated remote branch.

1. Create directory "RebaseDemo" and go inside the folder


MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

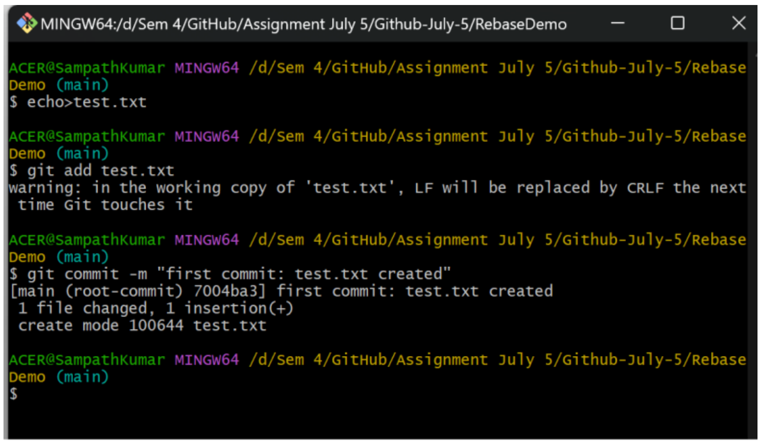
```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5 (main)
$ mkdir RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5 (main)
$ cd RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ 
```

2. Initialize the directory to git repository


MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

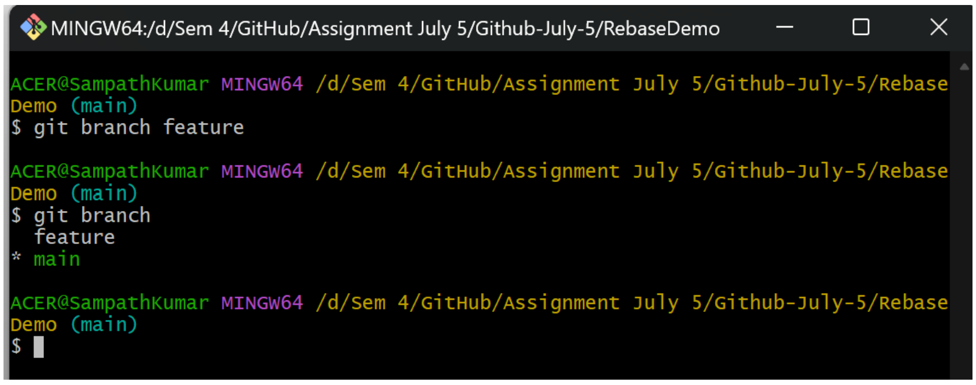
```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git init
Initialized empty Git repository in D:/Sem 4/GitHub/Assignment July 5/Github-Jul
y-5/RebaseDemo/.git/

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$
```

3. Create file test.txt and put to staging and then commit it.


MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

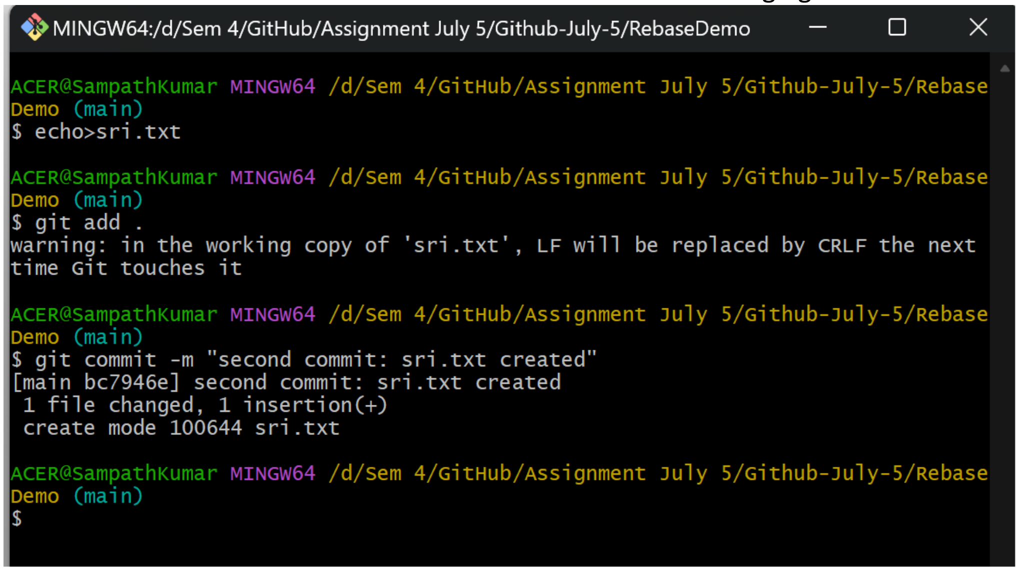
```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ echo>test.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git add test.txt
warning: in the working copy of 'test.txt', LF will be replaced by CRLF the next
 time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git commit -m "first commit: test.txt created"
[main (root-commit) 7004ba3] first commit: test.txt created
 1 file changed, 1 insertion(+)
 create mode 100644 test.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$
```

4. Create feature branch and check



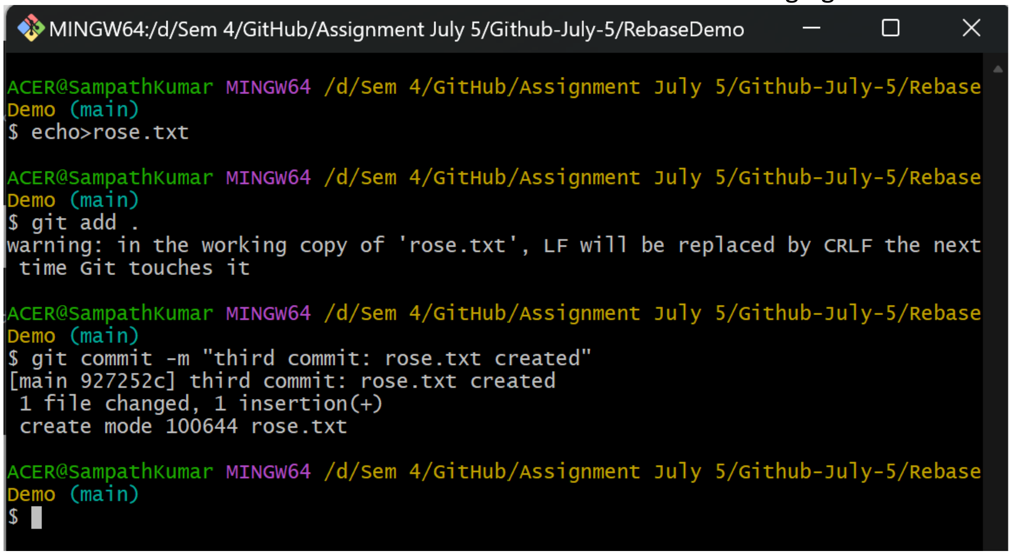
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git branch feature

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git branch
  feature
* main

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ 
```

5. Create one more file "sri.txt" in master branch and add to staging area and then commit it.



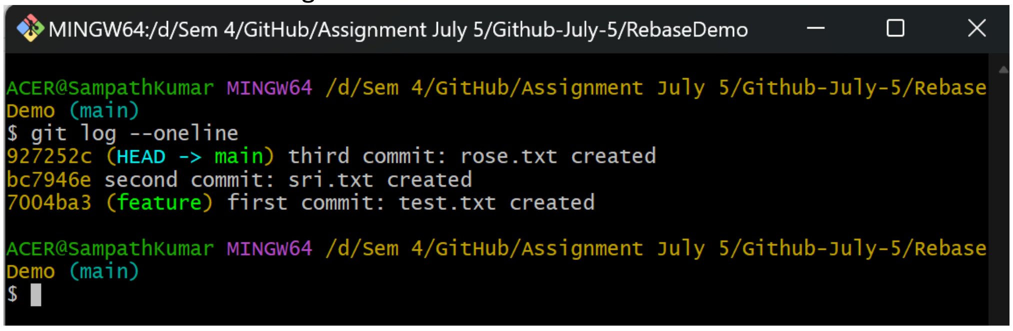
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ echo>sri.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git add .
warning: in the working copy of 'sri.txt', LF will be replaced by CRLF the next
time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git commit -m "second commit: sri.txt created"
[main bc7946e] second commit: sri.txt created
 1 file changed, 1 insertion(+)
 create mode 100644 sri.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$
```

6. Create one more file "rose.txt" in master branch and add to staging area and then commit it.



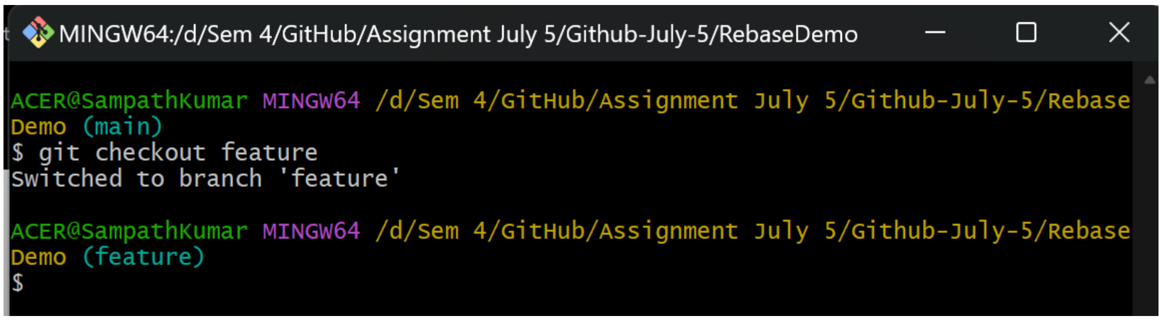
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ echo>rose.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git add .
warning: in the working copy of 'rose.txt', LF will be replaced by CRLF the next
 time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git commit -m "third commit: rose.txt created"
[main 927252c] third commit: rose.txt created
 1 file changed, 1 insertion(+)
 create mode 100644 rose.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ 
```

7. Check all the commits git



```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git log --oneline
927252c (HEAD -> main) third commit: rose.txt created
bc7946e second commit: sri.txt created
7004ba3 (feature) first commit: test.txt created

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ 
```

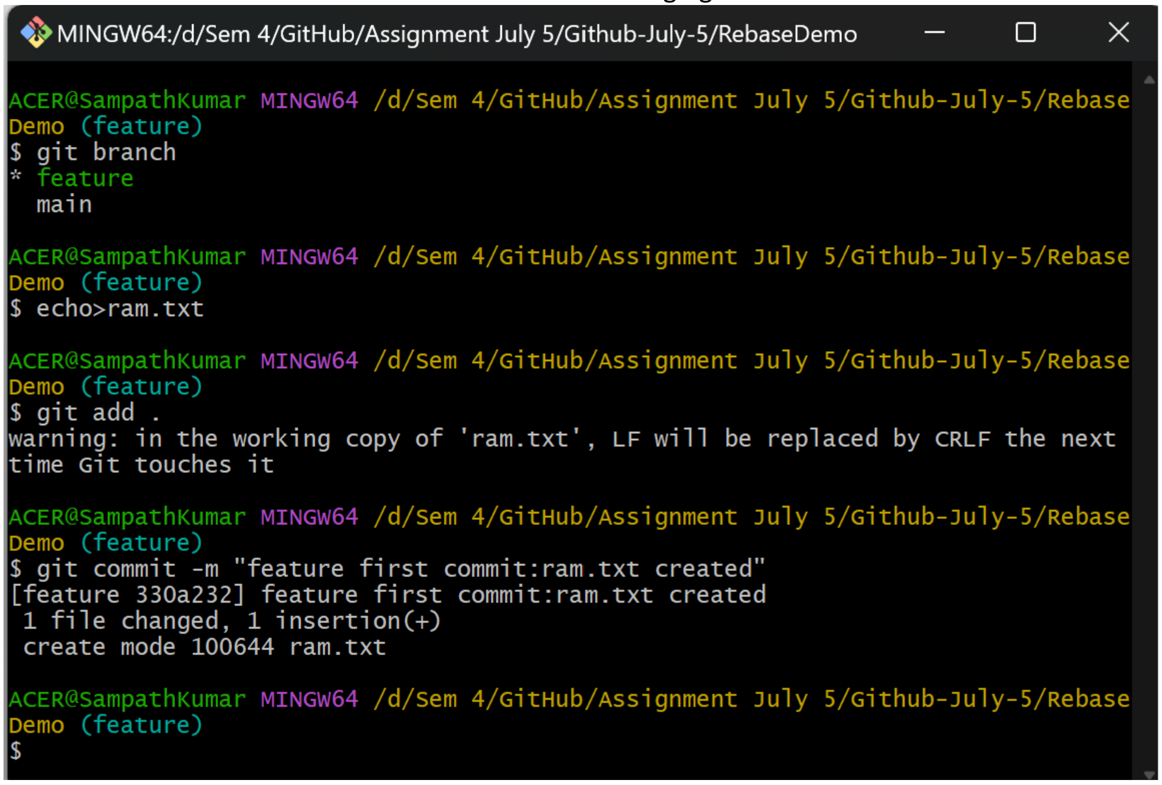
8. Switch to feature branch



MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git checkout feature
Switched to branch 'feature'

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$
```

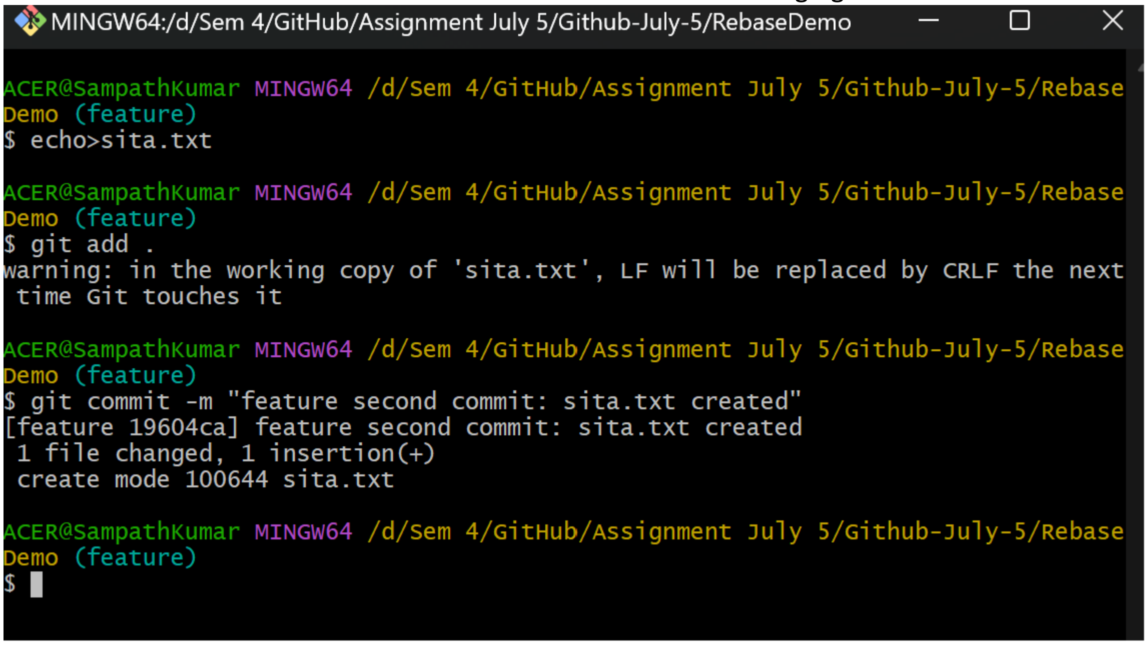
9. Create file in feature branch "ram.txt" and add to staging area and then commit it.



MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$ git branch
* feature
  main

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$ echo>ram.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$ git add .
warning: in the working copy of 'ram.txt', LF will be replaced by CRLF the next
time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$ git commit -m "feature first commit:ram.txt created"
[feature 330a232] feature first commit:ram.txt created
 1 file changed, 1 insertion(+)
 create mode 100644 ram.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$
```

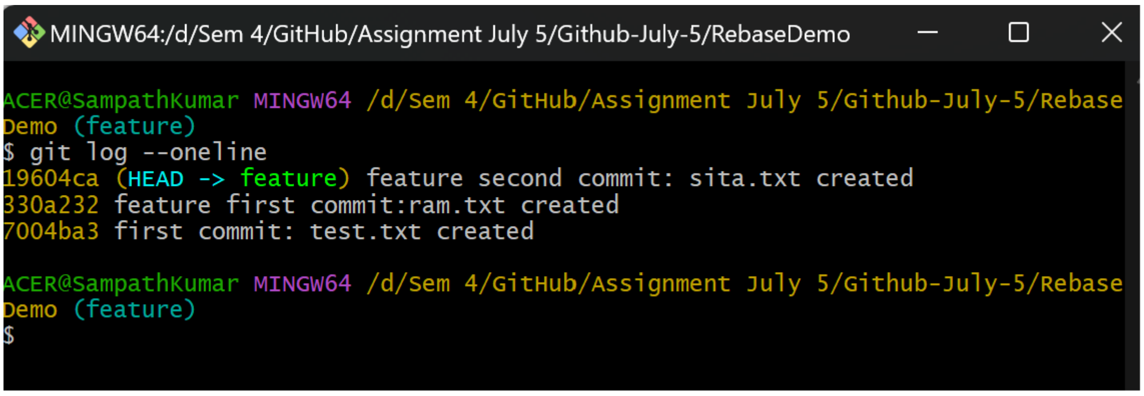
10. Create one more file "sita.txt" in feature branch and add to staging area and then commit it.



MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$ echo>sita.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$ git add .
warning: in the working copy of 'sita.txt', LF will be replaced by CRLF the next
 time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$ git commit -m "feature second commit: sita.txt created"
[feature 19604ca] feature second commit: sita.txt created
 1 file changed, 1 insertion(+)
 create mode 100644 sita.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$
```

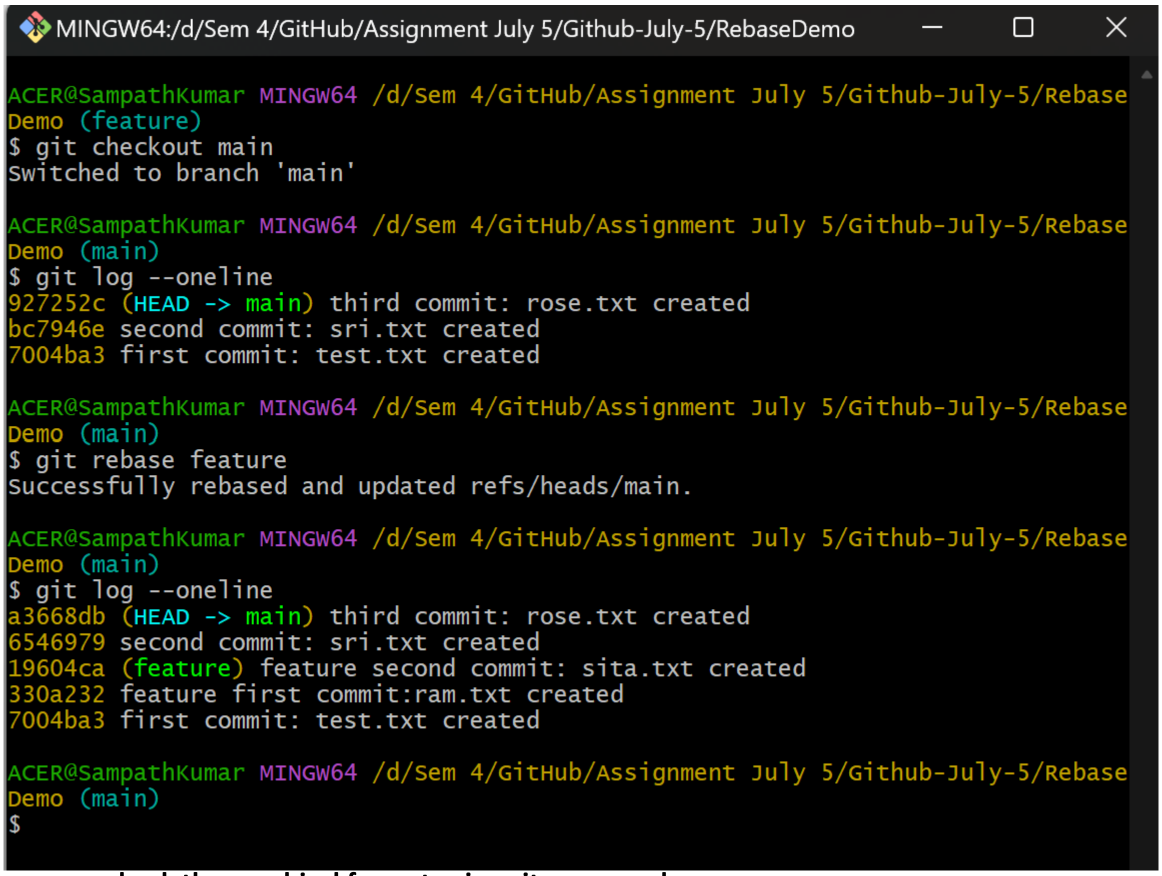
11. Check the commits in feature branch



```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$ git log --oneline
19604ca (HEAD -> feature) feature second commit: sita.txt created
330a232 feature first commit:ram.txt created
7004ba3 first commit: test.txt created

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$
```

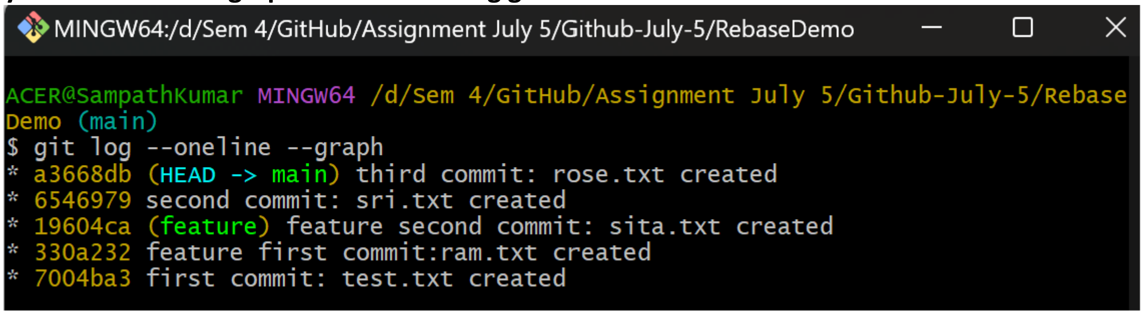
Rebase Process

12. Checkout to master branch and use rebase command to move all feature branch commits to master branch.



```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (feature)
$ git checkout main
Switched to branch 'main'

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git log --oneline
927252c (HEAD -> main) third commit: rose.txt created
bc7946e second commit: sri.txt created
7004ba3 first commit: test.txt created

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git rebase feature
Successfully rebased and updated refs/heads/main.

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git log --oneline
a3668db (HEAD -> main) third commit: rose.txt created
6546979 second commit: sri.txt created
19604ca (feature) feature second commit: sita.txt created
330a232 feature first commit:ram.txt created
7004ba3 first commit: test.txt created

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$
```

you can check the graphical format using git command



```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git log --oneline --graph
* a3668db (HEAD -> main) third commit: rose.txt created
* 6546979 second commit: sri.txt created
* 19604ca (feature) feature second commit: sita.txt created
* 330a232 feature first commit:ram.txt created
* 7004ba3 first commit: test.txt created
```

Activity-6 (Collaboration and Remote Repositories)

Q. Write the command to merge "feature-branch" into "master" while providing a custom commit message for the merge

To create a new branch named "feature-branch," switch to the "master" branch, and merge the "feature-branch" into "master" in Git.

Step-1: Make sure you are in the "master" branch by switching to it.

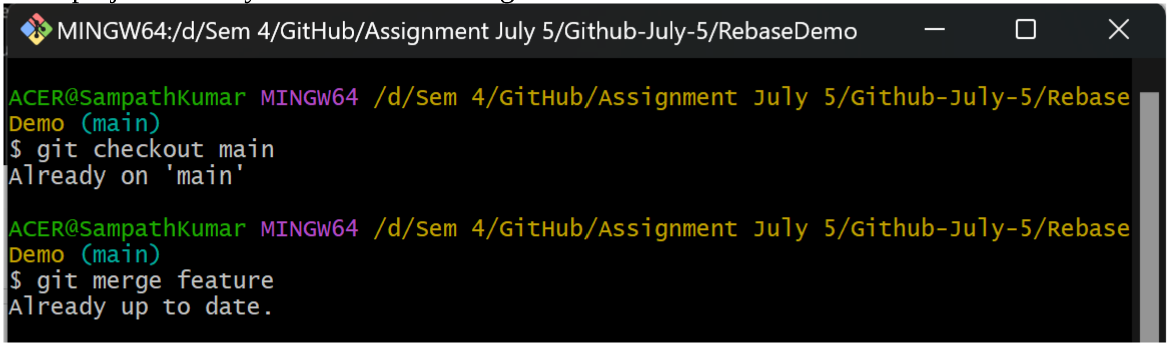
Step-2: Create a new branch named "feature-branch" and switch to it.

Step-3: Make your changes in the "feature-branch" by adding, modifying, or deleting files as needed.

Step-4. Stage and commit your changes in the "feature-branch".

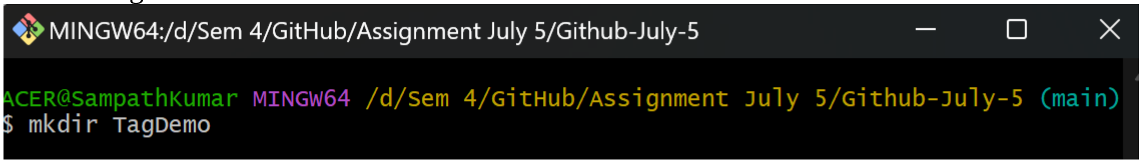
Step-5. Switch back to the "master" branch.

Step-6. Merge the "feature-branch" into the "master" branch.
This command will incorporate the changes from the "feature-branch" into the "master" branch.
Now, your changes from the "feature-branch" have been merged into the "master" branch.
Your project's history will reflect the changes made in both branches.

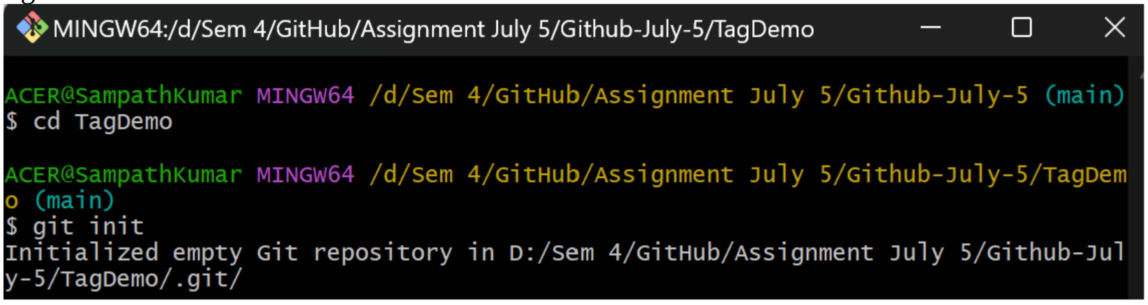

```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/RebaseDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git checkout main
Already on 'main'

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/Rebase
Demo (main)
$ git merge feature
Already up to date.
```

Activity-7 (Git Tags and Releases

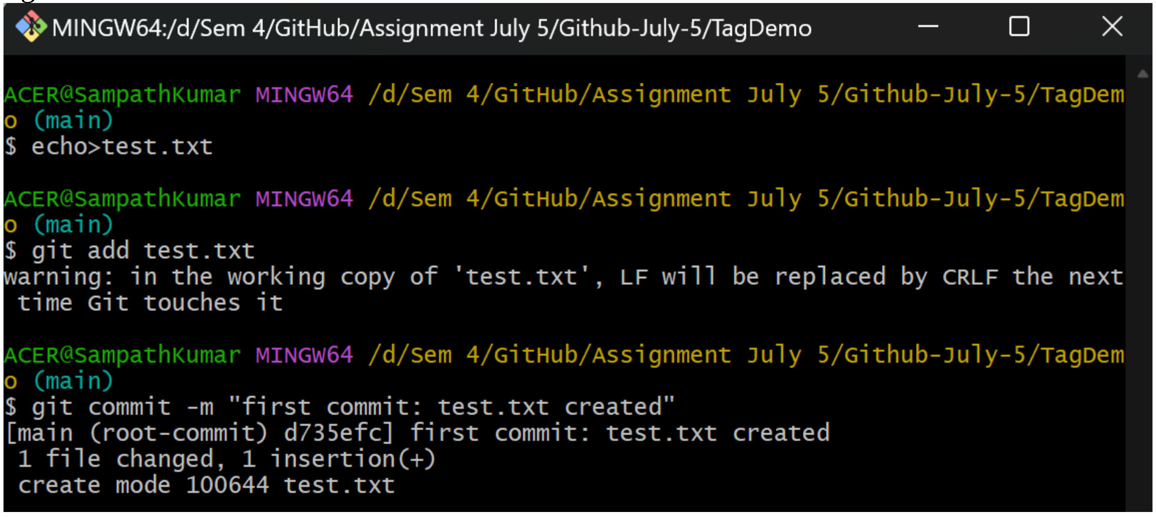
Steps-1: create working directory/folder "Tagdemo" using command
>mkdir TagDemo


```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5 (main)
$ mkdir TagDemo
```

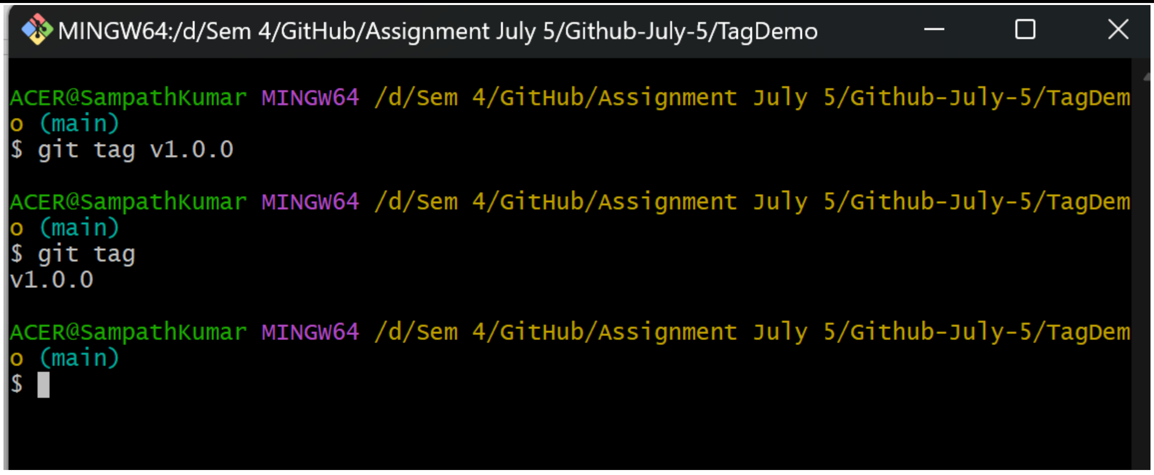
Step-2: go inside the working directory and initialize this directory as git repository using command
>cd TagDemo
>git init


```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5 (main)
$ cd TagDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git init
Initialized empty Git repository in D:/Sem 4/GitHub/Assignment July 5/Github-Jul
y-5/TagDemo/.git/
```

Step-2: create file "test.txt" and put it to staging and then commit it using following command
>echo>test.txt
>git add test.txt
>git commit –m "first commit: test.txt created"

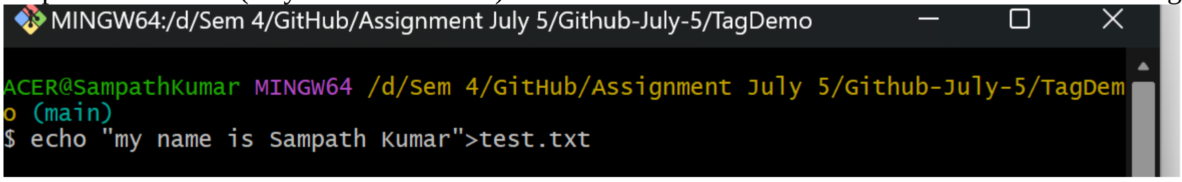

```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ echo>test.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git add test.txt
warning: in the working copy of 'test.txt', LF will be replaced by CRLF the next
 time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git commit -m "first commit: test.txt created"
[main (root-commit) d735efc] first commit: test.txt created
 1 file changed, 1 insertion(+)
 create mode 100644 test.txt
```

Step-3: now create the tag and check whether the tags are created using command
>git tag v1.0.0

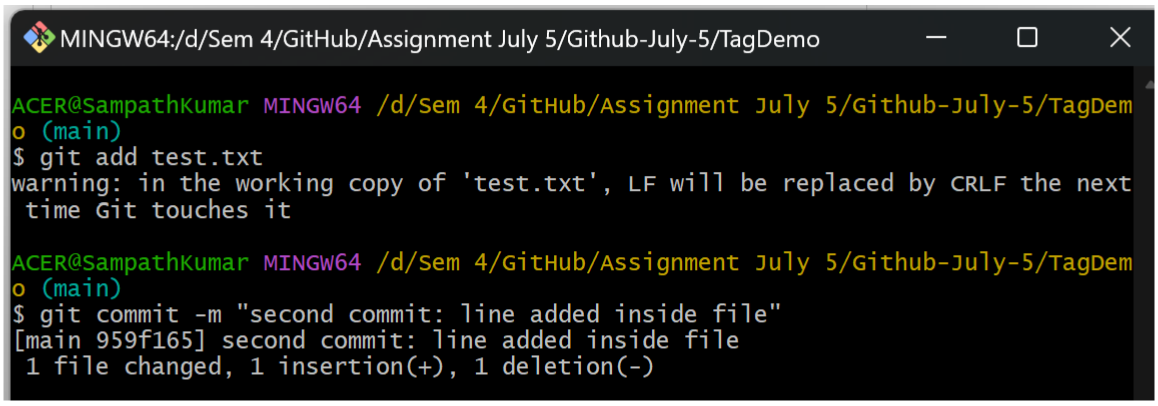

MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo

```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git tag v1.0.0

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git tag
v1.0.0

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$
```

Step-4 add one line ("my name is srinath") inside test.txt file and make 2nd commit and then add tag (v2.0.0)

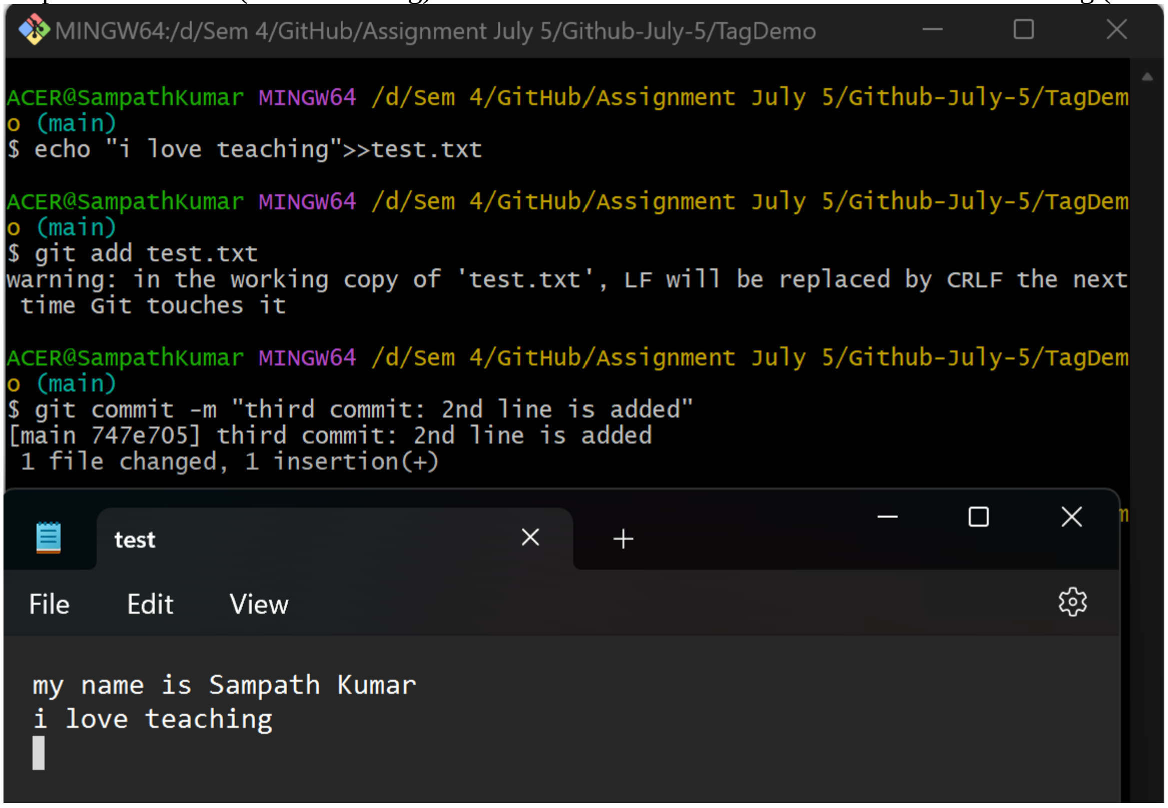

MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo

```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ echo "my name is Sampath Kumar">test.txt
```


MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo

```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git add test.txt
warning: in the working copy of 'test.txt', LF will be replaced by CRLF the next
 time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git commit -m "second commit: line added inside file"
[main 959f165] second commit: line added inside file
 1 file changed, 1 insertion(+), 1 deletion(-)
```

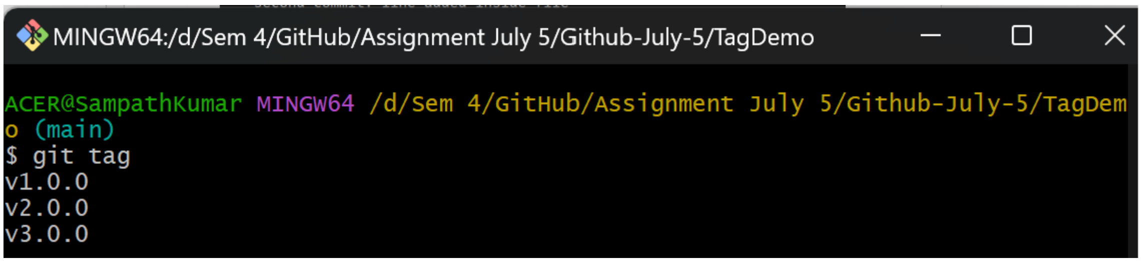
Step-5: add 2nd line (I love teaching) in the test.txt file and make 3rd commit and then add tag (v3.0.0)


MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo
test
File
Edit
View

```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ echo "i love teaching">>test.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git add test.txt
warning: in the working copy of 'test.txt', LF will be replaced by CRLF the next
 time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git commit -m "third commit: 2nd line is added"
[main 747e705] third commit: 2nd line is added
 1 file changed, 1 insertion(+)
```

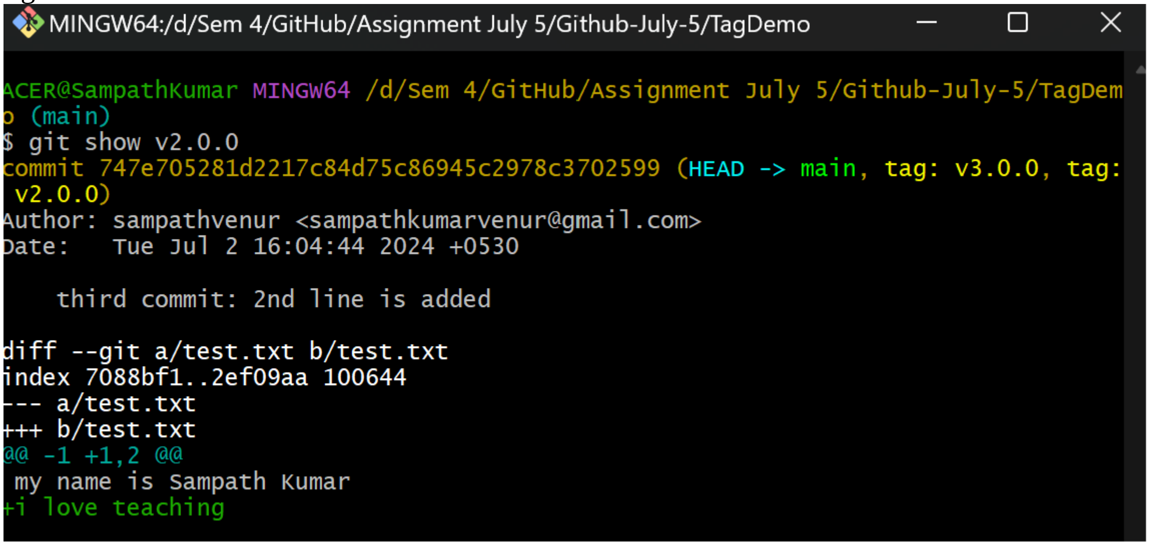
```
my name is Sampath Kumar
i love teaching
```

Step-6: check list of all tag avaiable using command
>git tag


MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo

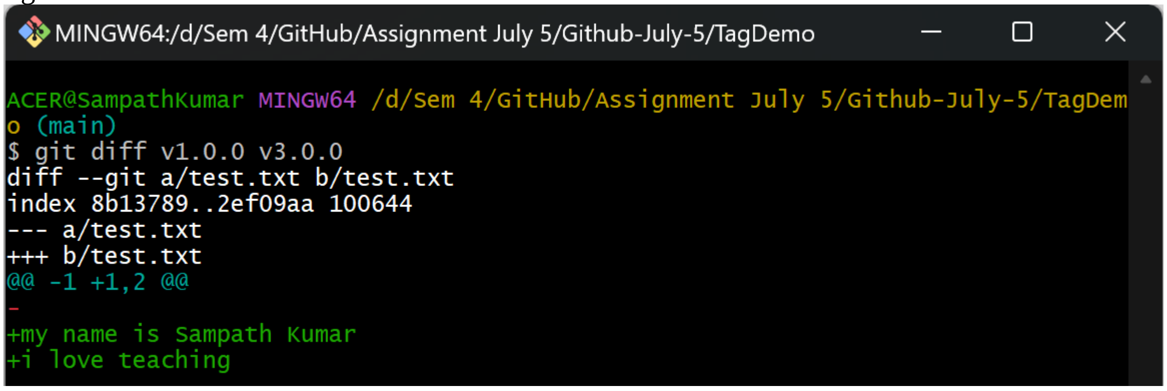
```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git tag
v1.0.0
v2.0.0
v3.0.0
```

Step-7: to see any particular tag and its detail use command
>git show "tag name"
example :
>git show v2.0.0


MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo

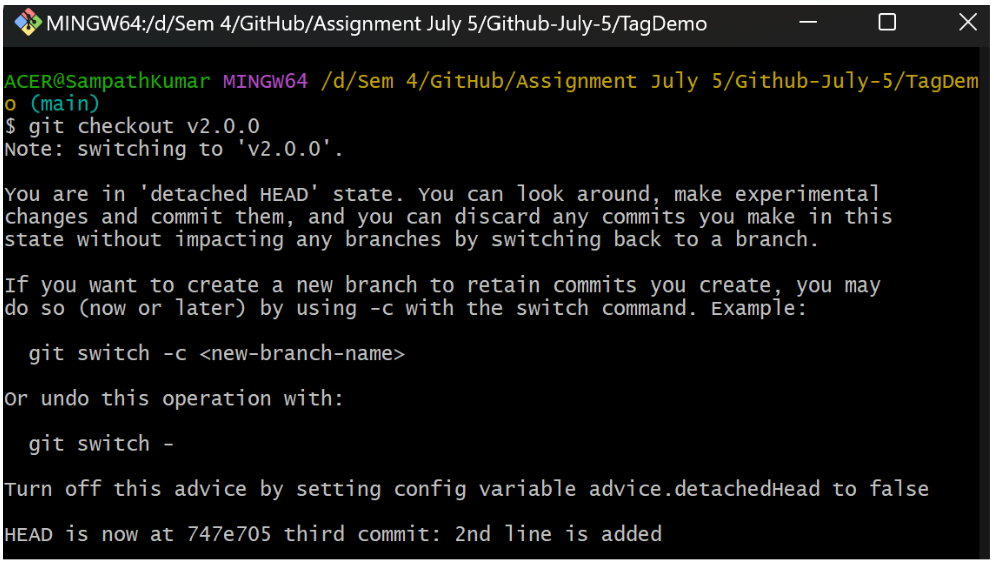
```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git show v2.0.0
commit 747e705281d2217c84d75c86945c2978c3702599 (HEAD -> main, tag: v3.0.0, tag:
 v2.0.0)
Author: sampathvenur <sampathkumarvenur@gmail.com>
Date:   Tue Jul 2 16:04:44 2024 +0530

    third commit: 2nd line is added

diff --git a/test.txt b/test.txt
index 7088bf1..2ef09aa 100644
--- a/test.txt
+++ b/test.txt
@@ -1 +1,2 @@
 my name is Sampath Kumar
+i love teaching
```

Step-8: To see the difference between two versions (i.e v1.0.0 and v3.0.0) use below command
>git diff v1.0.0 v3.0.0


MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo

```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git diff v1.0.0 v3.0.0
diff --git a/test.txt b/test.txt
index 8b13789..2ef09aa 100644
--- a/test.txt
+++ b/test.txt
@@ -1 +1,2 @@
-
+my name is Sampath Kumar
+i love teaching
```

You can we are now checked out to v2.0.0 version.


MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDemo

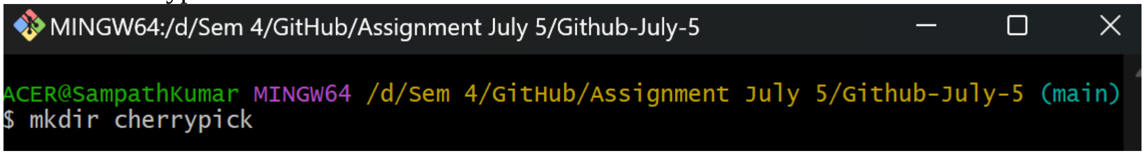
```
ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/TagDem
o (main)
$ git checkout v2.0.0
Note: switching to 'v2.0.0'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 747e705 third commit: 2nd line is added
```

Activity-8 (Advanced Git Operations)

Q. Write the command to cherry-pick a range of commits from "source-branch" to the current branch. ("pick a commit from one branch and place in another branch" we use cherry-pick)
Cherry pick is used if you want to apply particular commit from one branch into another branch
Note: if you don’t want to merge whole branch and you want some of the commits then cherry pick is helpful.

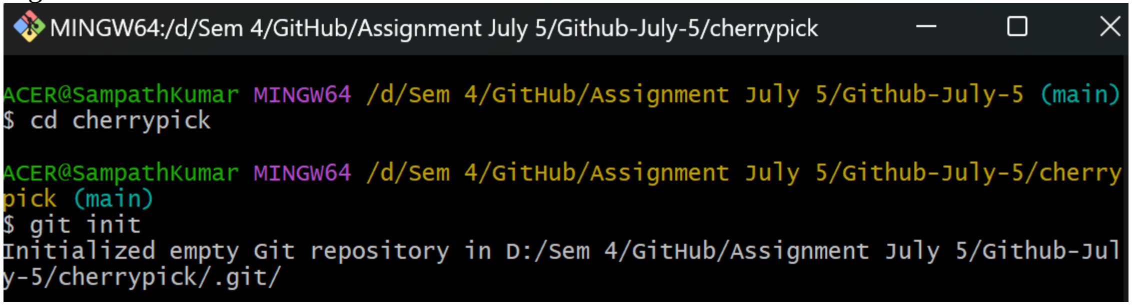
Cherry pick is same as rebase
Note : it is advised not to use cherry pick always, because it will cause duplicate commits

Steps-1: create new folder/project to demonstrate cherry pick
> mkdir cherrypick



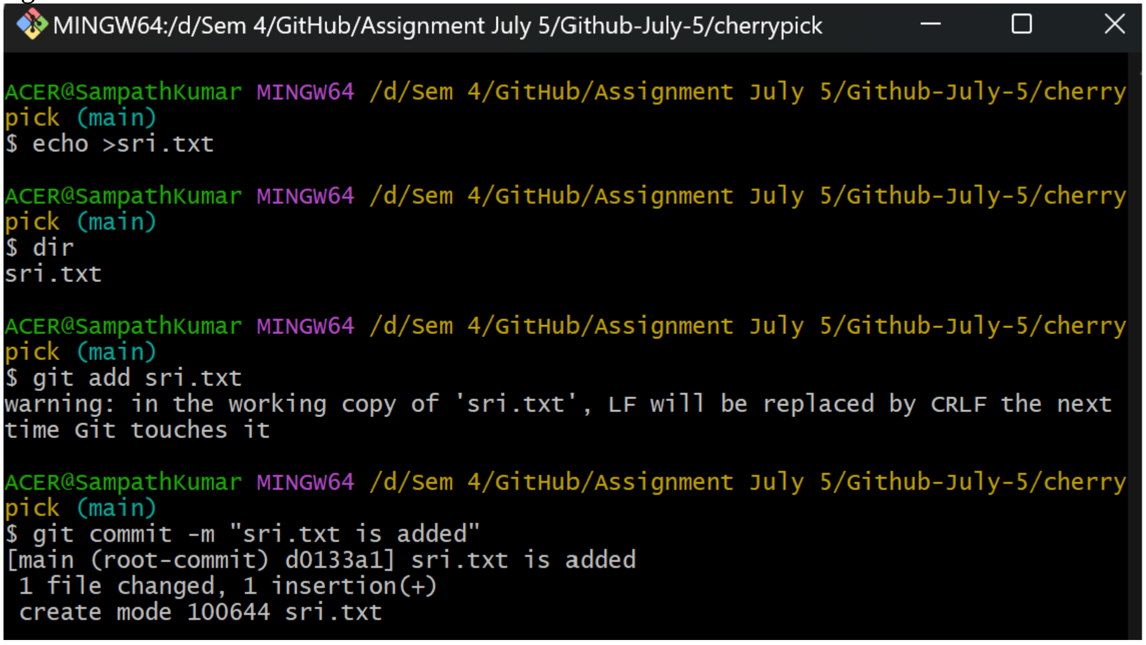
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5 (main)
$ mkdir cherrypick
```

Step-2: go inside the folder cherrypick and initialize the git repository using the command
> cd cherrypick
> git init



```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5 (main)
$ cd cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git init
Initialized empty Git repository in D:/Sem 4/GitHub/Assignment July 5/Github-Jul
y-5/cherrypick/.git/
```

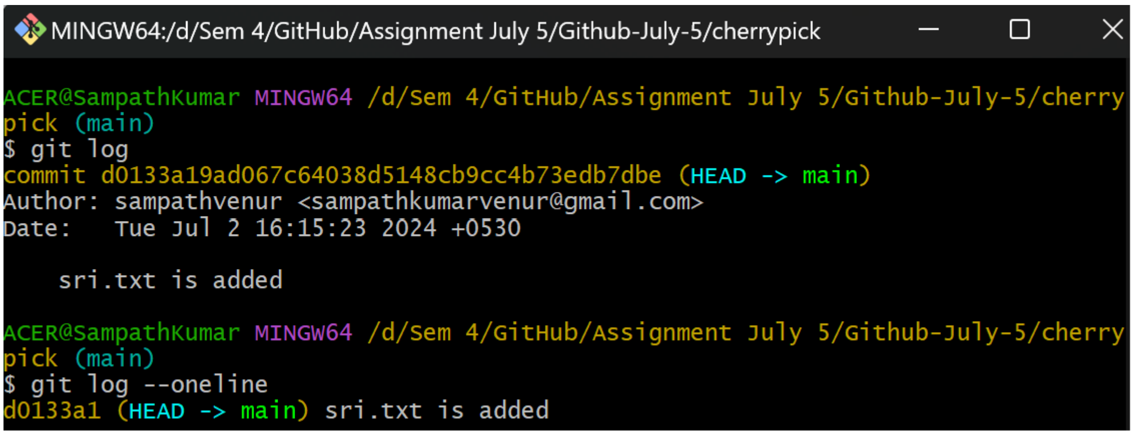
Step-3 : create a file "sri.txt" and add it to staging area then commit it using command
>echo > sri.txt
>git add sri.txt
>gir commit –m "sri.txt is added"



```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ echo >sri.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ dir
sri.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git add sri.txt
warning: in the working copy of 'sri.txt', LF will be replaced by CRLF the next
time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git commit -m "sri.txt is added"
[main (root-commit) d0133a1] sri.txt is added
 1 file changed, 1 insertion(+)
 create mode 100644 sri.txt
```

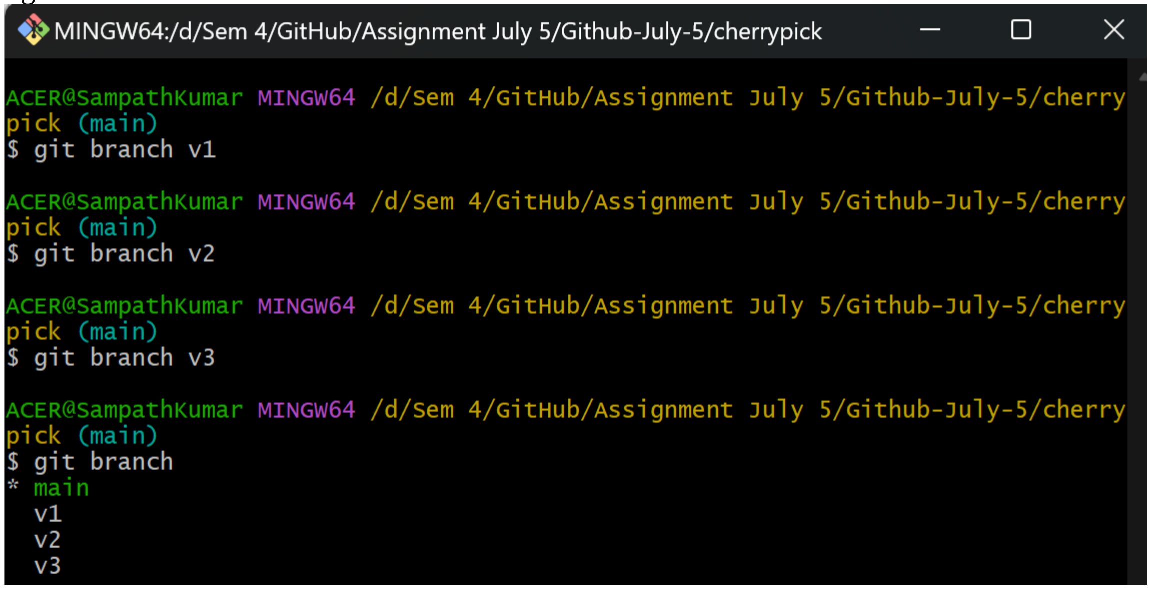
Step-4 : check the git log for the commit msg using command
> git log

Or you can use shorthand command
>git log –oneline



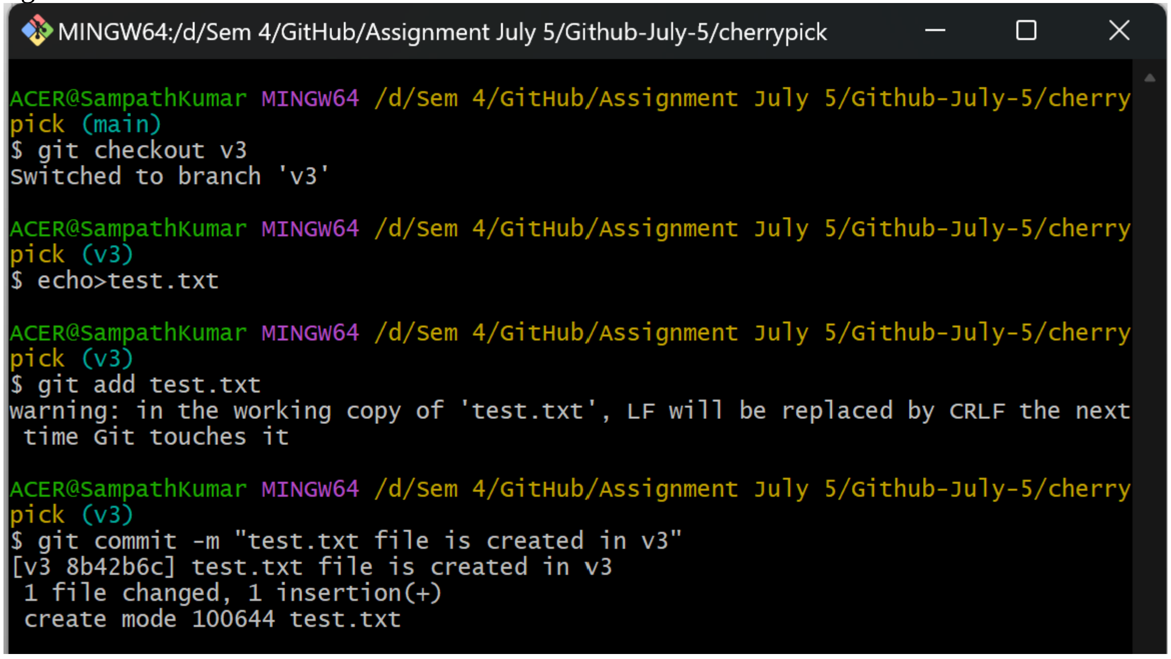
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git log
commit d0133a19ad067c64038d5148cb9cc4b73edb7dbe (HEAD -> main)
Author: sampathvenur <sampathkumarvenur@gmail.com>
Date:   Tue Jul 2 16:15:23 2024 +0530

    sri.txt is added

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git log --oneline
d0133a1 (HEAD -> main) sri.txt is added
```

Step-5 create three versions of the project that is three branches called v1, v2 and v3 and check the branches create or not using command
>git branch v1
>git branch v2
> git branch v3
Check
>git branch



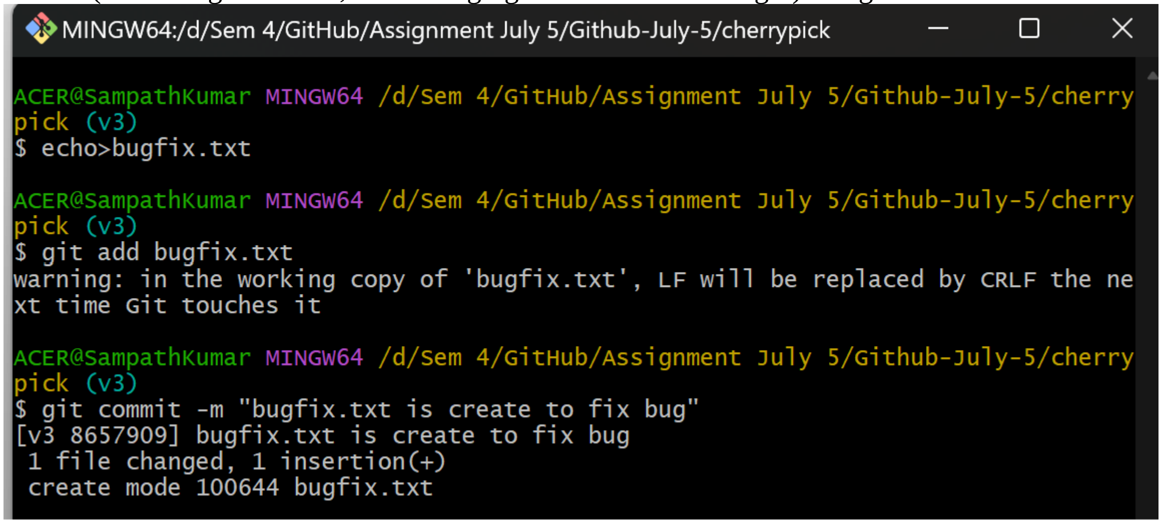
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git branch v1

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git branch v2

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git branch v3

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git branch
* main
  v1
  v2
  v3
```

Step-6 go to v3 branch and create a file "test.txt" and add it to staging and then commit it using command
>git checkout v3



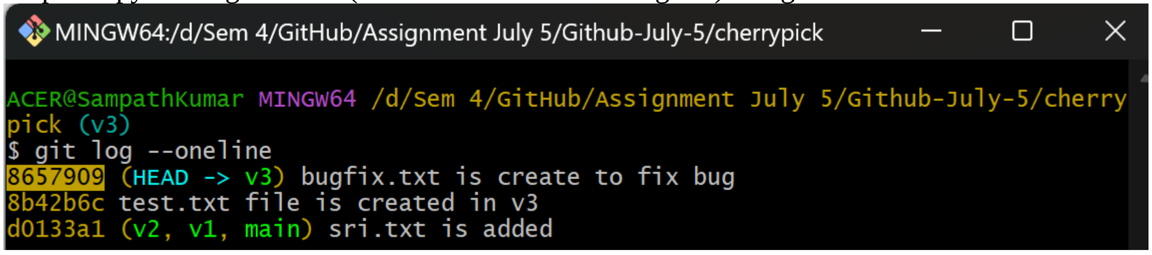
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (main)
$ git checkout v3
Switched to branch 'v3'

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v3)
$ echo>test.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v3)
$ git add test.txt
warning: in the working copy of 'test.txt', LF will be replaced by CRLF the next
 time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v3)
$ git commit -m "test.txt file is created in v3"
[v3 8b42b6c] test.txt file is created in v3
 1 file changed, 1 insertion(+)
 create mode 100644 test.txt
```

Note: if there is a bug in the V3 then we need to fix the bug, so we simulate this bugfix by adding bugfix.txt file in this branch (create bugfix.txt file, add to staging and then committing it) using command



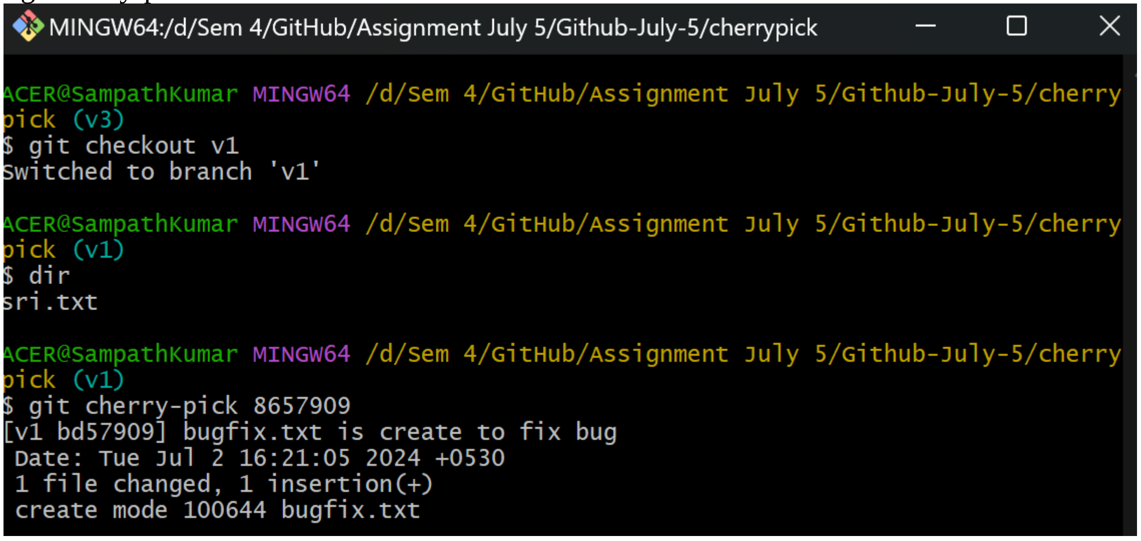
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v3)
$ echo>bugfix.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v3)
$ git add bugfix.txt
warning: in the working copy of 'bugfix.txt', LF will be replaced by CRLF the ne
xt time Git touches it

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v3)
$ git commit -m "bugfix.txt is create to fix bug"
[v3 8657909] bugfix.txt is create to fix bug
 1 file changed, 1 insertion(+)
 create mode 100644 bugfix.txt
```

From below figure you can see bugfix is made only in v3 branch and hence and now we need appy this commit in v1 and v2 as well but don't want to merge this v2 bugfix in v1 and v2 because we are still working on it and not completed the fix fully. Hence in this scenario we go for cherry pick to copy the current entry into v2 and v1.
Step-1 copy the bug fix hash (i.e c057a7c see below figure) using Ctrl C command



```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v3)
$ git log --oneline
8657909 (HEAD -> v3) bugfix.txt is create to fix bug
8b42b6c test.txt file is created in v3
d0133a1 (v2, v1, main) sri.txt is added
```

Step-2: checkout from V3 branch to V1 and then use cherry pick command to apply the commit as show below
> git checkout v1
>dir
Note : you see only one file (i.e sri.txt is available before cherry pick command is applied)
After applying cherry-pick command you see both files (i.e sri.txt and bugfix.txt)
>git cherry-pick

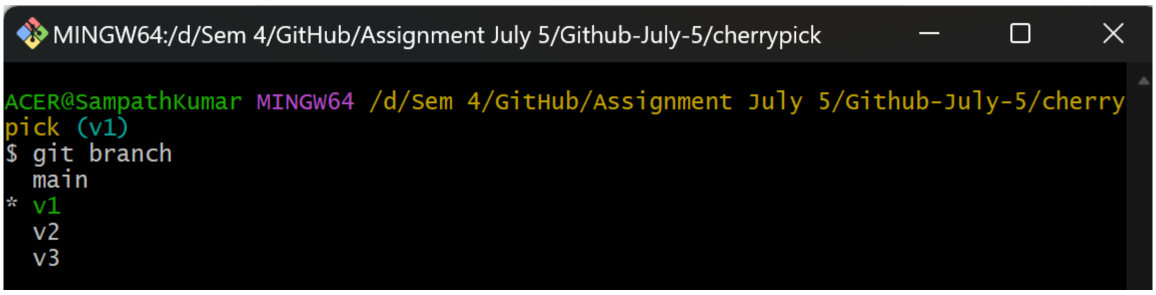


```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v3)
$ git checkout v1
Switched to branch 'v1'

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v1)
$ dir
sri.txt

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v1)
$ git cherry-pick 8657909
[v1 bd57909] bugfix.txt is create to fix bug
 Date: Tue Jul 2 16:21:05 2024 +0530
 1 file changed, 1 insertion(+)
 create mode 100644 bugfix.txt
```

Activity-9 (Analysing and Changing Git History)

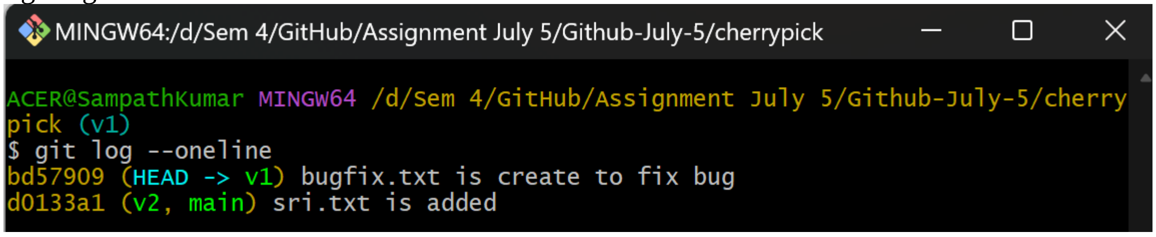
Q. Given a commit ID, how would you use Git to view the details of that specific commit, including the author, date, and commit message?

Step -1 : use git branch command to see which branch you are in
> git branch



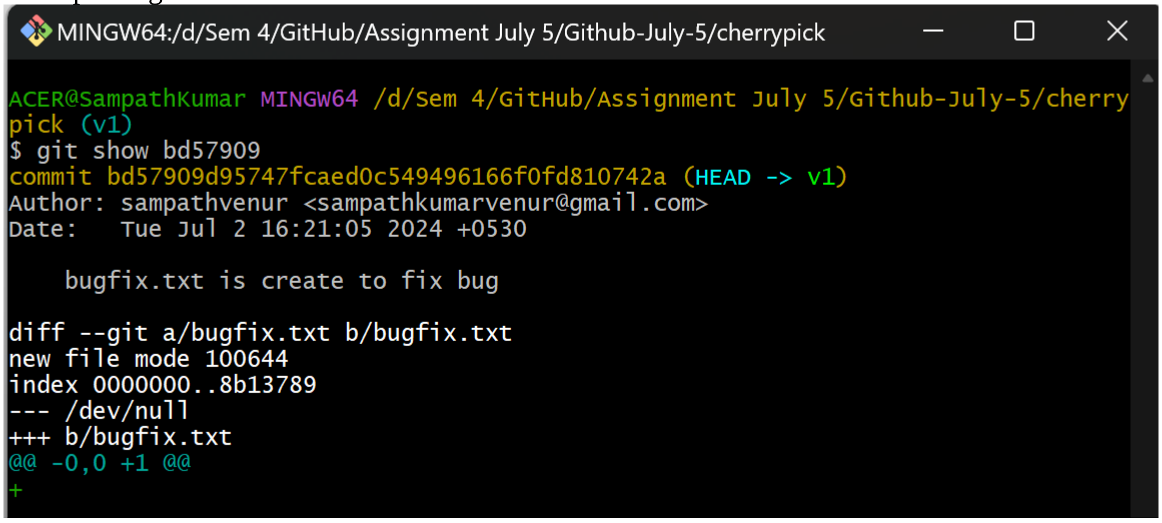
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v1)
$ git branch
  main
* v1
  v2
  v3
```

Step-2 check the log to get hash and commit message using following command
>git log –oneline



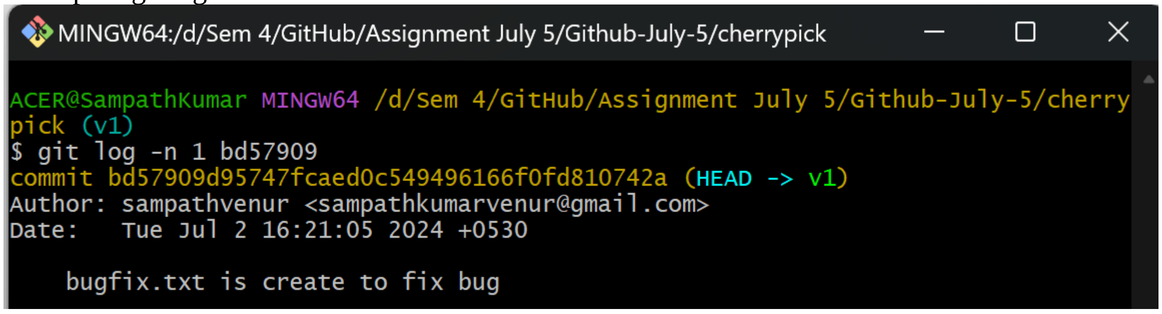
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v1)
$ git log --oneline
bd57909 (HEAD -> v1) bugfix.txt is create to fix bug
d0133a1 (v2, main) sri.txt is added
```

Step-3: To view the details of a specific commit, including the author, date, and commit message, you can use the git show command
>git show <commit_id>
Example : >git show 96117fd



```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v1)
$ git show bd57909
commit bd57909d95747fcaed0c549496166f0fd810742a (HEAD -> v1)
Author: sampathvenur <sampathkumarvenur@gmail.com>
Date:   Tue Jul 2 16:21:05 2024 +0530

    bugfix.txt is create to fix bug

diff --git a/bugfix.txt b/bugfix.txt
new file mode 100644
index 0000000..8b13789
--- /dev/null
+++ b/bugfix.txt
@@ -0,0 +1 @@
+
```

Note : if you want to see only author, date and commit message use the following command
$ git log -n 1 <commit-ID>
Example : git log –n



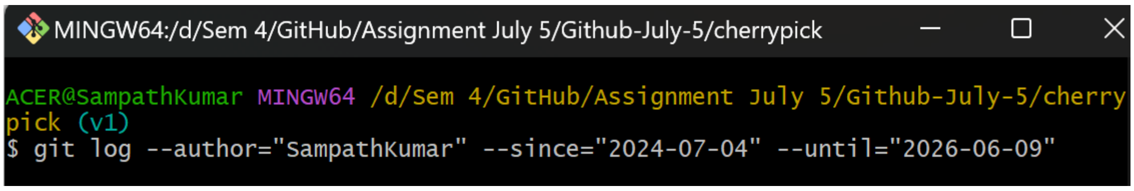
```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v1)
$ git log -n 1 bd57909
commit bd57909d95747fcaed0c549496166f0fd810742a (HEAD -> v1)
Author: sampathvenur <sampathkumarvenur@gmail.com>
Date:   Tue Jul 2 16:21:05 2024 +0530

    bugfix.txt is create to fix bug
```

Activity-10 (Analysing and Changing Git History)

Q. Write the command to list all commits made by the author "JohnDoe" between "2023-01-01" and "2023-12-31."

Step-1 : To list all commits made by the author "JohnDoe" between "2023-01-01" and "2023-12-31," you can use the git log command with the --author and –since and –until option

```
> git log --author="JohnDoe" --since="2023-01-01" --until="2023-12-31"
```

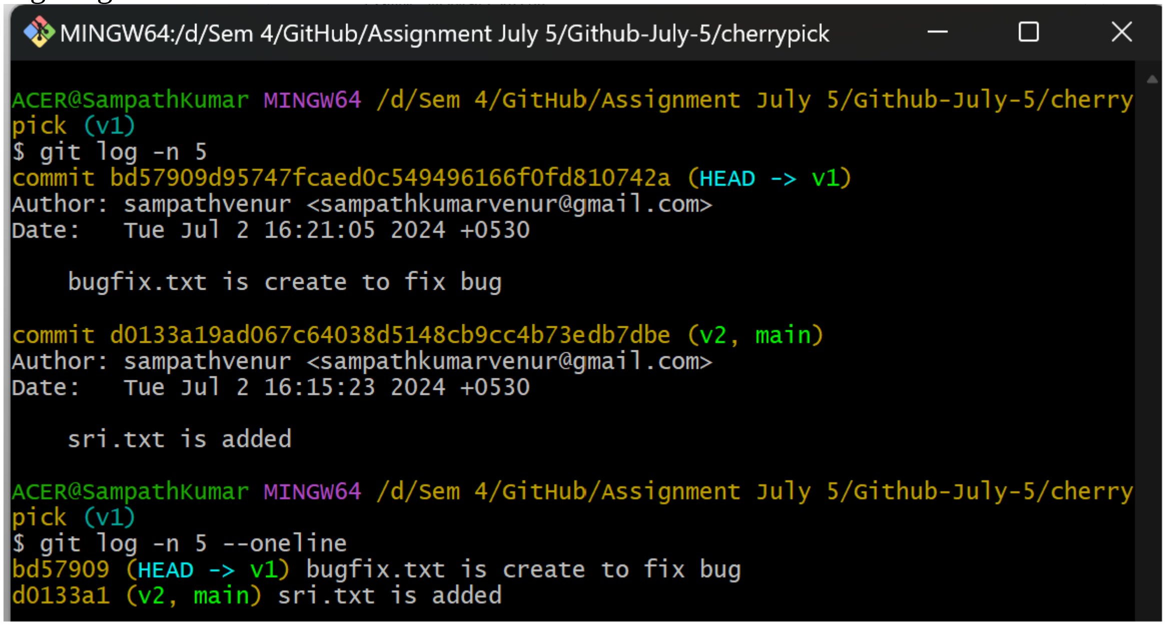


```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v1)
$ git log --author="SampathKumar" --since="2024-07-04" --until="2026-06-09"
```

Activity-11 (Analysing and Changing Git History)

Q. Write the command to display the last five commits in the repository's history.
To display the last five commits in the repository's history, you can use the git log command with the -n option to limit the number of commits displayed. Here's the command:
> git log -n 5

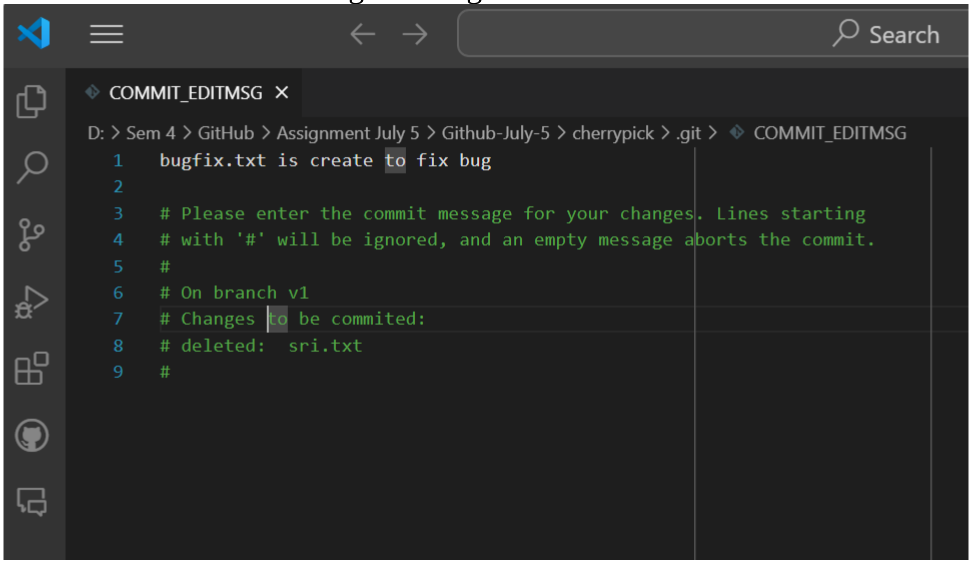
Note:- If you want a more condensed view, you can use the --oneline option to display each commit on a single line:
> git log -n 5 –oneline



```
MINGW64:/d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherrypick

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v1)
$ git log -n 5
commit bd57909d95747fcaed0c549496166f0fd810742a (HEAD -> v1)
Author: sampathvenur <sampathkumarvenur@gmail.com>
Date:   Tue Jul 2 16:21:05 2024 +0530

    bugfix.txt is create to fix bug

commit d0133a19ad067c64038d5148cb9cc4b73edb7dbe (v2, main)
Author: sampathvenur <sampathkumarvenur@gmail.com>
Date:   Tue Jul 2 16:15:23 2024 +0530

    sri.txt is added

ACER@SampathKumar MINGW64 /d/Sem 4/GitHub/Assignment July 5/Github-July-5/cherry
pick (v1)
$ git log -n 5 --oneline
bd57909 (HEAD -> v1) bugfix.txt is create to fix bug
d0133a1 (v2, main) sri.txt is added
```

Activity-12 (Analysing and Changing Git History)

Q. Write the command to undo the changes introduced by the commit with the ID "abc123".
To undo the changes introduced by a specific commit with the ID "abc123", you can use the "git revert" command. This command creates a new commit that undoes the changes made by the specified commit.
>git revert abc123
Sometimes It shows message waiting editor to close


Search
COMMIT_EDITMSG
D: > Sem 4 > GitHub > Assignment July 5 > Github-July-5 > cherrypick > .git > COMMIT_EDITMSG

```
bugfix.txt is create to fix bug

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch v1
# Changes to be commited:
# deleted:   sri.txt
#
```