MLOps Assignment 2 Solution

Name: Abhishek Rana **Course:** MTech Data Science **Roll Number:** 142502002

1. Repository Creation: Create a repository named git-assignment-[your-roll-no] within the assignment folder as discussed in the class with the following specific initial structure and required Initial Commits (must be created in this exact order):

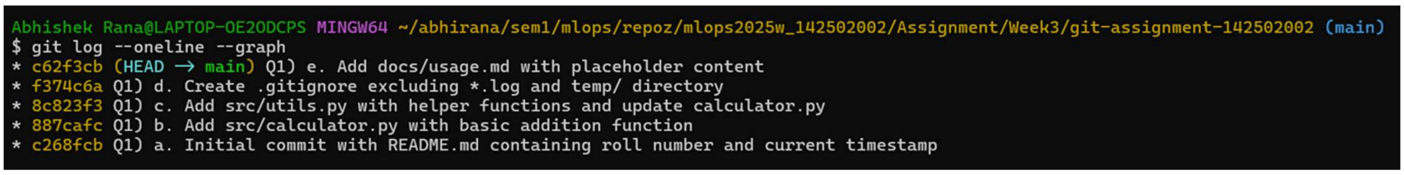
a. Initial commit with README.md containing your roll number and current timestamp

b. Add src/calculator.py with basic addition function

c. Add src/utils.py with helper functions

d. Create .gitignore excluding *.log and temp/ directory

e. Add docs/usage.md with placeholder content

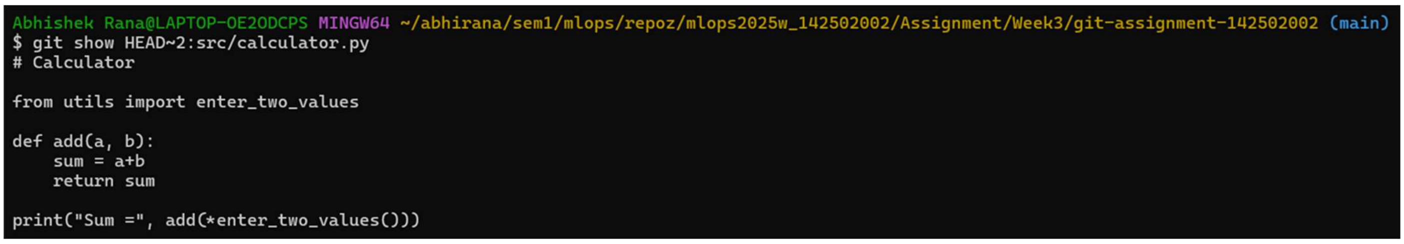


```
Abhishek Rana@LAPTOP-OE20DCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git log --oneline --graph
* c62f3cb (HEAD -> main) Q1) e. Add docs/usage.md with placeholder content
* f374c6a Q1) d. Create .gitignore excluding *.log and temp/ directory
* 8c823f3 Q1) c. Add src/utils.py with helper functions and update calculator.py
* 887cafc Q1) b. Add src/calculator.py with basic addition function
* c268fcb Q1) a. Initial commit with README.md containing roll number and current timestamp
```

2. Revision selection: Write the exact Git commands for the following

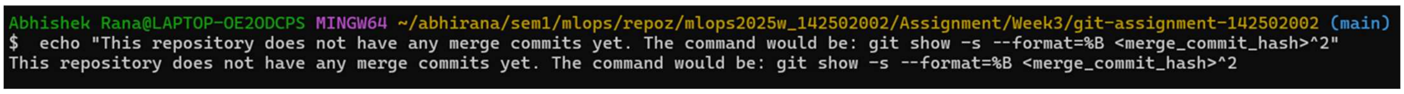
Attach the screenshot of the output of all the commands.

a. Show the content of calculator.py from 2 commits before HEAD



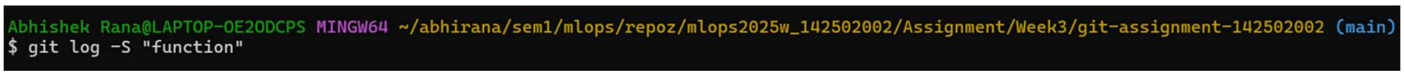
```
Abhishek Rana@LAPTOP-OE20DCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git show HEAD~2:src/calculator.py
# Calculator

from utils import enter_two_values

def add(a, b):
    sum = a+b
    return sum

print("Sum =", add(*enter_two_values()))
```

b. Display the commit message of the second parent of a merge commit (if exists)



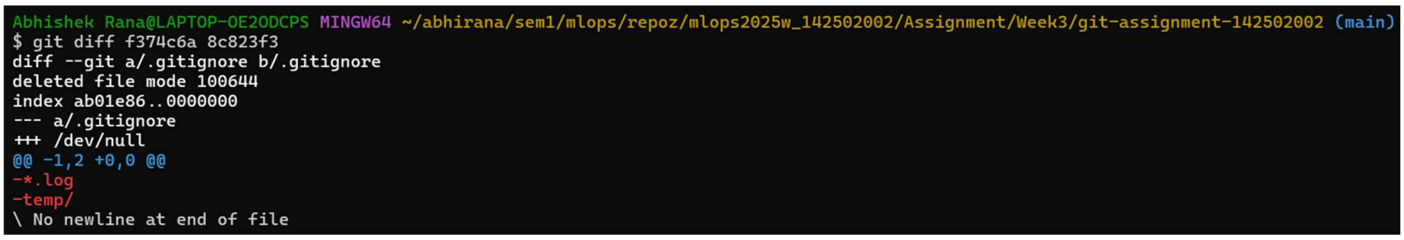
```
Abhishek Rana@LAPTOP-OE20DCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$  echo "This repository does not have any merge commits yet. The command would be: git show -s --format=%B <merge_commit_hash>^2"
This repository does not have any merge commits yet. The command would be: git show -s --format=%B <merge_commit_hash>^2
```

c. Find the commit that introduced the word "function" in any file



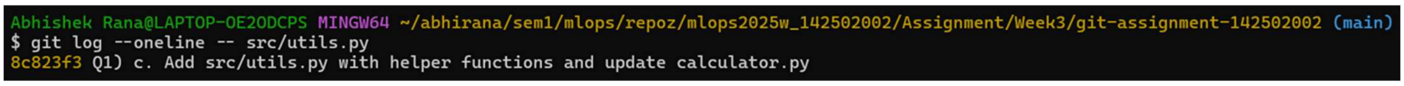
```
Abhishek Rana@LAPTOP-OE20DCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git log -S "function"
```

d. Show differences between the commit where .gitignore was added and its immediate predecessor



```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git diff f374c6a 8c823f3
diff --git a/.gitignore b/.gitignore
deleted file mode 100644
index ab01e86..0000000
--- a/.gitignore
+++ /dev/null
@@ -1,2 +0,0 @@
-*.log
-temp/
\ No newline at end of file
```

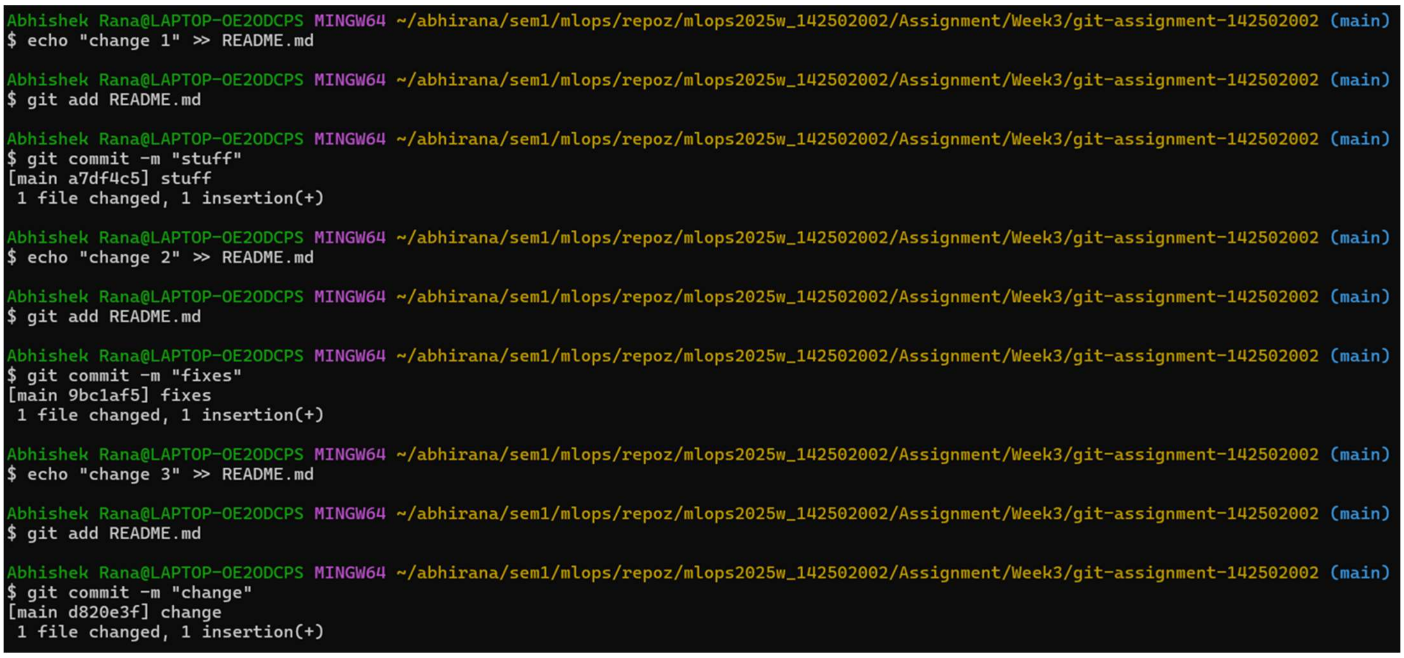
e. List all commits that modified src/utils.py specifically



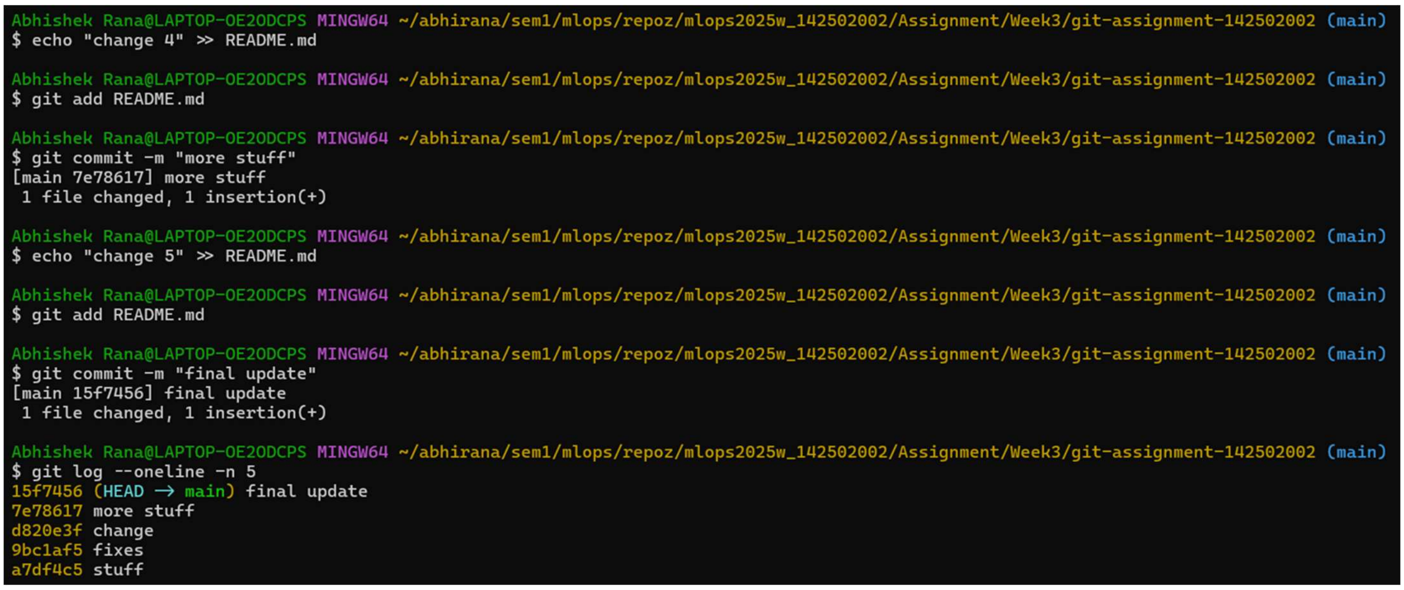
```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git log --oneline -- src/utils.py
8c823f3 Q1) c. Add src/utils.py with helper functions and update calculator.py
```

3. Commit: Attach the screenshot of the outputs for each step.

a. Create 5 commits with intentionally poor commit messages like "stuff", "fixes", "change"



```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ echo "change 1" >> README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git add README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "stuff"
[main a7df4c5] stuff
 1 file changed, 1 insertion(+)

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ echo "change 2" >> README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git add README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "fixes"
[main 9bc1af5] fixes
 1 file changed, 1 insertion(+)

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ echo "change 3" >> README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git add README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "change"
[main d820e3f] change
 1 file changed, 1 insertion(+)
```



```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ echo "change 4" >> README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git add README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "more stuff"
[main 7e78617] more stuff
 1 file changed, 1 insertion(+)

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ echo "change 5" >> README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git add README.md

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "final update"
[main 15f7456] final update
 1 file changed, 1 insertion(+)

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git log --oneline -n 5
15f7456 (HEAD -> main) final update
7e78617 more stuff
d820e3f change
9bc1af5 fixes
a7df4c5 stuff
```

b. Use interactive rebase to:

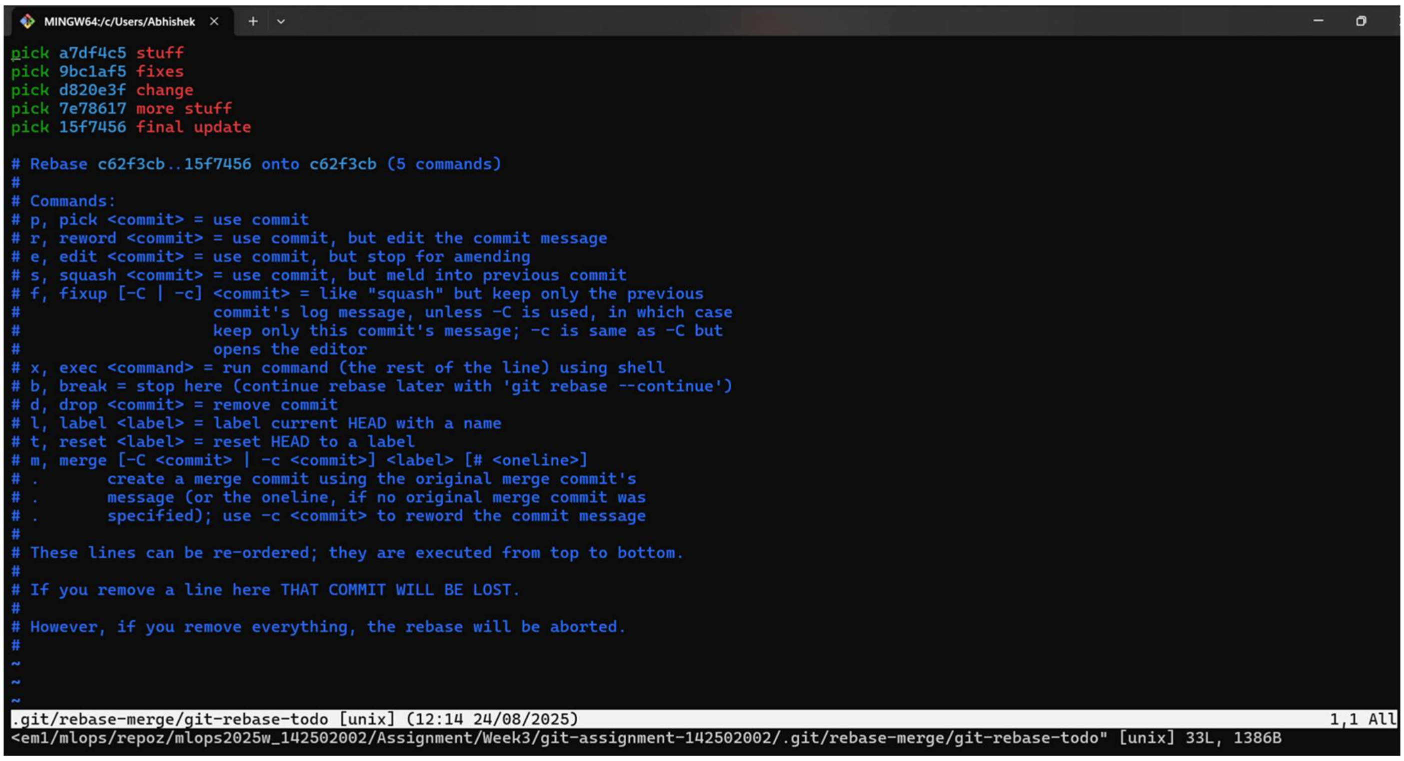
i. Rewrite 3 commit messages to be descriptive

ii. Squash 2 commits into 1

iii. Reorder commits to make logical sense

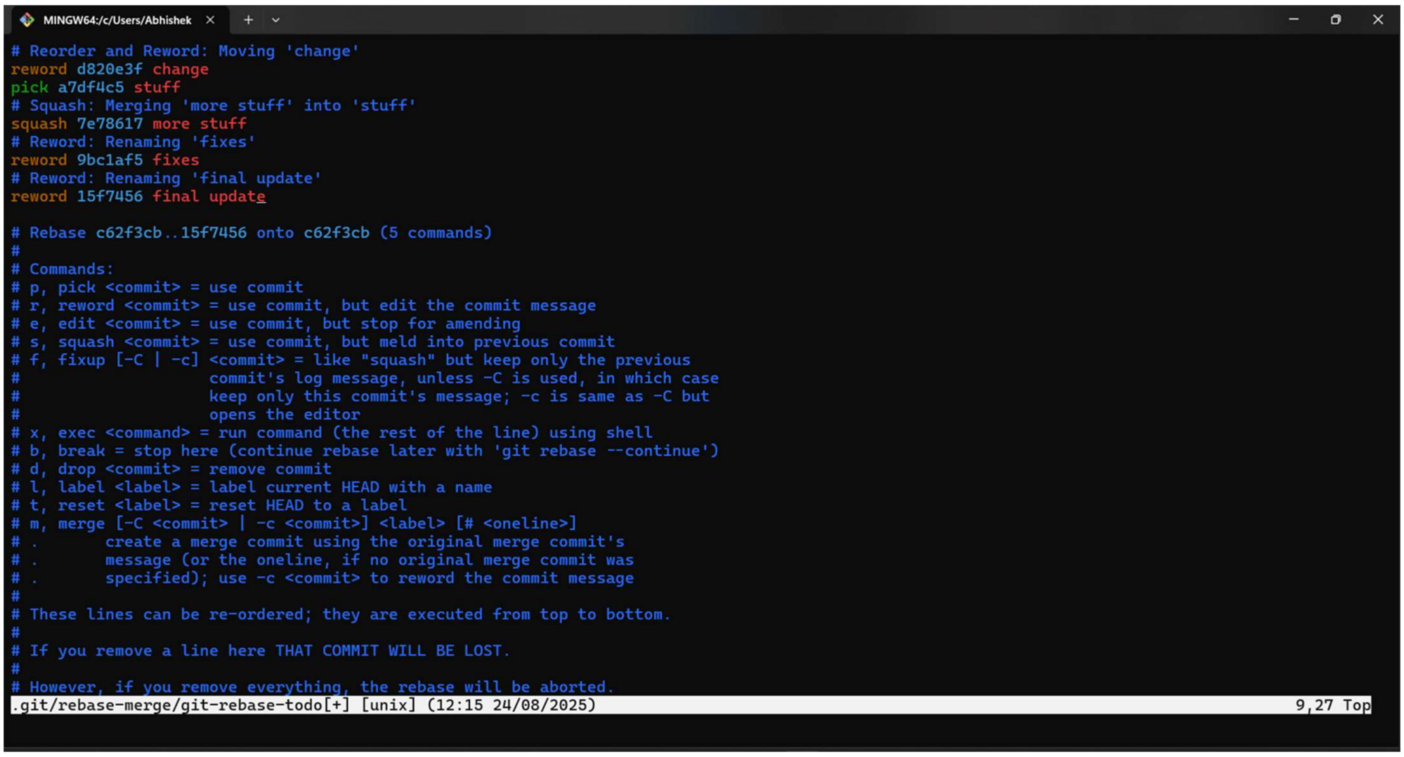
c. Document each step of the interactive rebase process

Started interactive rebase using: git rebase -i HEAD~5



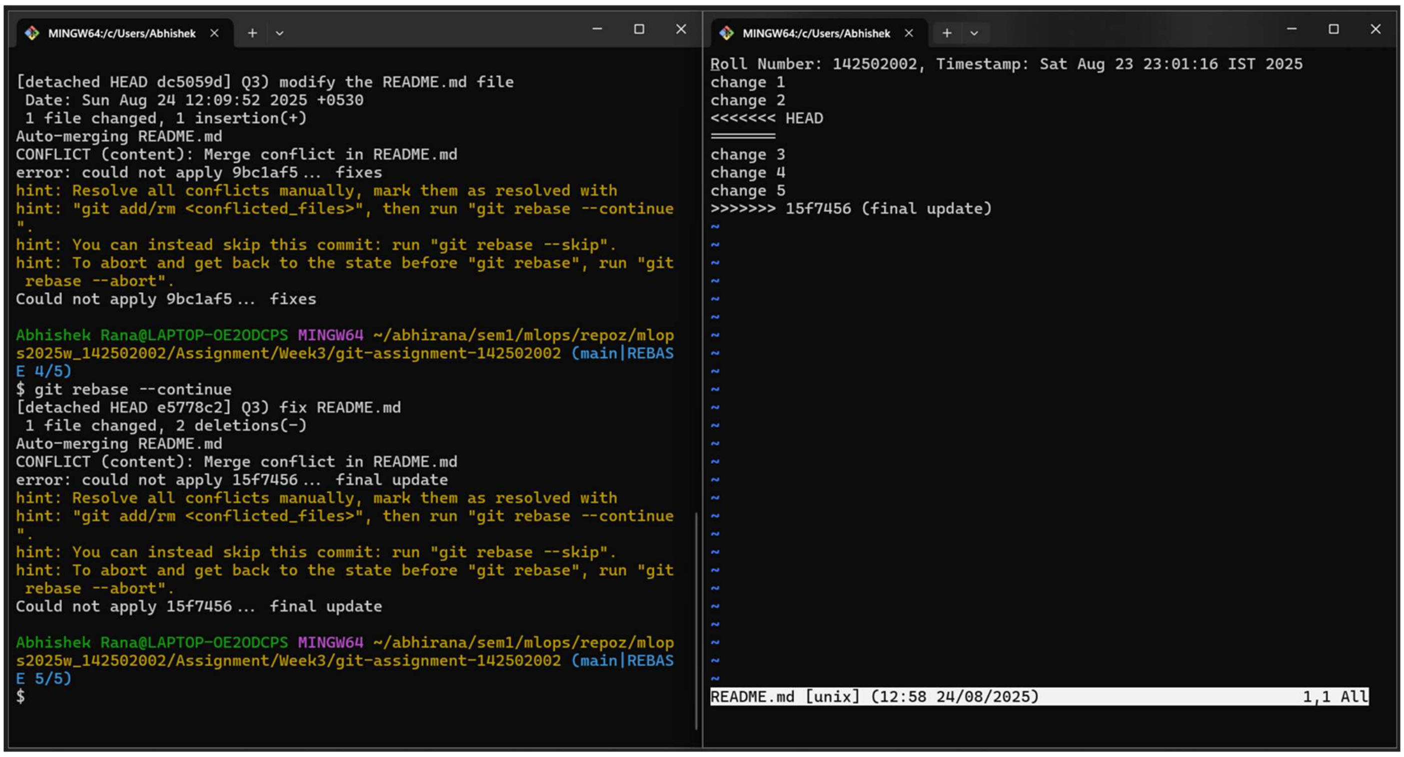
```
pick a7df4c5 stuff
pick 9bc1af5 fixes
pick d820e3f change
pick 7e78617 more stuff
pick 15f7456 final update

# Rebase c62f3cb..15f7456 onto c62f3cb (5 commands)
#
# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup [-C | -c] <commit> = like "squash" but keep only the previous
#                    commit's log message, unless -C is used, in which case
#                    keep only this commit's message; -c is same as -C but
#                    opens the editor
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .       create a merge commit using the original merge commit's
# .       message (or the oneline, if no original merge commit was
# .       specified); use -c <commit> to reword the commit message
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
#
~
~
~
.git/rebase-merge/git-rebase-todo [unix] (12:14 24/08/2025)                                      1,1 All
<em1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002/.git/rebase-merge/git-rebase-todo" [unix] 33L, 1386B
```

Made the relevant changes as per the question

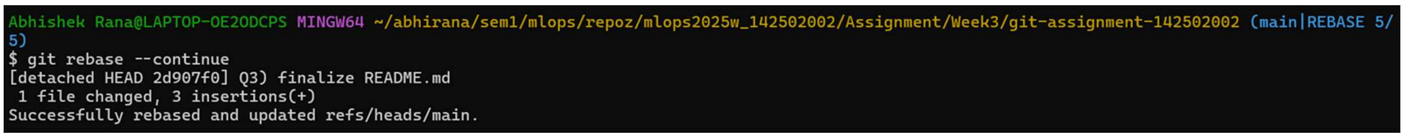


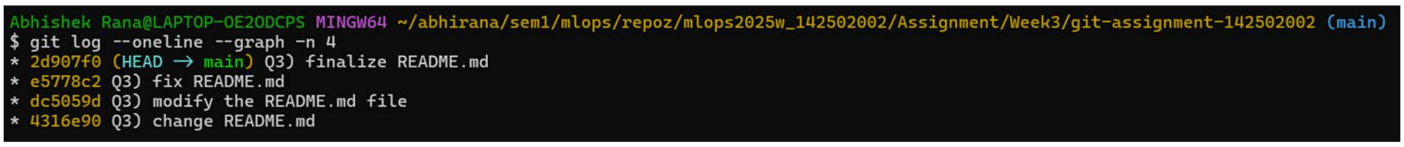
```
# Reorder and Reword: Moving 'change'
reword d820e3f change
pick a7df4c5 stuff
# Squash: Merging 'more stuff' into 'stuff'
squash 7e78617 more stuff
# Reword: Renaming 'fixes'
reword 9bc1af5 fixes
# Reword: Renaming 'final update'
reword 15f7456 final update

# Rebase c62f3cb..15f7456 onto c62f3cb (5 commands)
#
# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup [-C | -c] <commit> = like "squash" but keep only the previous
#                    commit's log message, unless -C is used, in which case
#                    keep only this commit's message; -c is same as -C but
#                    opens the editor
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .       create a merge commit using the original merge commit's
# .       message (or the oneline, if no original merge commit was
# .       specified); use -c <commit> to reword the commit message
#
# These lines can be re-ordered; they are executed from top to bottom.
#
# If you remove a line here THAT COMMIT WILL BE LOST.
#
# However, if you remove everything, the rebase will be aborted.
.git/rebase-merge/git-rebase-todo[+] [unix] (12:15 24/08/2025)                               9,27 Top
```

Resolved merge conflicts and rewrote commit message


```
[detached HEAD dc5059d] Q3) modify the README.md file
 Date: Sun Aug 24 12:09:52 2025 +0530
 1 file changed, 1 insertion(+)
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
error: could not apply 9bc1af5... fixes
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue
".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git
 rebase --abort".
Could not apply 9bc1af5... fixes

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlop
s2025w_142502002/Assignment/Week3/git-assignment-142502002 (main|REBAS
E 4/5)
$ git rebase --continue
[detached HEAD e5778c2] Q3) fix README.md
 1 file changed, 2 deletions(-)
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
error: could not apply 15f7456... final update
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue
".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git
 rebase --abort".
Could not apply 15f7456... final update

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlop
s2025w_142502002/Assignment/Week3/git-assignment-142502002 (main|REBAS
E 5/5)
$
```

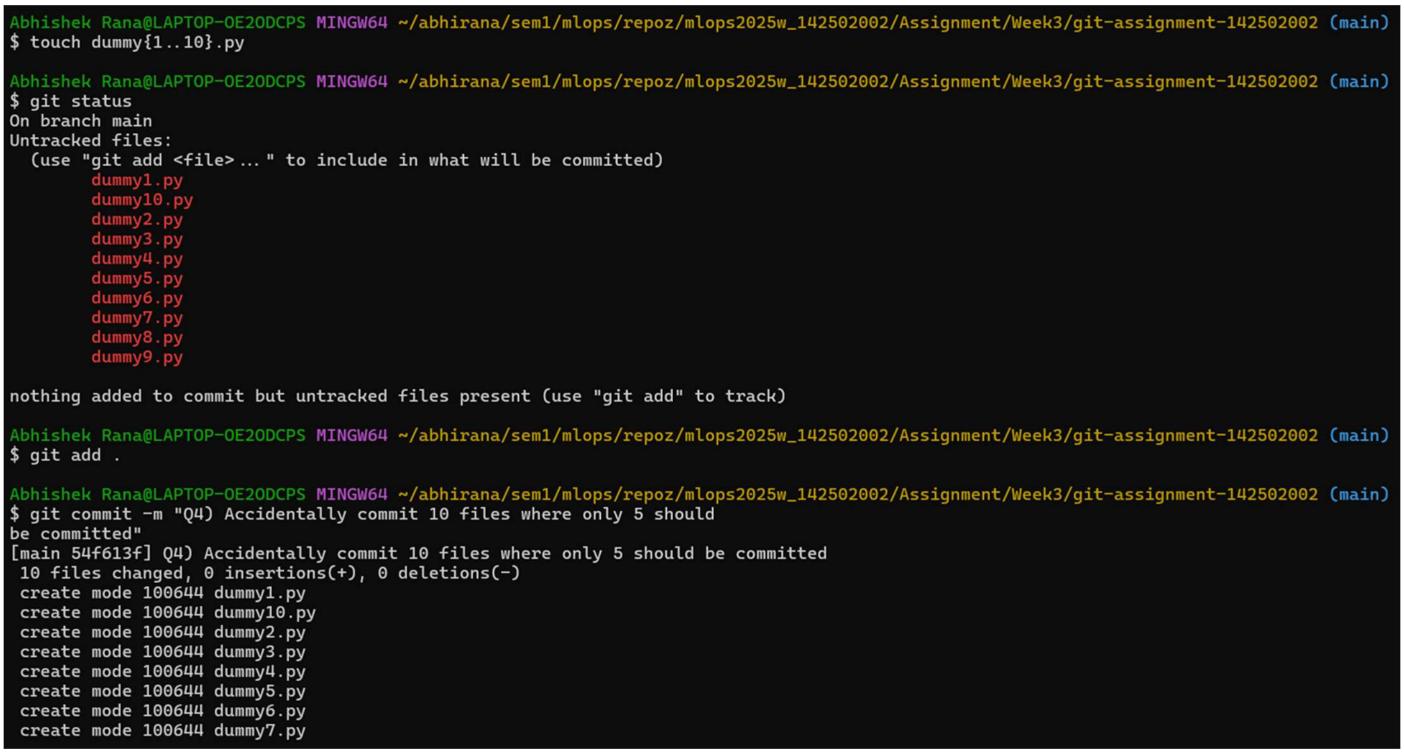
```
Roll Number: 142502002, Timestamp: Sat Aug 23 23:01:16 IST 2025
change 1
change 2
<<<<<<< HEAD
=======
change 3
change 4
change 5
>>>>>>> 15f7456 (final update)
~
~
README.md [unix] (12:58 24/08/2025)                     1,1 All
```


```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main|REBASE 5/
5)
$ git rebase --continue
[detached HEAD 2d907f0] Q3) finalize README.md
 1 file changed, 3 insertions(+)
Successfully rebased and updated refs/heads/main.
```


```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git log --oneline --graph -n 4
* 2d907f0 (HEAD -> main) Q3) finalize README.md
* e5778c2 Q3) fix README.md
* dc5059d Q3) modify the README.md file
* 4316e90 Q3) change README.md
```

4. Mistake: Attach the screenshot of the outputs for each step.

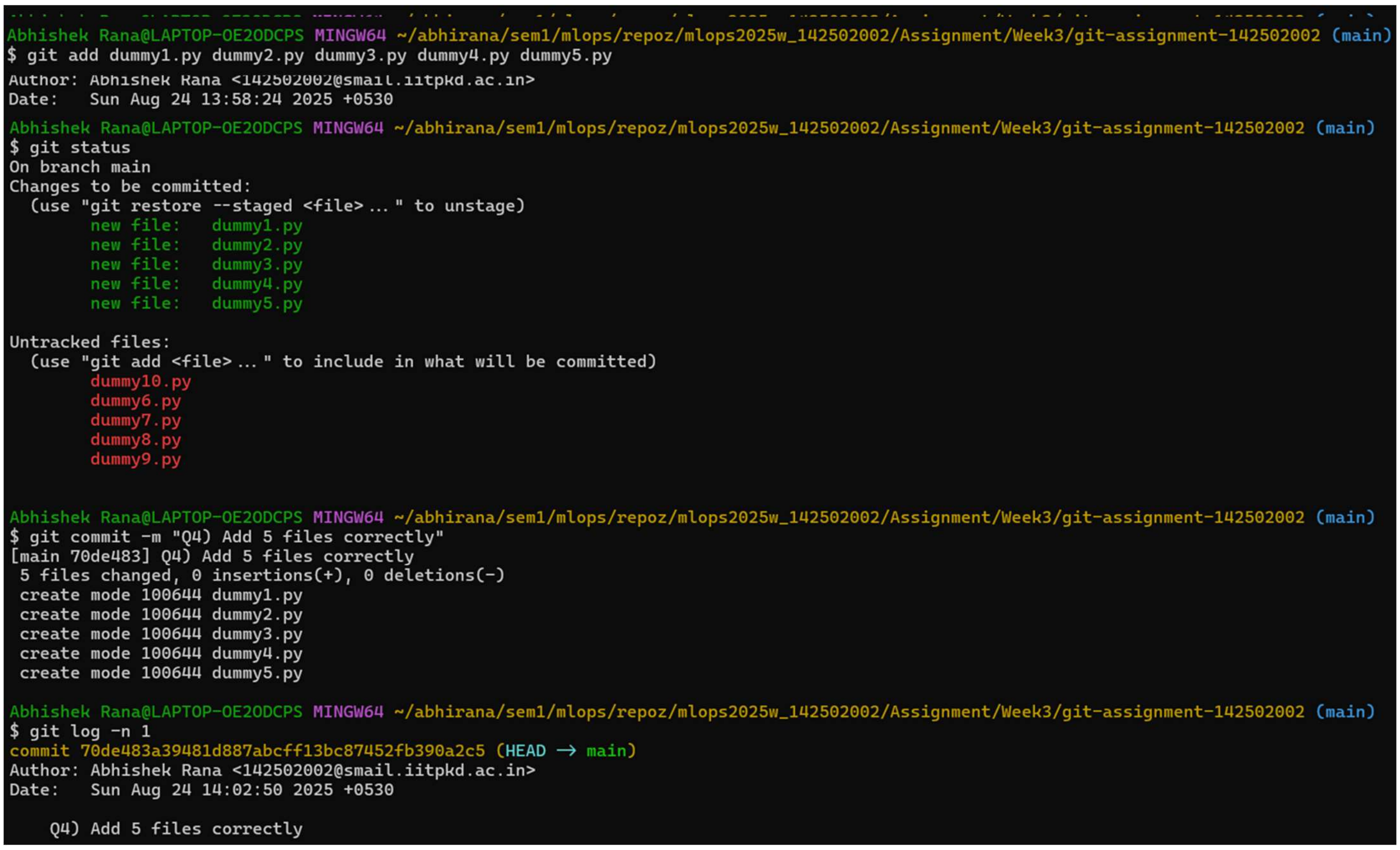
a. Accidentally commit 10 files where only 5 should be committed (You can use dummy files as well such as dummy1.py, dummy2.py, etc)


```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ touch dummy{1..10}.py

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        dummy1.py
        dummy10.py
        dummy2.py
        dummy3.py
        dummy4.py
        dummy5.py
        dummy6.py
        dummy7.py
        dummy8.py
        dummy9.py

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

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git add .

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "Q4) Accidentally commit 10 files where only 5 should
be committed"
[main 54f613f] Q4) Accidentally commit 10 files where only 5 should be committed
 10 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 dummy1.py
 create mode 100644 dummy10.py
 create mode 100644 dummy2.py
 create mode 100644 dummy3.py
 create mode 100644 dummy4.py
 create mode 100644 dummy5.py
 create mode 100644 dummy6.py
 create mode 100644 dummy7.py
```

b. The commit was NOT pushed yet

did not push the commit

c. Use git reset to uncommit, then re-commit correctly



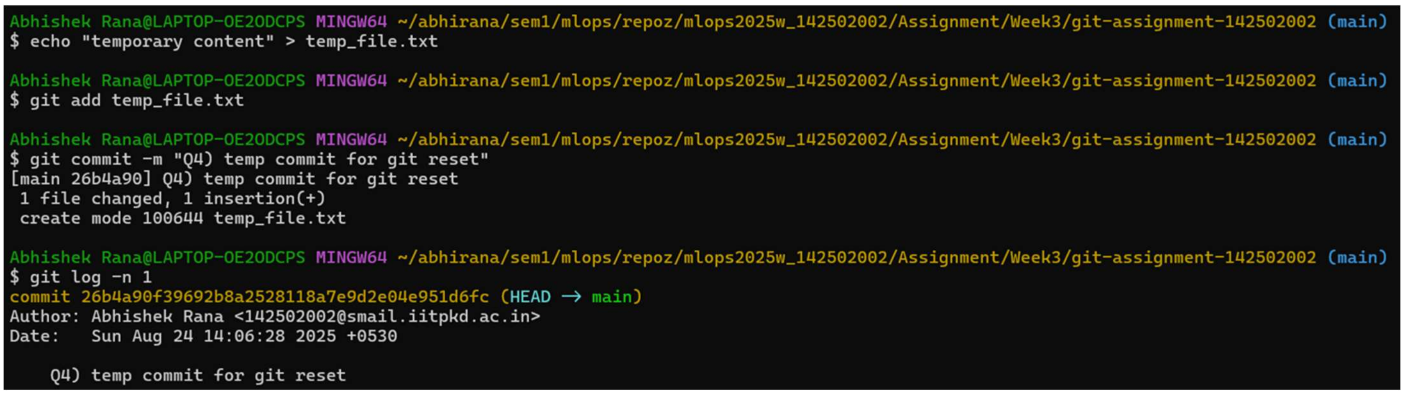
```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git add dummy1.py dummy2.py dummy3.py dummy4.py dummy5.py
Author: Abhishek Rana <142502002@smail.iitpkd.ac.in>
Date:   Sun Aug 24 13:58:24 2025 +0530

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   dummy1.py
        new file:   dummy2.py
        new file:   dummy3.py
        new file:   dummy4.py
        new file:   dummy5.py

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        dummy10.py
        dummy6.py
        dummy7.py
        dummy8.py
        dummy9.py


Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "Q4) Add 5 files correctly"
[main 70de483] Q4) Add 5 files correctly
 5 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 dummy1.py
 create mode 100644 dummy2.py
 create mode 100644 dummy3.py
 create mode 100644 dummy4.py
 create mode 100644 dummy5.py

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git log -n 1
commit 70de483a39481d887abcff13bc87452fb390a2c5 (HEAD -> main)
Author: Abhishek Rana <142502002@smail.iitpkd.ac.in>
Date:   Sun Aug 24 14:02:50 2025 +0530

    Q4) Add 5 files correctly
```

d. Show working directory and staging area status throughout

Shown above

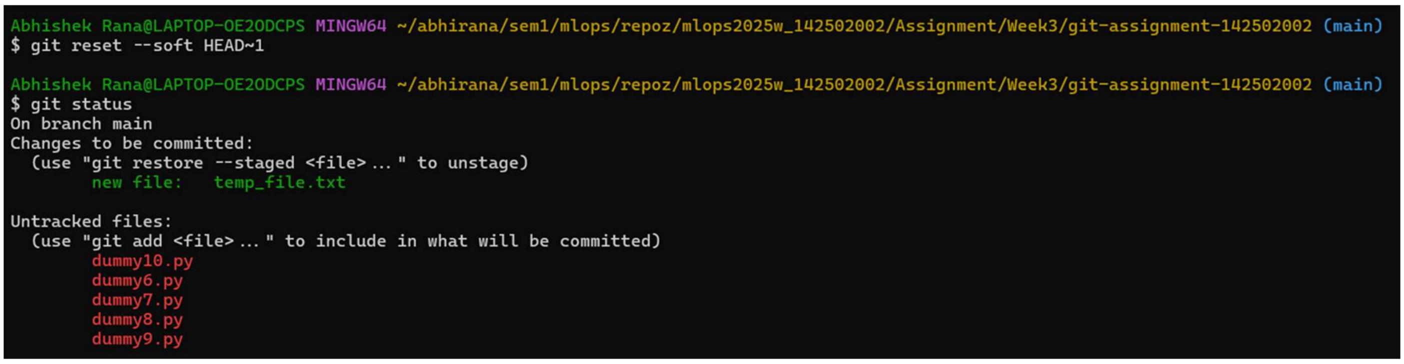
e. Show difference between git reset --soft, --mixed, and --hard

Creating a temporary file and commit it to have something to reset



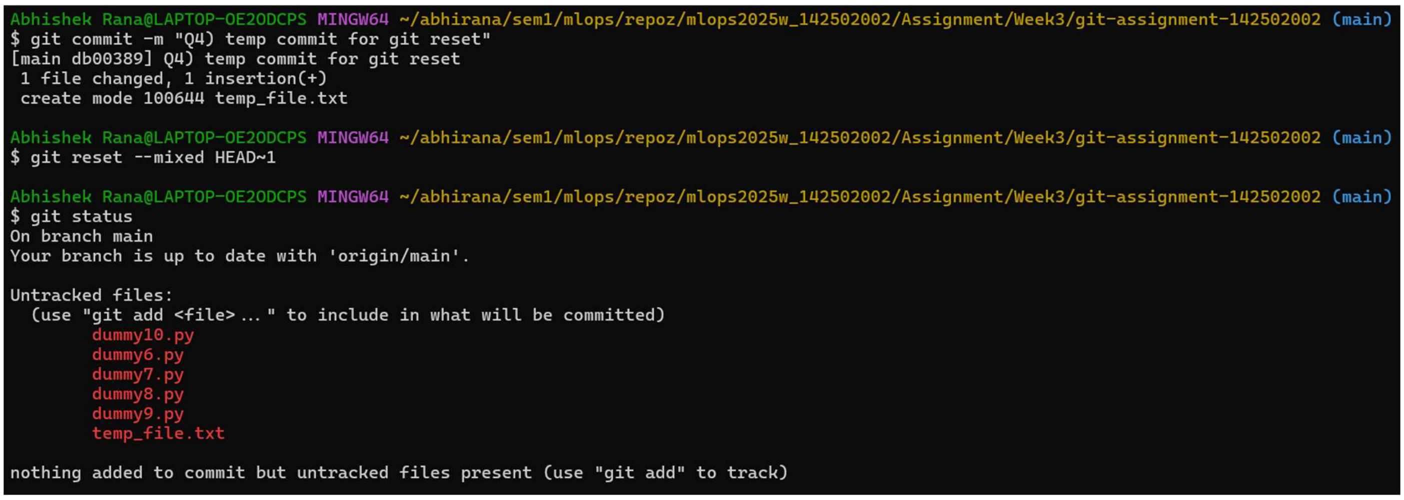
```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ echo "temporary content" > temp_file.txt

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git add temp_file.txt

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "Q4) temp commit for git reset"
[main 26b4a90] Q4) temp commit for git reset
 1 file changed, 1 insertion(+)
 create mode 100644 temp_file.txt

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git log -n 1
commit 26b4a90f39692b8a2528118a7e9d2e04e951d6fc (HEAD -> main)
Author: Abhishek Rana <142502002@smail.iitpkd.ac.in>
Date:   Sun Aug 24 14:06:28 2025 +0530

    Q4) temp commit for git reset
```

git reset –soft HEAD~1: un-commits, and keeps changes staged



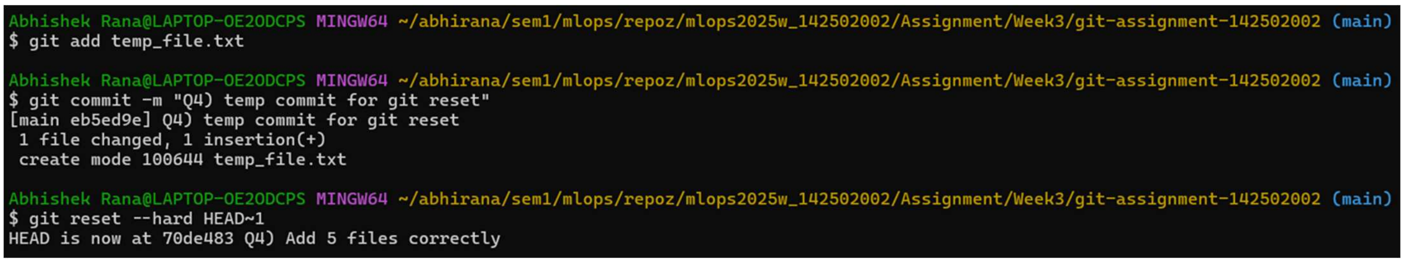
```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git reset --soft HEAD~1

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   temp_file.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        dummy10.py
        dummy6.py
        dummy7.py
        dummy8.py
        dummy9.py
```

git reset –mixed HEAD~1: (the default) un-commits, and moves changes to working directory

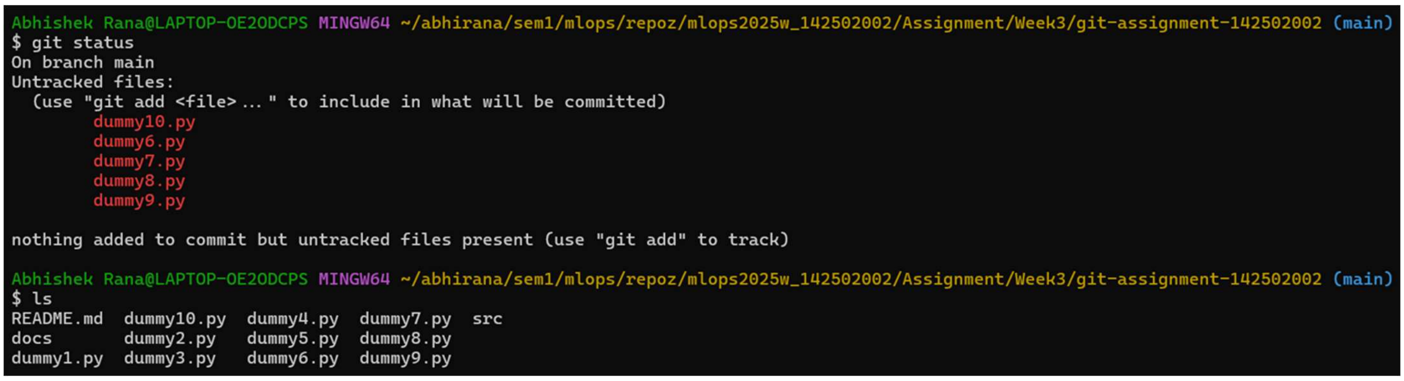


```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "Q4) temp commit for git reset"
[main db00389] Q4) temp commit for git reset
 1 file changed, 1 insertion(+)
 create mode 100644 temp_file.txt

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git reset --mixed HEAD~1

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        dummy10.py
        dummy6.py
        dummy7.py
        dummy8.py
        dummy9.py
        temp_file.txt

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

git reset --hard HEAD~1: un-commits, and deletes changes



```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git add temp_file.txt

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git commit -m "Q4) temp commit for git reset"
[main eb5ed9e] Q4) temp commit for git reset
 1 file changed, 1 insertion(+)
 create mode 100644 temp_file.txt

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git reset --hard HEAD~1
HEAD is now at 70de483 Q4) Add 5 files correctly
```



```
Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        dummy10.py
        dummy6.py
        dummy7.py
        dummy8.py
        dummy9.py

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

Abhishek Rana@LAPTOP-OE2ODCPS MINGW64 ~/abhirana/sem1/mlops/repoz/mlops2025w_142502002/Assignment/Week3/git-assignment-142502002 (main)
$ ls
README.md  dummy10.py  dummy4.py  dummy7.py  src
docs       dummy2.py   dummy5.py  dummy8.py
dummy1.py  dummy3.py   dummy6.py  dummy9.py
```