**Lab Exercise 2- Working with Git Reset**

**Objective**

* Learn how to use git reset to modify the commit history, unstage files, or discard changes.
* Understand the differences between --soft, --mixed, and --hard reset modes.

**Prerequisites**

1. Install Git on your system.
2. Set up a Git repository:

git init git-reset-lab

cd git-reset-lab

**Steps**

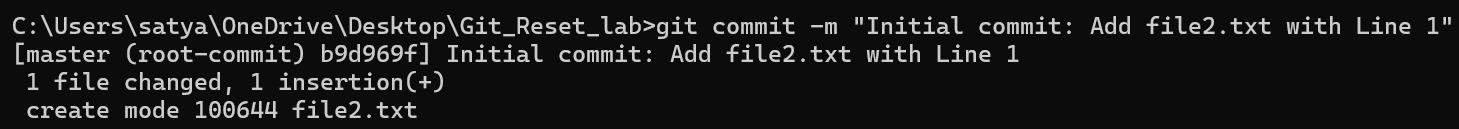
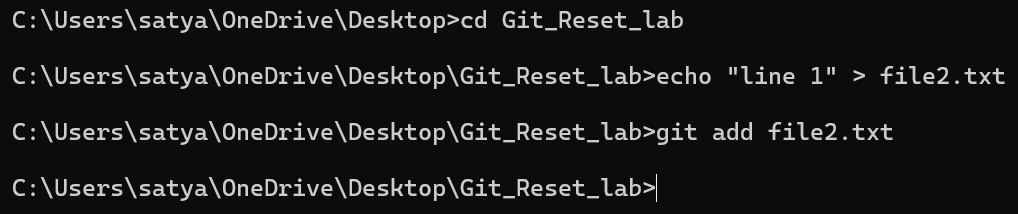
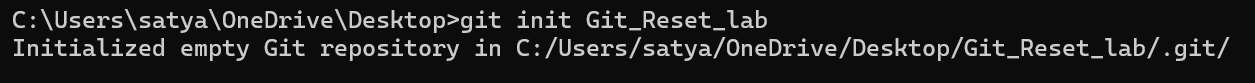
**1. Set Up the Repository**

1. Create and commit an initial file:

echo "Line 1" > file.txt

git add file.txt

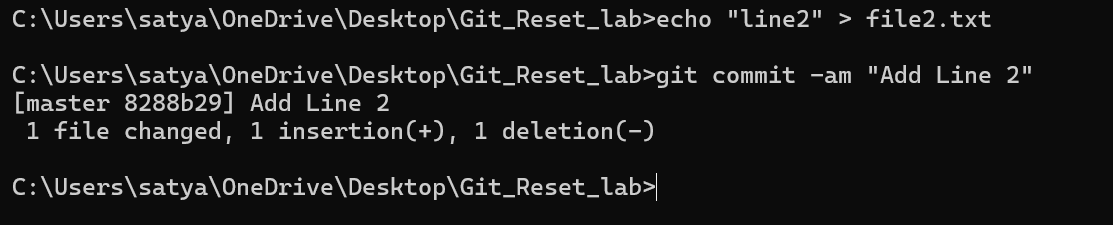
git commit -m "Initial commit: Add Line 1"



1. Add a second change:

echo "Line 2" >> file.txt

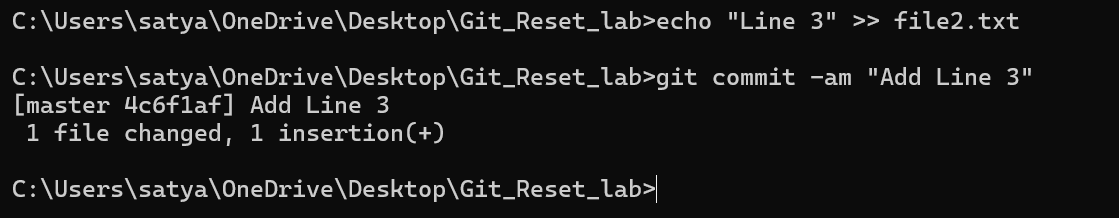
git commit -am "Add Line 2"



1. Add a third change:

echo "Line 3" >> file.txt

git commit -am "Add Line 3"



1. Check the commit history:

git log –oneline



**2. Use git reset --soft**

This mode moves the HEAD pointer to an earlier commit but keeps the changes in the staging area.

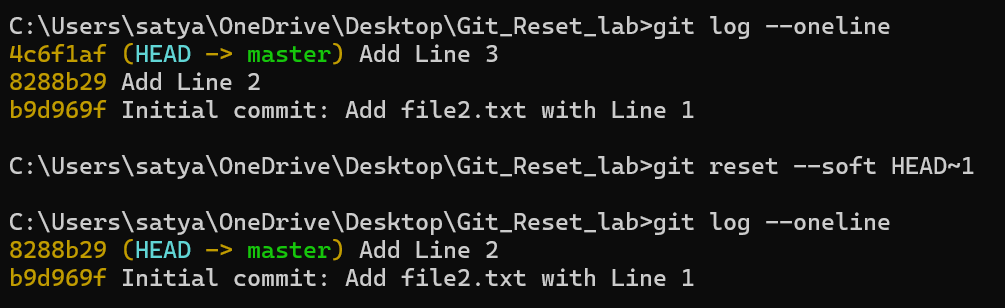
1. Reset to the second commit:

git reset --soft HEAD~1

1. Check the commit history:

git log –oneline

Output:

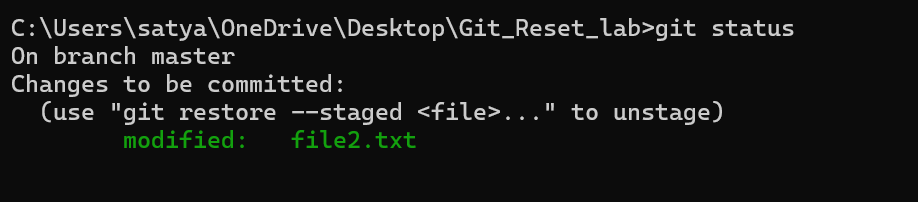


1. Verify the staged changes:

git status

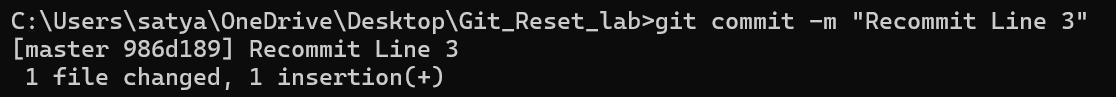
Output:

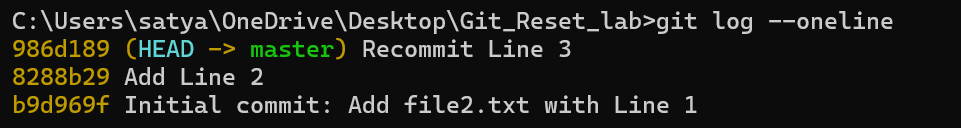
Changes to be committed:



1. If needed, re-commit the changes:

git commit -m "Recommit Line 3"



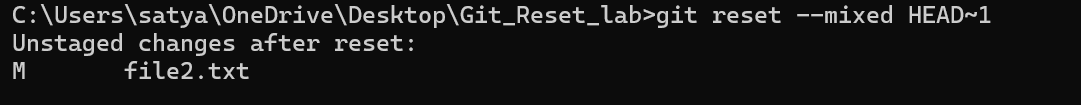


**3. Use git reset --mixed**

This mode moves the HEAD pointer and unstages the changes but keeps them in the working directory.

1. Reset to the first commit:

git reset --mixed HEAD~1



1. Check the commit history:

git log --oneline

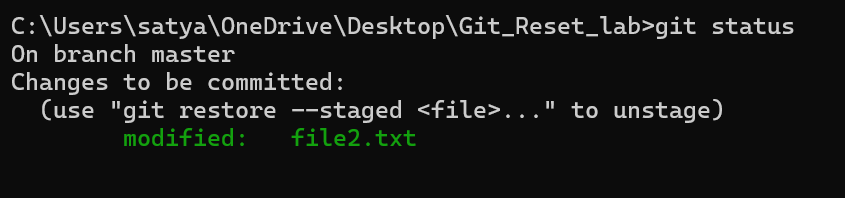
Output:



1. Verify the changes in the working directory:

git status

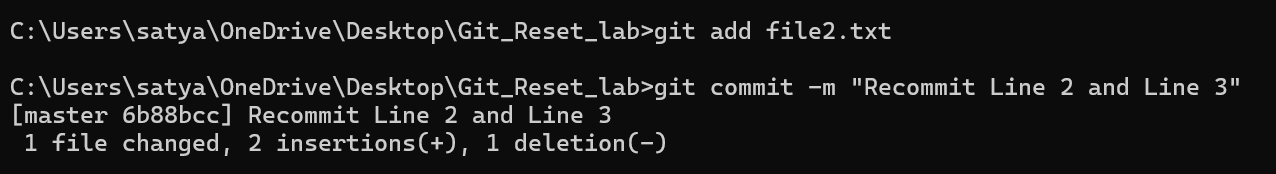
Output:



1. If needed, stage and re-commit:

git add file.txt

git commit -m "Recommit Line 2 and Line 3"

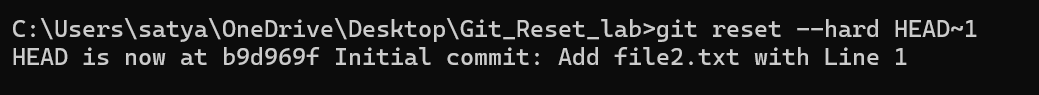


**4. Use git reset --hard**

This mode moves the HEAD pointer and discards all changes in the staging area and working directory.

1. Reset to the initial commit:

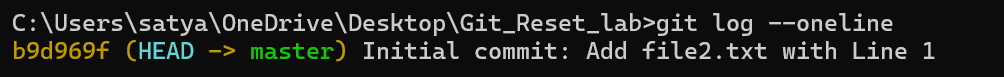
git reset --hard HEAD~1



1. Check the commit history:

git log –oneline

Output:



1. Verify the working directory:

cat file.txt

Output:



**5. Use git reset with a Commit Hash**

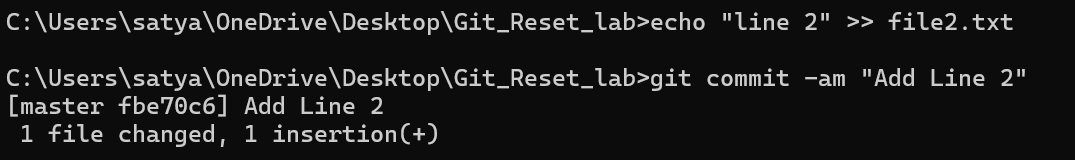
1. Add some changes for demonstration:

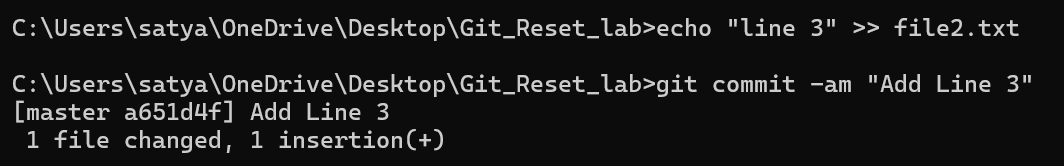
echo "Line 2" >> file.txt

git commit -am "Add Line 2"

echo "Line 3" >> file.txt

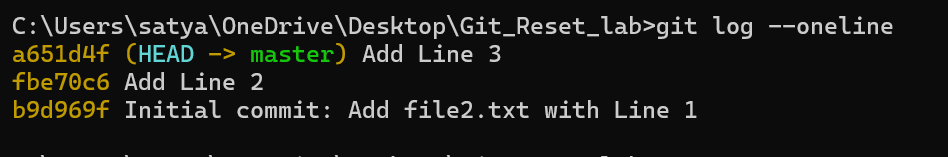
git commit -am "Add Line 3"





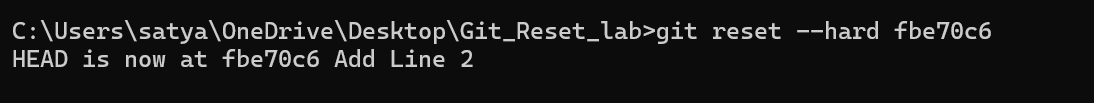
1. Get the commit hash for the initial commit:

git log –oneline



1. Reset to the initial commit using the hash:

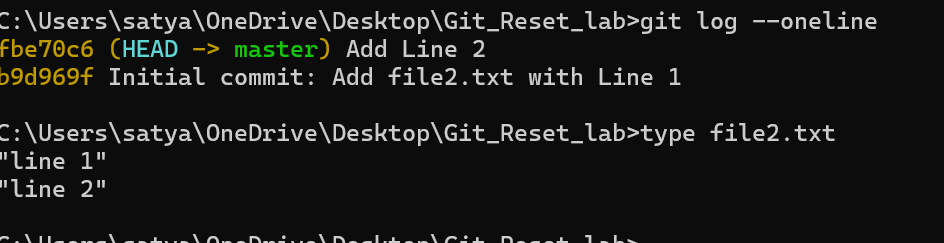
git reset --hard <commit-hash>



1. Verify the working directory and commit history:

git log --oneline

cat file.txt



**Summary of Commands**

| **Mode** | **Effect** | **Command Example** |
| --- | --- | --- |
| --soft | Moves HEAD, keeps changes staged. | git reset --soft HEAD~1 |
| --mixed | Moves HEAD, unstages changes, keeps them in working dir. | git reset --mixed HEAD~1 |
| --hard | Moves HEAD, discards all changes in staging and working dir. | git reset --hard HEAD~1 |