Lab Exercise 2- Working with Git Reset

1. Set up a Git repository:

```
mohdanas@Mohds-MacBook-Air DEVSECOPS_LAB % mkdir git-reset-lab
mohdanas@Mohds-MacBook-Air DEVSECOPS_LAB % git init git-reset-lab
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /Users/mohdanas/Desktop/DEVSECOPS_LAB/git-reset-lab/.git/
```

2. Create and commit an initial file:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % echo "Line 1 " > file.txt
mohdanas@Mohds-MacBook-Air git-reset-lab % git add file.txt
mohdanas@Mohds-MacBook-Air git-reset-lab % git commit -m "Initial commit:Add Line 1"
```

3. Add a second change:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % echo "Line 2" >> file.txt
mohdanas@Mohds-MacBook-Air git-reset-lab % git commit -am "Add Line 2"
[master 1ff0085] Add Line 2
 1 file changed, 1 insertion(+)
```

4. Add a third change:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % echo "Line 3" >> file.txt
mohdanas@Mohds-MacBook-Air git-reset-lab % git commit -am "Add Line 3"
[master 6b3df58] Add Line 3
 1 file changed, 1 insertion(+)
```

6. Check the commit history:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git log --oneline
6b3df58 (HEAD -> master) Add Line 3
1ff0085 Add Line 2
8e9285e Initial commit:Add Line 1
```

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:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git reset --soft HEAD~1
```

2. Check the commit history:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git log --oneline
1ff0085 (HEAD -> master) Add Line 2
8e9285e Initial commit:Add Line 1
mohdanas@Mohds-MacBook-Air git-reset-lab % git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   file.txt
```

3. Verify the staged changes:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git commit -m "Recommit Line 3"
[master 17c7d90] Recommit Line 3
 1 file changed, 1 insertion(+)
```

---

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:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git reset --mixed HEAD~1
Unstaged changes after reset:
M       file.txt
```

2. Check the commit history:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git log --oneline
1ff0085 (HEAD -> master) Add Line 2
8e9285e Initial commit:Add Line 1
```

3. Verify the changes in the working directory:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git status

On branch master
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:   file.txt

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

4. If needed, stage and re-commit:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git add file.txt
mohdanas@Mohds-MacBook-Air git-reset-lab % git commit -m "Recommit Line 2 and Line 3"
[master 7e3a0f3] Recommit Line 2 and Line 3
 1 file changed, 1 insertion(+)
```

---

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:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git reset --hard HEAD~1
HEAD is now at 1ff0085 Add Line 2
```

2. Check the commit history:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git log --oneline
1ff0085 (HEAD -> master) Add Line 2
8e9285e Initial commit:Add Line 1
```

3. Verify the working directory:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % cat file.txt
Line 1
Line 2
```

5. Use git reset with a Commit Hash

1. Add some changes for demonstration:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % echo "Line 2" >> file.txt
mohdanas@Mohds-MacBook-Air git-reset-lab % git commit -am "Add Line 2"
[master 6bab4f8] Add Line 2
 1 file changed, 1 insertion(+)
mohdanas@Mohds-MacBook-Air git-reset-lab % echo "Line 3" >> file.txt

mohdanas@Mohds-MacBook-Air git-reset-lab % git commit -am "Add Line 3"

[master 61a60cb] Add Line 3
 1 file changed, 1 insertion(+)
```

2. Get the commit hash for the initial commit:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git log --oneline
61a60cb (HEAD -> master) Add Line 3
6bab4f8 Add Line 2
1ff0085 Add Line 2
8e9285e Initial commit:Add Line 1
```

3. Reset to the initial commit using the hash:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git reset --hard 6bab4f8
HEAD is now at 6bab4f8 Add Line 2
```

4. Verify the working directory and commit history:

```
mohdanas@Mohds-MacBook-Air git-reset-lab % git log --oneline
6bab4f8 (HEAD -> master) Add Line 2
1ff0085 Add Line 2
8e9285e Initial commit:Add Line 1
mohdanas@Mohds-MacBook-Air git-reset-lab % cat file.txt
Line 1
Line 2
Line 2
```