😊 Rollback

In Git, a "rollback" typically refers to reverting changes made in a commit or a series of commits to a previous state. This is often done when you want to undo changes that were introduced in one or more commits without simply deleting them from history, which could potentially cause issues with collaboration or tracking changes.

There are a few ways to accomplish a rollback in Git:

1. **Revert:** This creates a new commit that undoes the changes introduced by a previous commit or commits. It's a safe way to undo changes because it doesn't alter the commit history.
2. **Reset:** This command can be used to reset the current branch to a specific commit, effectively removing any commits that came after that point in history. It's a more aggressive approach and should be used with caution, especially if you've already pushed your changes to a shared repository.
3. **Checkout:** You can use the git checkout command to restore specific files or even entire directories to their state at a certain commit.

Each method has its use cases and implications, so it's important to understand the differences and choose the one that best fits your situation.

In this Git lab exercise, we're exploring the rollback feature to understand how to revert changes effectively. The goal is to learn different methods such as `git revert`, `git reset`, and `git checkout` for rolling back changes in a Git repository. By doing so, we gain the ability to undo alterations to files or entire commits while preserving the commit history, ensuring better version control management and collaboration.

😄 To begin with the Lab:

1. In this lab, we are going to learn about the Rollback feature of git.
2. First, we are going to add some content in any of our files. Below you can see that first we listed all our objects then we added some content in one of our file. You can take any code from Google and paste it in the file.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ ls
gsat9/  insat1a/  moon10.py  moon11.py  scross1/  tesla1  tesla2  tesla3  tesla4

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ vim tesla1

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ cat tesla1
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```

3. Then we used the git checkout command to rollback all the changes we did in our file. So, git checkout and the file name is going to just roll back if you made the changes have and have not gone into staging or commit area.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git checkout tesla1
Updated 1 path from the index

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ cat tesla1

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$
```

4. So, once again we made some changes in our same file

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ vim tesla1

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ cat tesla1
<!DOCTYPE html>
<html>
<body>

<h2>HTML Buttons</h2>
<p>HTML buttons are defined with the button tag:</p>

<button>Click me</button>

</body>
</html>
```

5. After that we run the git status command to show the status of our file. Then we used git diff command to actually see the difference in the file that you have changed. You can see that below too in the snapshot.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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:   tesla1

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

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git diff
warning: in the working copy of 'tesla1', LF will be replaced by CRLF the next time Git touches it
diff --git a/tesla1 b/tesla1
index e69de29..1db2d87 100644
--- a/tesla1
+++ b/tesla1
@@ -0,0 +1,11 @@
+<!DOCTYPE html>
+<html>
+<body>
+
+<h2>HTML Buttons</h2>
+<p>HTML buttons are defined with the button tag:</p>
+
+<button>Click me</button>
+
+</body>
+</html>
```

6. Now if you want to roll back you can do that by using the git checkout command.
7. So, below we have used the git add . command then we checked the status. It changed to green from red.
8. After that we used git diff --cached, so it's going to do the check. The difference is the file that was staged. So, if you have already staged it, you want to see the difference between the current commit and the next one that is going to happen.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git add .
warning: in the working copy of 'tesla1', LF will be replaced by CRLF the next time Git touches it

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   tesla1


PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git diff --cached
diff --git a/tesla1 b/tesla1
index e69de29..1db2d87 100644
--- a/tesla1
+++ b/tesla1
@@ -0,0 +1,11 @@
+<!DOCTYPE html>
+<html>
+<body>
+
+<h2>HTML Buttons</h2>
+<p>HTML buttons are defined with the button tag:</p>
+
+<button>Click me</button>
+
+</body>
+</html>
```

9. Now we used the git restore --staged command. This command rollback to where we started before staging.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git restore --staged tesla1

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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:   tesla1

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

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git diff
warning: in the working copy of 'tesla1', LF will be replaced by CRLF the next time Git touches it
diff --git a/tesla1 b/tesla1
index e69de29..1db2d87 100644
--- a/tesla1
+++ b/tesla1
@@ -0,0 +1,11 @@
+<!DOCTYPE html>
+<html>
+<body>
+
+<h2>HTML Buttons</h2>
+<p>HTML buttons are defined with the button tag:</p>
+
+<button>Click me</button>
+
+</body>
+</html>
```

10. Now this time we used the git add command and we committed the changes. You can see that the status is changed to nothing to commit and we can not see any data using git diff command.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git add .
warning: in the working copy of 'tesla1', LF will be replaced by CRLF the next time Git touches it

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git commit -m "rollback"
[main 7c97fc8] rollback
 1 file changed, 11 insertions(+)

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git diff

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git diff --cached
```

11. After that we used git log command to see that logs here then we used git diff command and here we write the id of previous commit dot dot and the current commit. Then you can see the actual difference between them.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git log --oneline
7c97fc8 (HEAD -> main) rollback
cf41efa (origin/mars1, origin/main, origin/HEAD) more changes
ccc886c added some new moons
b3a4821 some new names
fa53ffb moon names
bccee09 about insat1a
10cf38b new files commit

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git diff cf41efa..7c97fc8
diff --git a/tesla1 b/tesla1
index e69de29..1db2d87 100644
--- a/tesla1
+++ b/tesla1
@@ -0,0 +1,11 @@
+<!DOCTYPE html>
+<html>
+<body>
+
+<h2>HTML Buttons</h2>
+<p>HTML buttons are defined with the button tag:</p>
+
+<button>Click me</button>
+
+</body>
+</html>
```

12. Now if you want to rollback you can use this command git revert HEAD and the ID of your current commit. Then press enter.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git revert HEAD 7c97fc8
```

13. Then it will give you this kind of message and ask you once again if you want to commit the changes or not.

```
MINGW64:/c/Users/PULKIT/D
Revert "rollback"

This reverts commit 7c97fc8ae707238d8bb22f7e9a559b477e6d5ff4.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch main
# Your branch is ahead of 'origin/main' by 1 commit.
#   (use "git push" to publish your local commits)
#
# Revert currently in progress.
#
# Changes to be committed:
#       modified:   tesla1
#
~
```

14. Below you can see that we rolled back to where we started from.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git revert HEAD 7c97fc8
[main 5550142] Revert "rollback"
 1 file changed, 11 deletions(-)

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ cat tesla1

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$
```

1. Now if you write the git log command then you can see the new commit ID for revert. So, it also restores the history of what changes you made. Git revert is a softer way where you are storing the history also.

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git log --oneline
5550142 (HEAD -> main) Revert "rollback"
7c97fc8 rollback
cf41efa (origin/mars1, origin/main, origin/HEAD) more changes
ccc886c added some new moons
b3a4821 some new names
fa53ffb moon names
bccee09 about insat1a
10cf38b new files commit
```

2. Now if you don’t want to store the history then you can give this command.
   git reset --hard
3. So when you revert it is also going to store the history of what changes you made. But if you don't want to store any history whatsoever, then you can use git reset command. Git reset hyphen hyphen hard and you give the commit ID. Now here you go back to that particular commit ID and all the commit ID After that, all the commit history after that is removed, the data and all the history is removed. So this is a more direct way of rollback

```
PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git log --oneline
5550142 (HEAD -> main) Revert "rollback"
7c97fc8 rollback
cf41efa (origin/mars1, origin/main, origin/HEAD) more changes
ccc886c added some new moons
b3a4821 some new names
fa53ffb moon names
bccee09 about insat1a
10cf38b new files commit

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git reset --hard cf41efa
HEAD is now at cf41efa more changes

PULKIT@LAPTOP-G2CAKBK8 MINGW64 ~/Desktop/gitrepository/sattelitework (main)
$ git log --oneline
cf41efa (HEAD -> main, origin/mars1, origin/main, origin/HEAD) more changes
ccc886c added some new moons
b3a4821 some new names
fa53ffb moon names
bccee09 about insat1a
10cf38b new files commit
```