

Name: L Guru Priya

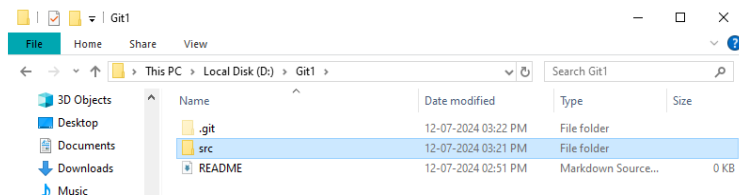
Reg No: 73152113034

Cecsgurupriya25@gmail.com

Exercise 1:

1. Create a new directory and change into it.
2. Use the init command to create a Git repository in that directory.
3. Observe that there is now a .git directory.
4. Create a README file.
5. Look at the output of the status command; the README you created should appear as an untracked file.
6. Use the add command to add the new file to the staging area. Again, look at the output of the status command.
7. Now use the commit command to commit the contents of the staging area.
8. Create a src directory and add a couple of files to it.
9. Use the add command, but name the directory, not the individual files. Use the status command. See how both files have been staged. Commit them.
10. Make a change to one of the files. Use the diff command to view the details of the change.
11. Next, add the changed file, and notice how it moves to the staging area in the status output. Also observe that the diff command you did before using add now gives no output. Why not? What do you have to do to see a diff of the things in the staging area? (Hint: review the slides if you can't remember.)
12. Now – without committing – make another change to the same file you changed in step 10. Look at the status output, and the diff output. Notice how you can have both staged and unstaged changes, even when you're talking about a single file. Observe the difference when you use the add command to stage the latest round of changes. Finally, commit them. You should now have started to get a feel for the staging area.
13. Use the log command in order to see all of the commits you made so far.
14. Use the show command to look at an individual commit. How many characters of the commit identifier can you get away with typing at a minimum?
15. Make a couple more commits, at least one of which should add an extra file.

```
Git CMD
C:\Users\home>d:
D:\>mkdir Git1
D:\>cd Git1
D:\Git1>git init
Initialized empty Git repository in D:/Git1/.git/
D:\Git1>git status
On branch master
```



```
D:\Git1>git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        README.md

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

```
D:\Git1>git add README.md
D:\Git1>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   README.md
```

```
D:\Git1>git commit -m "README Commit"
[master (root-commit) b066f54] README Commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 README.md
```

```
D:\Git1>mkdir src
D:\Git1>git add src
D:\Git1>git status
On branch master

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   src/File1.txt
        new file:   src/File2.txt
```

```
D:\Git1>git commit -m "src commit"
[master da228ad] src commit
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 src/File1.txt
 create mode 100644 src/File2.txt

D:\Git1>git diff
diff --git a/src/File1.txt b/src/File1.txt
index e69de29..5e1c309 100644
--- a/src/File1.txt
+++ b/src/File1.txt
@@ -0,0 +1 @@
+Hello World
\ No newline at end of file
```

```
D:\Git1>git add src/File1.txt
D:\Git1>git status
On branch master

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   src/File1.txt
```

```
D:\Git>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   src/File1.txt

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:   src/File1.txt
```

```
D:\Git>git diff
diff --git a/src/File1.txt b/src/File1.txt
index 5e1c309..fb78c1c 100644
--- a/src/File1.txt
+++ b/src/File1.txt
@@ -1,1 @@
-Hello world
\ No newline at end of file
+Hello Everyone
\ No newline at end of file
```

```
D:\Git>git add src/File1.txt
D:\Git>git diff
D:\Git>git commit -m "text commit"
[master 55b3c2a] text commit
1 file changed, 1 insertion(+)
```

```
D:\Git>git commit -m "text commit"
[master 55b3c2a] text commit
1 file changed, 1 insertion(+)
```

```
Git CMD

D:\Git>git log
commit 55b3c2a4285109276f00516e92bf18fd038c2a13 (HEAD -> master)
Author: GuruPriya <gurupriya2728@gmail.com>
Date:   Fri Jul 12 15:09:48 2024 +0530

    text commit

commit da228ad682835958c23fcf001f5821faf23ce4b5
Author: GuruPriya <gurupriya2728@gmail.com>
Date:   Fri Jul 12 14:54:10 2024 +0530

    src commit

commit b066f54f3a2d6ece623990673ecac4fca946909
Author: GuruPriya <gurupriya2728@gmail.com>
Date:   Fri Jul 12 14:52:26 2024 +0530

    README Commit

D:\Git>git show 55b3c2a4285109276f00516e92bf18fd038c2a13
commit 55b3c2a4285109276f00516e92bf18fd038c2a13 (HEAD -> master)
Author: GuruPriya <gurupriya2728@gmail.com>
Date:   Fri Jul 12 15:09:48 2024 +0530

    text commit

diff --git a/src/File1.txt b/src/File1.txt
index e69de29..fb78c1c 100644
--- a/src/File1.txt
+++ b/src/File1.txt
@@ -0,0 +1 @@
+Hello Everyone
\ No newline at end of file

D:\Git>git diff da228ad
diff --git a/src/File1.txt b/src/File1.txt
index e69de29..fb78c1c 100644
--- a/src/File1.txt
+++ b/src/File1.txt
@@ -0,0 +1 @@
+Hello Everyone
\ No newline at end of file
```

```
D:\Git>git add src/File3.txt
D:\Git>git add src/File4.txt
D:\Git>git commit -m "File Commit"
[master 8c7d6da] File Commit
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 src/File3.txt
create mode 100644 src/File4.txt
```

```
D:\Git>git log
commit 8c7d6da26975f4ebcaf8b124549bfe17b46067df (HEAD -> master)
Author: GuruPriya <gurupriya2728@gmail.com>
Date: Fri Jul 12 15:22:26 2024 +0530

    File Commit

commit 55b3c2a4285109276f00516e92bf18fd038c2a13
Author: GuruPriya <gurupriya2728@gmail.com>
Date: Fri Jul 12 15:09:48 2024 +0530

    text commit

commit da228ad682835958c23fcf001f5821faf23ce4b5
Author: GuruPriya <gurupriya2728@gmail.com>
Date: Fri Jul 12 14:54:10 2024 +0530

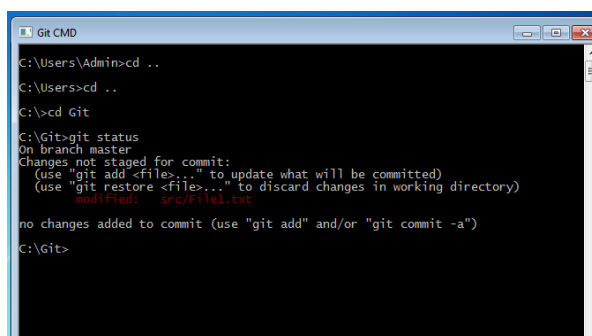
    src commit

commit b066f54f3a2d6ece623990673ecacf4fca946909
Author: GuruPriya <gurupriya2728@gmail.com>
Date: Fri Jul 12 14:52:26 2024 +0530

    README Commit
```

Exercise 2 Main Task

1. Run the status command. Notice how it tells you what branch you are in.
2. Use the branch command to create a new branch.
3. Use the checkout command to switch to it.
4. Make a couple of commits in the branch – perhaps adding a new file and/or editing existing ones.
5. Use the log command to see the latest commits. The two you just made should be at the top of the list.
6. Use the checkout command to switch back to the master branch. Run log again. Notice your commits don't show up now. Check the files also – they should have their original contents.
7. Use the checkout command to switch back to your branch. Use gitk to take a look at the commit graph; notice it's linear.
8. Now checkout the master branch again. Use the merge command to merge your branch in to it. Look for information about it having been a fast-forward merge. Look at git log, and see that there is no merge commit. Take a look in gitk and see how the DAG is linear.
9. Switch back to your branch. Make a couple more commits.
10. Switch back to master. Make a commit there, which should edit a different file from the ones you touched in your branch – to be sure there is no conflict.
11. Now merge your branch again. (Aside: you don't need to do anything to inform Git that you only want to merge things added since your previous merge. Due to the way Git works, that kind of issue simply does not come up, unlike in early versions of Subversion.)
12. Look at git log. Notice that there is a merge commit. Also look in gitk. Notice the DAG now shows how things forked, and then were joined up again by a merge commit.



```
C:\Users\Admin>cd ..
C:\Users>cd ..
C:\>cd Git
C:\Git>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:   src/File1.txt

no changes added to commit (use "git add" and/or "git commit -a")
C:\Git>
```



```
C:\Git>git branch Branch-1
C:\Git>git checkout Branch-1
Switched to branch 'Branch-1'
M       src/File1.txt
C:\Git>git add File.txt
C:\Git>git add File1.txt
```

```
C:\Git>git commit -m "Text file"
[Branch-1 e5e5e37] Text file
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 File.txt
create mode 100644 File1.txt
```

```
C:\Git>git log
commit e5e5e379e54a2ec4d70e892bd7fd43db2af0d69b (HEAD -> Branch-1)
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Sat Jul 13 11:32:58 2024 +0530

    Text file

commit cc42d3462213df409f6d40af093c69806f5fb2bd (master)
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Sat Jul 13 09:54:59 2024 +0530

    File 2 commit

commit a2e72d566d14aa5646dde8527750437b33b6868
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Sat Jul 13 09:51:52 2024 +0530

    File commit

commit 85977d275c97de9e872efb044a8e7ebb0cd8ae16
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Sat Jul 13 09:50:07 2024 +0530

    src commit
```

```
C:\Git>git checkout master
Switched to branch 'master'

C:\Git>git log
commit cc42d3462213df409f6d40af093c69806f5fb2bd (HEAD -> master)
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Sat Jul 13 09:54:59 2024 +0530

    File 2 commit

commit a2e72d566d14aa5646dde8527750437b33b6868
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Sat Jul 13 09:51:52 2024 +0530

    File commit

commit 85977d275c97de9e872efb044a8e7ebb0cd8ae16
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Sat Jul 13 09:50:07 2024 +0530

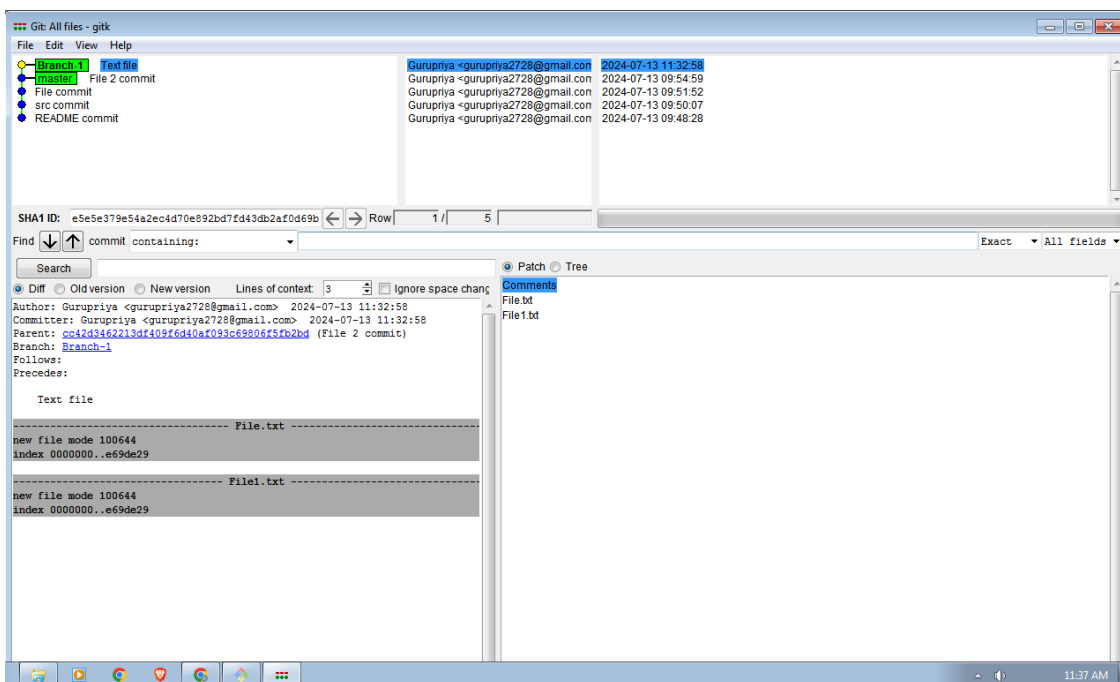
    src commit

commit db78bd3e0a0f17ed101d6f47c204501f22fe95d8
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Sat Jul 13 09:48:28 2024 +0530

    README commit
```

```
C:\Git>git checkout Branch-1
Switched to branch 'Branch-1'

C:\Git>gitk
```



```

C:\Git>git checkout Branch-1
Switched to branch 'Branch-1'

```

```

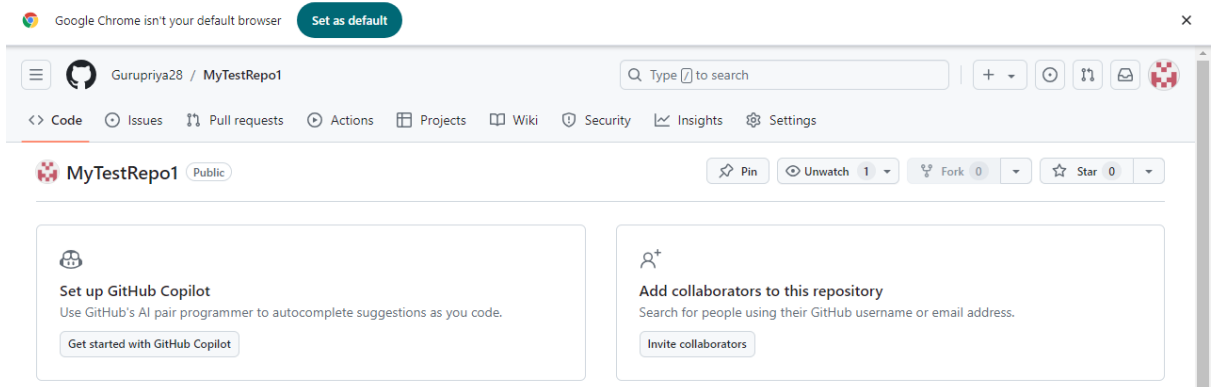
C:\Git>git merge Branch-1
Updating cc42d34..e5e5e37
Fast-forward
 File.txt | 0
 File1.txt | 0
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 File.txt

```


Exercise 3

Main Task

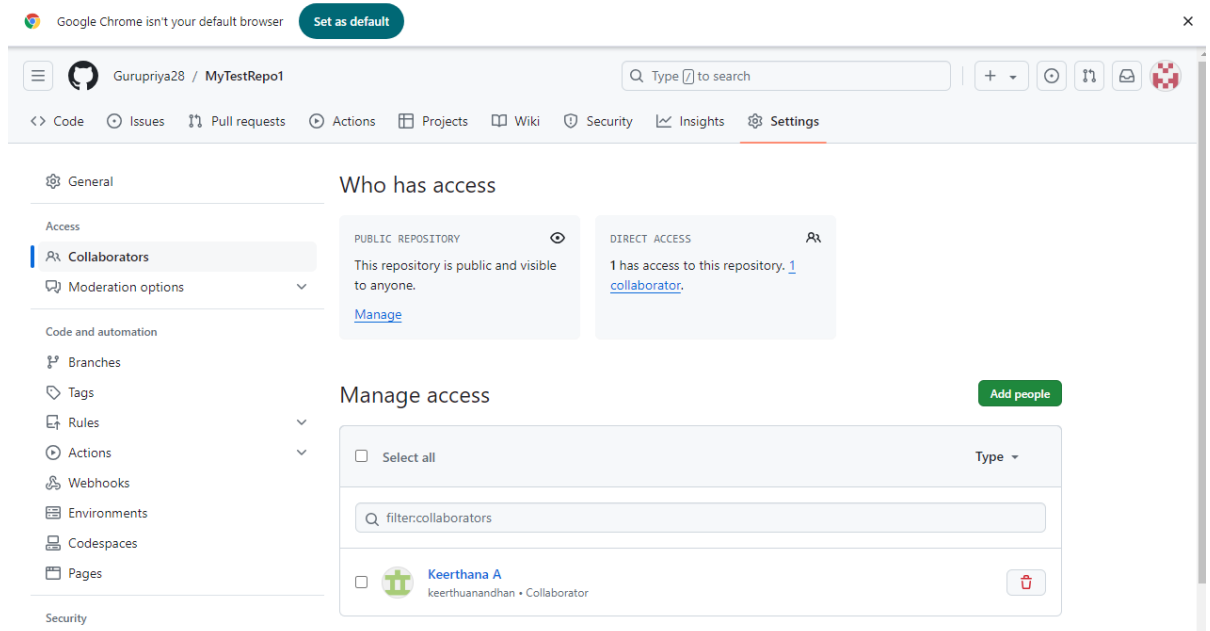
1. First, one person in the group should create a public repository using their GitHub account.



2. This same person should then follow the instructions from GitHub to add a remote, and then push their repository. Do not forget the `-u` flag, as suggested by GitHub!

```
C:\Git>git add *.txt
C:\Git>git commit -m "Text File commit"
[master (root-commit) 6b1c07c] Text File commit
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 File.txt
create mode 100644 File1.txt
C:\Git>git remote add origin https://github.com/Gurupriya28/MyTestRepo1.git
error: remote origin already exists.
C:\Git>git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 224 bytes | 224.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Gurupriya28/MyTestRepo1.git
 * [new branch] master -> master
branch 'master' set up to track 'origin/master'.
```

3. All of the other members of the group should then be added as collaborators, so they can commit to the repository also.



4. Next, everyone else in the group should clone the repository from GitHub. Verify that the context of the repository is what is expected.

```
D:\>cd TestRepo
D:\TestRepo>git clone https://github.com/Gurupriya28/MyTestRepo1
Cloning into 'MyTestRepo1'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
```

5. One of the group members who just cloned should now make a local commit, then push it. Everyone should verify that when they pull, that commit is added to their local repository (use git log to check for it).

```
D:\TestRepo>git add file2.txt
D:\TestRepo>git commit -m "file commit"
[master (root-commit) d23a282] file commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 TestRepo/file2.txt
D:\TestRepo>git pull
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 204 bytes | 25.00 KiB/s, done.
From https://github.com/Gurupriya28/MyTestRepo1
* [new branch] master -> origin/master
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.
```

6. Look at each other's git log output. Notice how the SHA-1 is the same for a given commit across every copy of the repository. Why is this important?

```
D:\TestRepo>git log
commit d23a28245afc46fbf83a1e84c56f54c32d8e7ef4 (HEAD -> master)
Author: keerthana <keerthana.a2019@gmail.com>
Date: Mon Jul 15 14:38:27 2024 +0530

    file commit
```

7. Two members of the group should now make a commit locally, and race to push it. To keep things simple, be sure to edit different files. What happens to the runner-up?

```
C:\Git>git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 224 bytes | 224.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Gurupriya28/MyTestRepo1.git
* [new branch] master -> master
branch 'master' set up to track 'origin/master'.
C:\Git>git log
commit 6b1c0c601bfc35ba758bdab35646dc0f99ce5f7 (HEAD -> master, origin/master)
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Mon Jul 15 14:01:31 2024 +0530

    Text File commit
```

8. The runner-up should now pull. As a group, look at the output of the command. Additionally, look at the git log, and notice that there is a merge commit. You may also wish to view the DAG in gitk.

```
D:\TestRepo>git pull
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 204 bytes | 25.00 KiB/s, done.
From https://github.com/Gurupriya28/MyTestRepo1
* [new branch] master -> origin/master
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

    git pull <remote> <branch>

If you wish to set tracking information for this branch you can do so with:

    git branch --set-upstream-to=origin/<branch> master
D:\TestRepo>
```

9. Repeat the last two steps a couple of times, to practice.

```
nothing to commit, working tree clean
D:\Git2>git add file12.txt
D:\Git2>git commit -m "file12"
[branch-2 ad4e267] file12
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 file12.txt
```

```
D:\Git2>git add file13.txt
D:\Git2>git commit -m "file13"
[master 4ea2282] file13
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 file13.txt
```

Exercise 4

1. Make a commit, and make a silly typo in the commit message.

```
C:\Users\homs>c:
C:\Users\homs>cd ..
C:\Users>cd ..
C:\>cd Git
C:\Git>git add File5.txt
C:\Git>git commit -m "commitmn"
[master e07e98a] commitmn
```

2. Use the --amend flag to enable you to fix the commit message.

```
C:\Git>git commit --amend -m "Commit 5"
[master ad0d45b] Commit 5
Date: Mon Jul 15 15:03:29 2024 +0530
```

3. Look at the log and notice how the mistake is magically gone.

```
C:\Git>git log
commit ad0d45b2a6bc329f16506f70ce121fe8db9f55f (HEAD -> master)
Merge: a4603c4 b937ed3
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Mon Jul 15 15:03:29 2024 +0530

    Commit 5

commit b937ed37e5ef89f81aa2ce67653cd4a1009fefb8 (origin/master)
Author: keerthana <keerthana.a2019@gmail.com>
Date: Mon Jul 15 14:59:36 2024 +0530

    commitmsg

commit a4603c4e1a0cc3c6f83f3ba91af873ce2ac885ae
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Mon Jul 15 14:55:48 2024 +0530

    File 3 commit

commit 6b1c07c601bfc35ba758bdab35646dc0f99ce5f7
Author: Gurupriya <gurupriya2728@gmail.com>
Date: Mon Jul 15 14:01:31 2024 +0530

    Text File commit
```

4. Now make a commit where you make a typo in one of the files. Once again, use --amend to magic away your problems.

```
C:\Git>git add File6.txt
C:\Git>git commit -m "File 6 cmmitt"
[master adb4583] File 6 cmmitt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 File6.txt

C:\Git>git commit --amend -m "File 6 commit"
[master 4ce7050] File 6 commit
Date: Mon Jul 15 15:08:34 2024 +0530
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 File6.txt
```

5. Create a branch. Make a commit.

```
C:\Git>git branch Branch-1
C:\Git>git checkout Branch-1
Switched to branch 'Branch-1'
C:\Git>git add File7.txt
C:\Git>git commit -m "File 7 commit"
[Branch-1 92777dd] File 7 commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 File7.txt
```

6. Now switch back to your master branch. Make a (non-conflicting) commit there also.

```
C:\Git>git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 3 commits.
(use "git push" to publish your local commits)

C:\Git>git add File7.txt
fatal: pathspec 'File7.txt' did not match any files

C:\Git>git add File7.txt
C:\Git>git commit -m "File7"
[master e2263c4] File7
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 File7.txt
```

7. Now switch back to your branch.

```
C:\Git>git checkout Branch-1
Switched to branch 'Branch-1'
```

8. Use the rebase command in your branch. Look at the DAG in gitk, and note that you have the commit from the master branch, but no merge commit.

```
C:\Git>git rebase master
warning: skipped previously applied commit 92777dd
hint: use --reapply-cherry-picks to include skipped commits
hint: Disable this message with "git config advice.skippedCherryPicks false"
Successfully rebased and updated refs/heads/Branch-1.
```

9. Make one more commit in your branch.

```
C:\Git>git add File8.txt
C:\Git>git commit -m "File 8 commit"
[Branch-1 ded10af] File 8 commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 File8.txt
```

10. Return to master. Merge your branch. Notice how, thanks to the rebase, this is a fastforward merge.

```
C:\Git>git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 4 commits.
(use 'git push' to publish your local commits)

C:\Git>git merge Branch-1
Updating e2263c4..ded10af
Fast-forward
 File8.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 File8.txt

C:\Git>
```