

Name: Khushi Kumawat

Roll no: 110

Batch: T23

Experiment No: 3

Aim:

To perform various git operations.

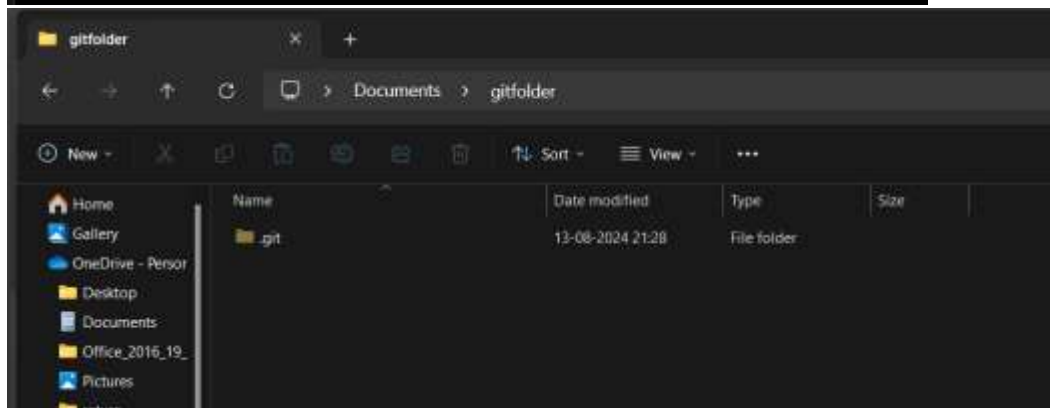
LO Mapping:

LO1,LO2

Commands:

1. Git initialization commands on local.
(Setup, Initializing, Staging, committing)

```
Admin@Harsh MINGW32 ~
$ cd Documents/gitfolder
Admin@Harsh MINGW32 ~/Documents/gitfolder
$ git init
Initialized empty Git repository in C:/Users/Admin/Documents/gitfolder/.git/
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ |
```

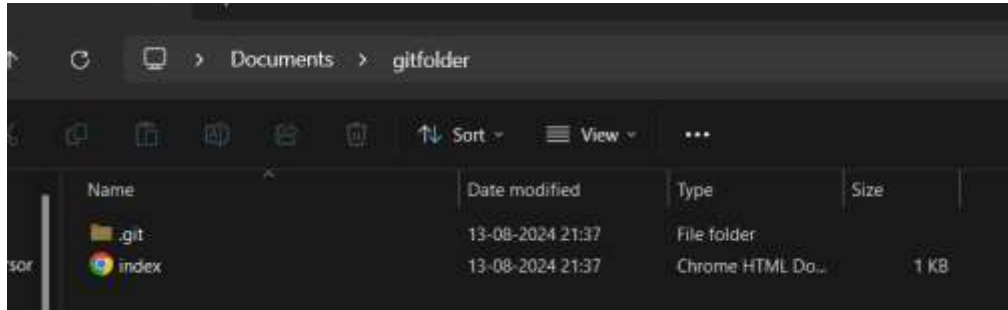


2. Git commands to add on repository
This will add the specified file(s) into the Git repository, the staging area, where they are already being tracked by Git and now ready to be committed.

Name: Khushi Kumawat

Roll no: 110

Batch: T23



```
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git add index.html

Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$
```

3. Commit command

This command records or snapshots files permanently in the version history. All the files, which are there in the directory right now, are being saved in the Git file system.

```
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git commit -m "index.html"
[master (root-commit) b4c9ebb] index.html
1 file changed, 9 insertions(+)
create mode 100644 index.html

Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git add index.html
git commit -m "Added a heading to index.html"
On branch master
nothing to commit, working tree clean

Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
```

4. Git status

This command will show the modified status of an existing file and the file addition status of a new file, if any, that has to be committed.

```
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git status
On branch master
nothing to commit, working tree clean
```

5. git reset

git reset is designed to undo local changes to the Staging Index and Working Directory.

```
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git reset index.html

Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git status
On branch master
nothing to commit, working tree clean
```

Name: Khushi Kumawat

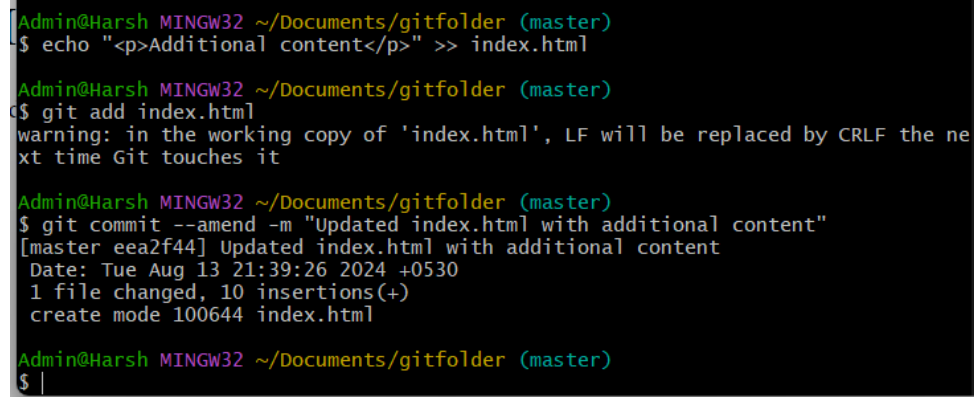
Roll no: 110

Batch: T23

6. git amend

commit --amend is used to modify the most recent commit. It combines changes in the staging environment with the latest commit, and creates a new commit.

This new commit replaces the latest commit entirely.

A terminal window showing the process of amending a commit. The user is in a directory ~/Documents/gitfolder on the master branch. They first add a new line to index.html using 'echo' and 'cat' (though the command shown is 'echo'). Then they run 'git add index.html', which shows a warning about LF vs CRLF. Finally, they run 'git commit --amend -m "Updated index.html with additional content"', which successfully amends the previous commit with the new content.

```
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ echo "<p>Additional content</p>" >> index.html

Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git add index.html
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git commit --amend -m "Updated index.html with additional content"
[master eea2f44] Updated index.html with additional content
Date: Tue Aug 13 21:39:26 2024 +0530
1 file changed, 10 insertions(+)
create mode 100644 index.html

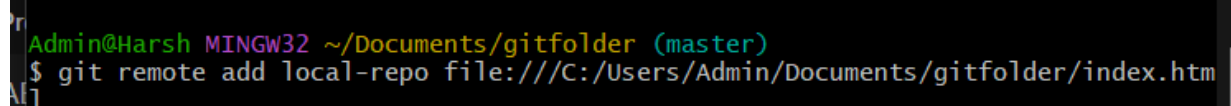
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$
```

Hello, World!

Additional content

7. git remote

Once everything is ready on our local system, we can start pushing our code to the remote (central) repository of the project.

A terminal window showing the command to add a local repository as a remote. The user is in the same directory and branch. They run 'git remote add local-repo file:///C:/Users/Admin/Documents/gitfolder/index.html'.

```
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git remote add local-repo file:///C:/Users/Admin/Documents/gitfolder/index.html
```

8. git branch

When multiple developers are collaborating on a project or repository, branches become essential for managing different workspaces. Using this command, we can create a new branch (for example, 'branch1'). This allows developers to work independently on their respective branches, making changes and commits without affecting the main branch or other branches.

Name: Khushi Kumawat

Roll no: 110

Batch: T23

```
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git branch
css
* master
```

9. Create a new branch

This command allows us to switch to an existing branch within our repository. It facilitates navigating to the desired branch, enabling us to add new files, make changes, and commit those files within that specific branch.

```
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git branch css
```

10. switching to a branch

This command allows us to switch to an existing branch within our repository. It facilitates navigating to the desired branch, enabling us to add new files, make changes, and commit those files within that specific branch.

```
Admin@Harsh MINGW32 ~/Documents/gitfolder (master)
$ git checkout css
Switched to branch 'css'
```

11. git merge This command will combine multiple sequences of commits into one unified history. In the most frequent use cases, git merge is used to combine two branches. The git merge command takes two commit pointers, usually the branch tips, and finds a common base commit between them. Once it finds a common base commit, it will create a commit sequence.

```
Admin@Harsh MINGW32 ~/Documents/gitfolder (feature/css)
$ git merge feature/css
Already up to date.
```

Conclusion:

In conclusion, this experiment covers essential Git operations that form the foundation of version control in software development. By learning commands for initializing a repository, staging changes, committing, branching, and merging, you gain the skills necessary to manage and collaborate on code efficiently. These commands help in maintaining a structured workflow, enabling smooth project development and version management.