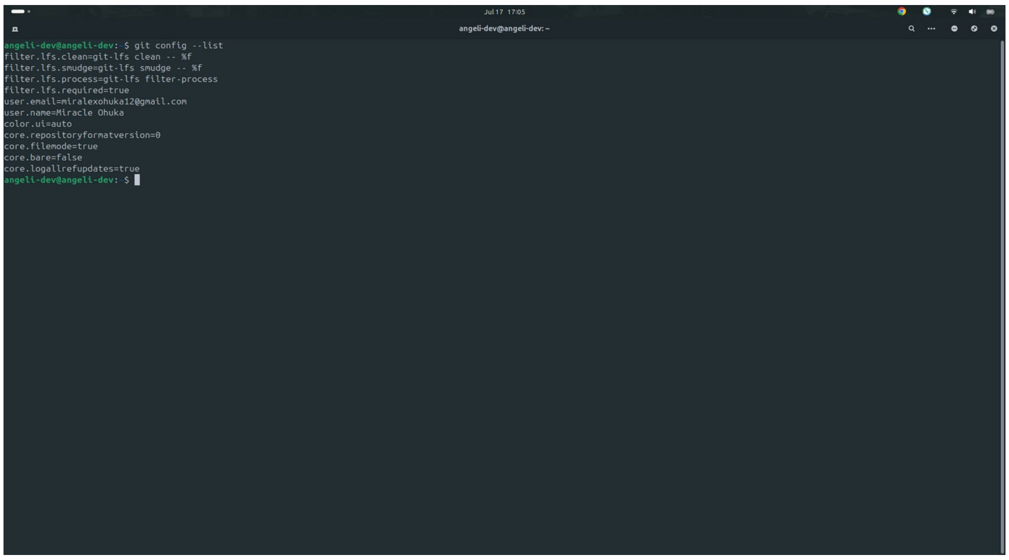
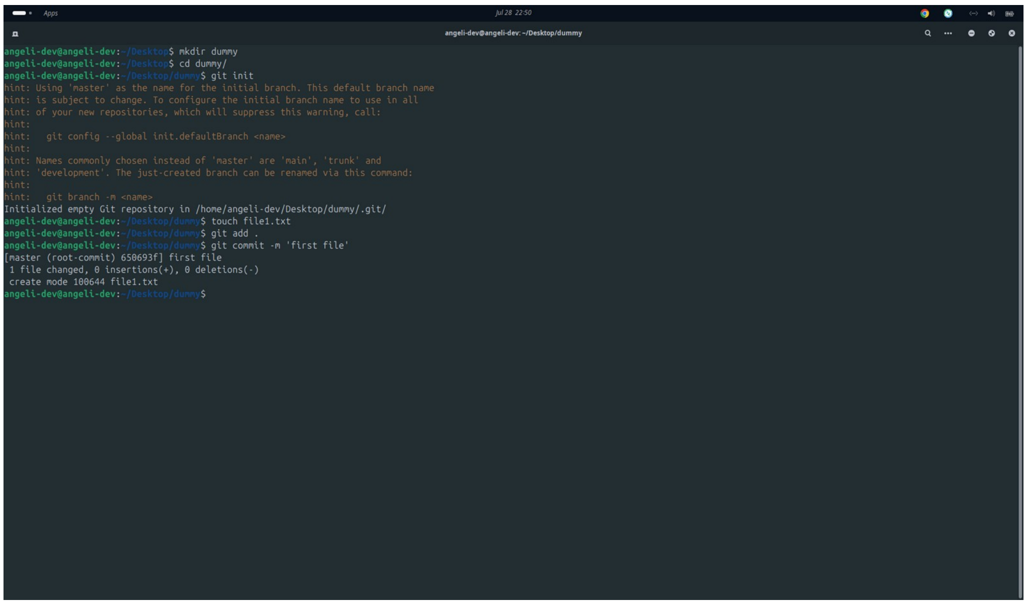
1. Set the global configuration file with your username and email. List all the properties which you just set.



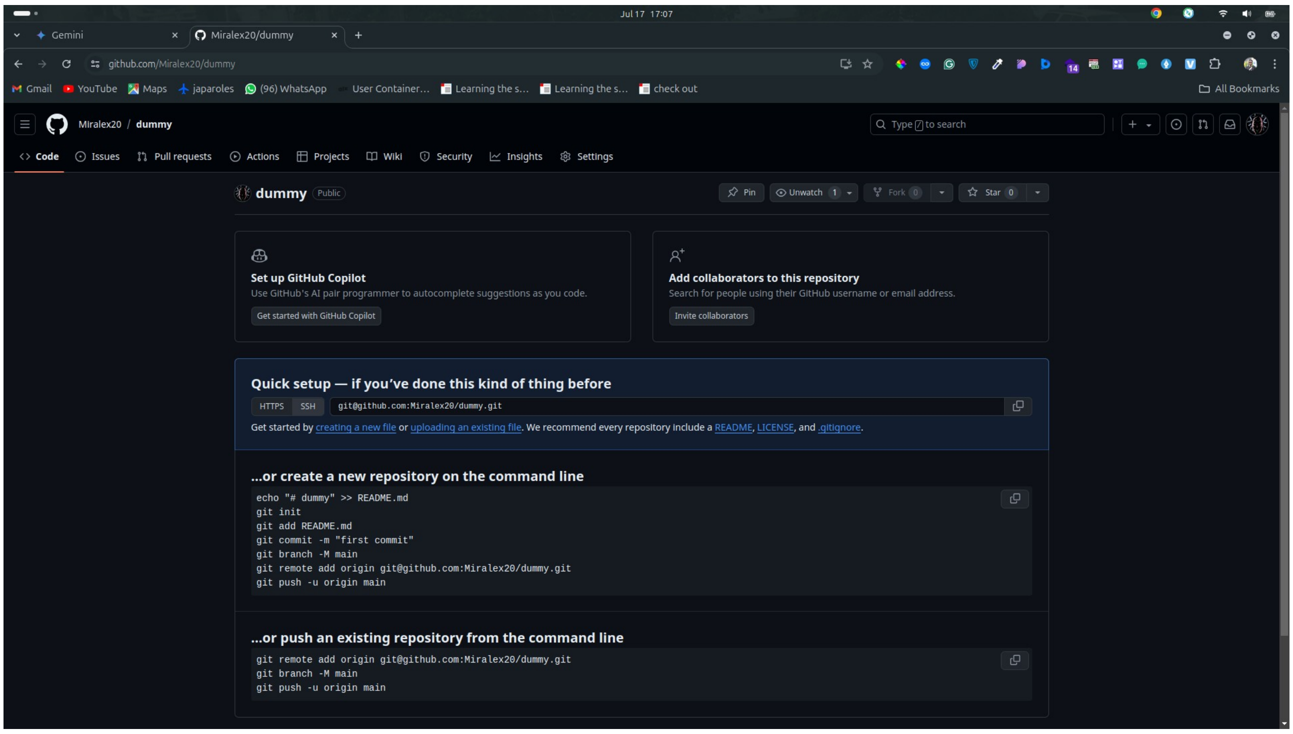
```
angeli-dev@angeli-dev:~$ git config --list
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
user.email=miralexohuka12@gmail.com
user.name=Miracle Ohuka
color.ui=auto
core.repositoryformatversion=0
core.filemode=true
core.bare=false
core.logallrefupdates=true
angeli-dev@angeli-dev:~$
```

2.. Create a fresh Git project. Add a file into the project. Commit the changes to the local repository.



```
angeli-dev@angeli-dev:~/Desktop$ mkdir dummy
angeli-dev@angeli-dev:~/Desktop$ cd dummy/
angeli-dev@angeli-dev:~/Desktop/dummy$ git init
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 /home/angeli-dev/Desktop/dummy/.git/
angeli-dev@angeli-dev:~/Desktop/dummy$ touch file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ git add .
angeli-dev@angeli-dev:~/Desktop/dummy$ git commit -m 'first file'
[master (root-commit) 650693f] first file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$
```

3. Create a GitHub/Gitlab account (or use the account if already registered). Clone a project from the remote repository to your local repository



Set up GitHub Copilot
Use GitHub's AI pair programmer to autocomplete suggestions as you code.
Get started with GitHub Copilot

Add collaborators to this repository
Search for people using their GitHub username or email address.
Invite collaborators

Quick setup — if you've done this kind of thing before
HTTPS SSH `git@github.com:Miralex20/dummy.git`
Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.

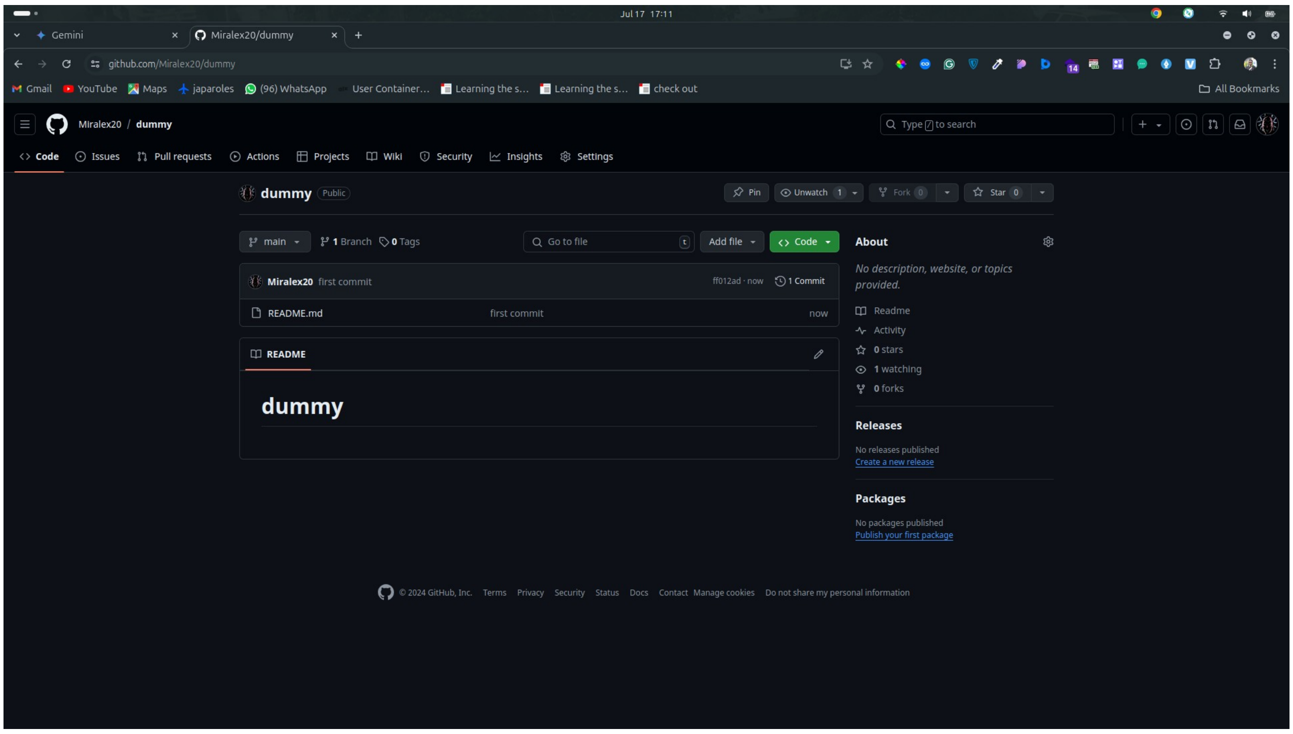
...or create a new repository on the command line

```
echo "# dummy" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:Miralex20/dummy.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin git@github.com:Miralex20/dummy.git
git branch -M main
git push -u origin main
```

4. Push the project created in assignment 2 to the remote repository.



Miralex20 first commit ff012ad · now 1 Commit
README.md first commit now

README

dummy

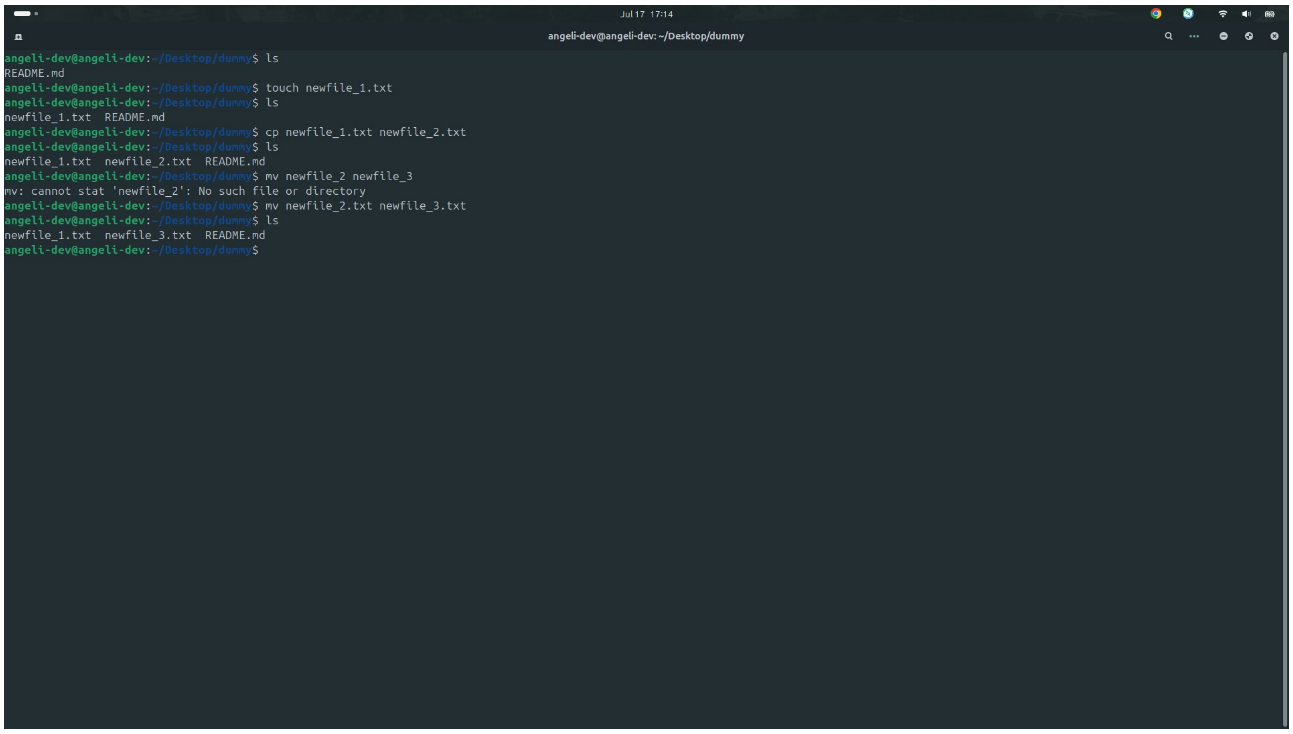
About
No description, website, or topics provided.
Readme
Activity
0 stars
1 watching
0 forks

Releases
No releases published
Create a new release

Packages
No packages published
Publish your first package

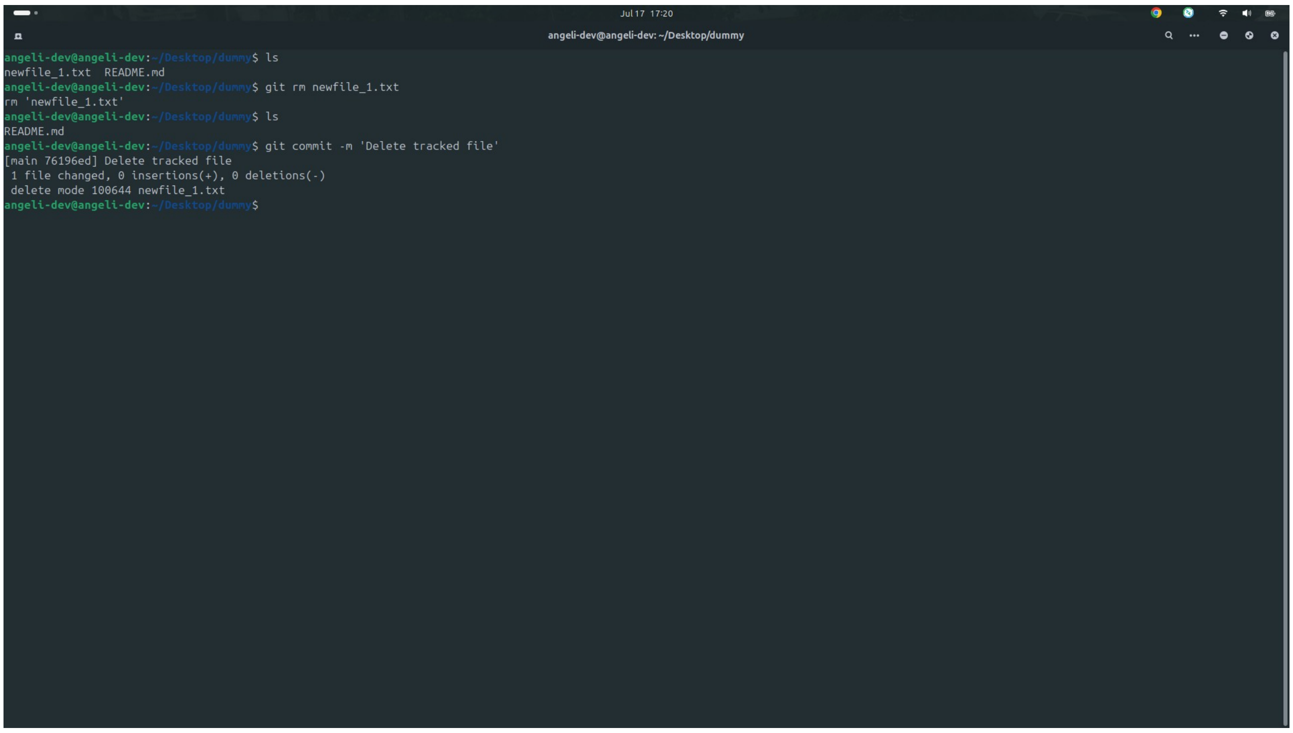
© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information

5. Try out all the different ways of renaming and moving files. Understand the differences between different options.



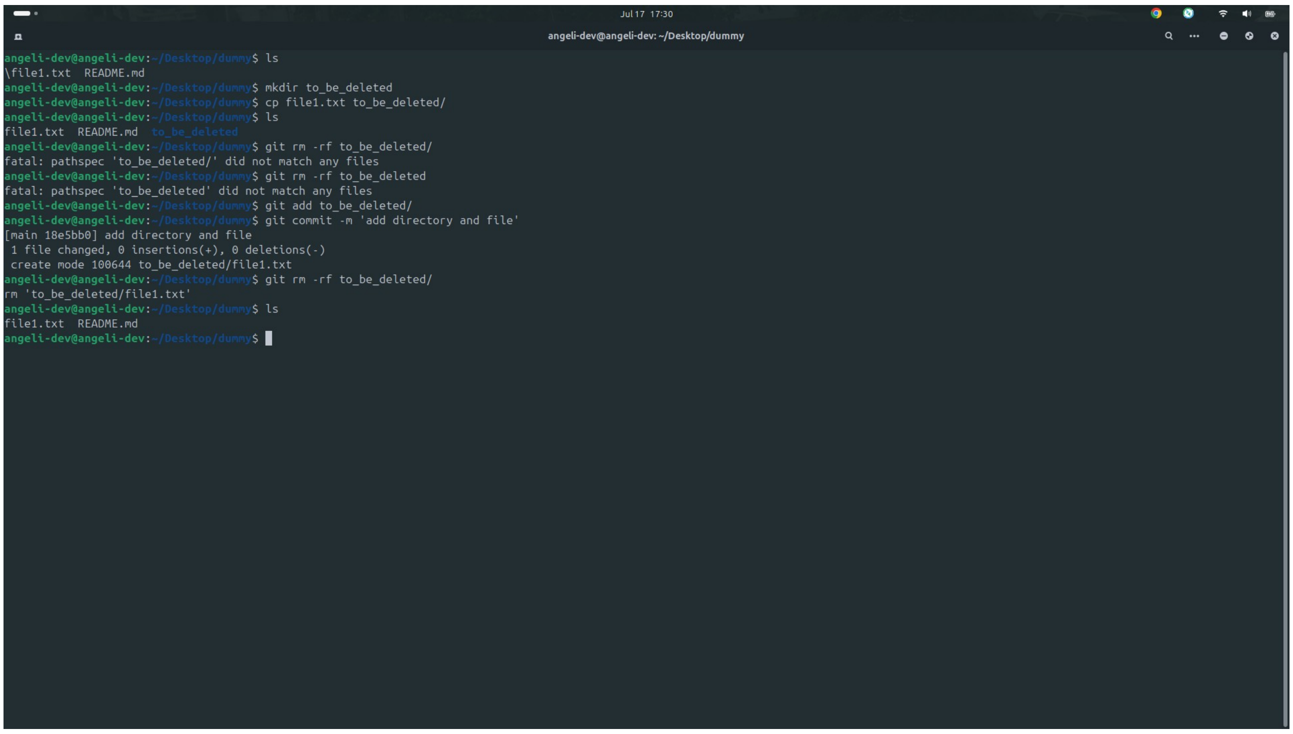
```
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ touch newfile_1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
newfile_1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ cp newfile_1.txt newfile_2.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
newfile_1.txt  newfile_2.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ mv newfile_2 newfile_3
mv: cannot stat 'newfile_2': No such file or directory
angeli-dev@angeli-dev:~/Desktop/dummy$ mv newfile_2.txt newfile_3.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
newfile_1.txt  newfile_3.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$
```

6. You just created a new file in your Git project, but then you decided that the file is to be removed. How do you delete this untracked file.



```
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
newfile_1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ git rm newfile_1.txt
rm 'newfile_1.txt'
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ git commit -m 'Delete tracked file'
[main 76196ed] Delete tracked file
 1 file changed, 0 insertions(+), 0 deletions(-)
 delete mode 100644 newfile_1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$
```

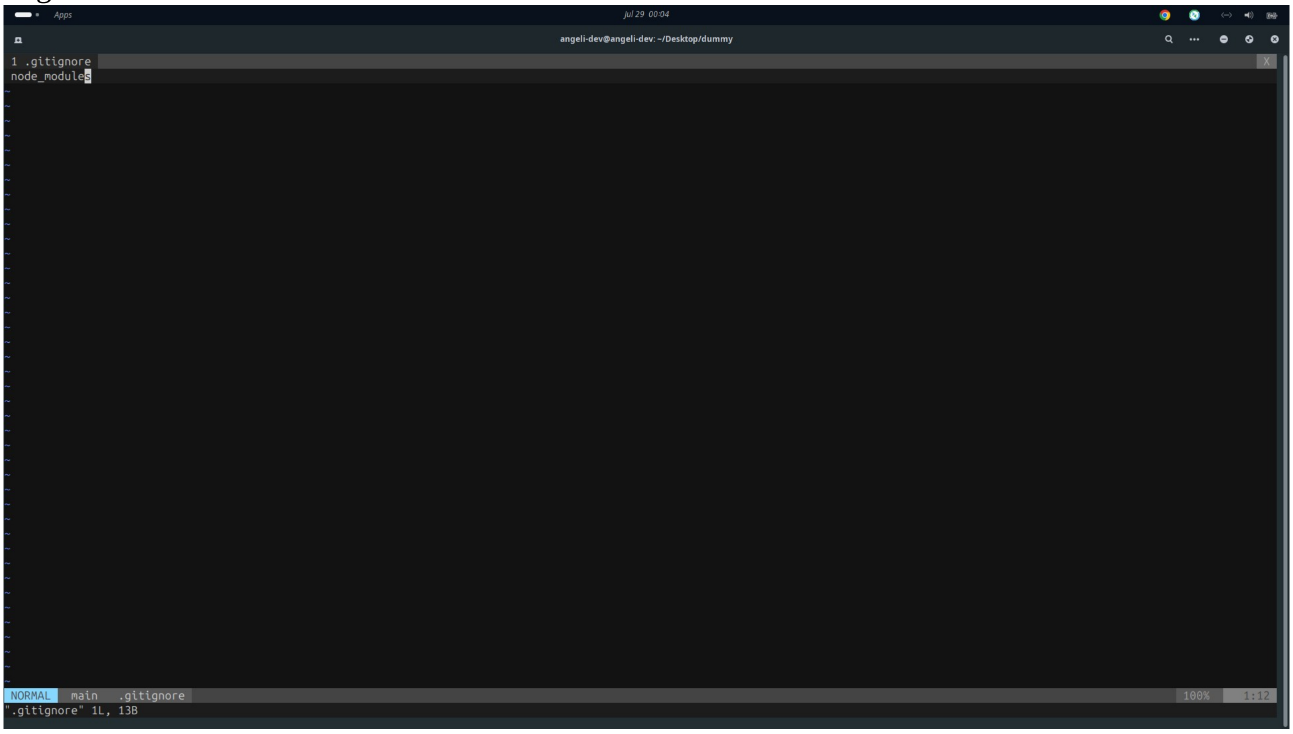
7. Demonstrate the following: a. delete of a tracked file b. backing out staged deletion c. recursive deletion



```
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
\file1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ mkdir to_be_deleted
angeli-dev@angeli-dev:~/Desktop/dummy$ cp file1.txt to_be_deleted/
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
file1.txt  README.md  to_be_deleted
angeli-dev@angeli-dev:~/Desktop/dummy$ git rm -rf to_be_deleted/
fatal: pathspec 'to_be_deleted/' did not match any files
angeli-dev@angeli-dev:~/Desktop/dummy$ git rm -rf to_be_deleted
fatal: pathspec 'to_be_deleted' did not match any files
angeli-dev@angeli-dev:~/Desktop/dummy$ git add to_be_deleted/
angeli-dev@angeli-dev:~/Desktop/dummy$ git commit -m 'add directory and file'
[main 18e5bb0] add directory and file
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 to_be_deleted/file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ git rm -rf to_be_deleted/
rm 'to_be_deleted/file1.txt'
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
file1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$
```

8. You have lot of changes in your Git project but you do not want to push certain folders/files of your project. How do you manage this?

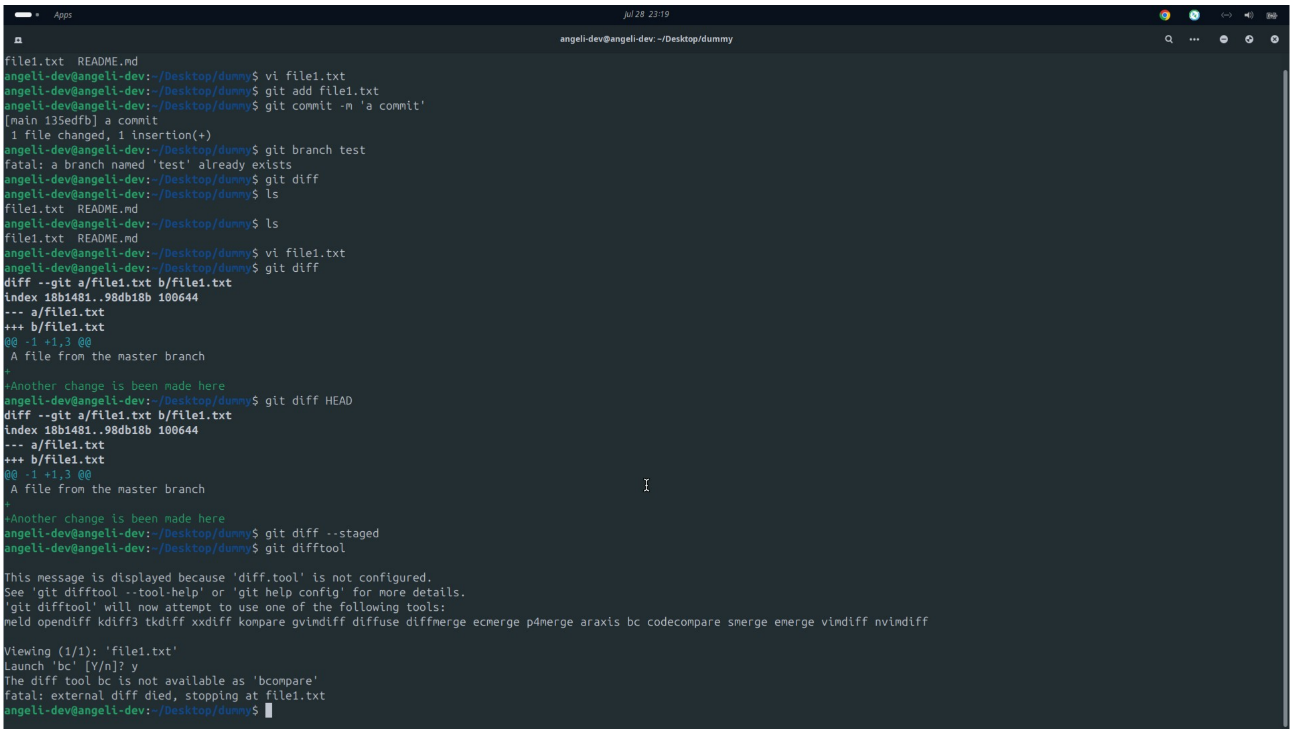
Answer.
Create a .gitignore file and add the name/ regular expression matching every file/folder you wish to ignore.



```
1 .gitignore
node_modules
~
NORMAL  main  .gitignore                                  100%  1:12
".gitignore" 1L, 13B
```

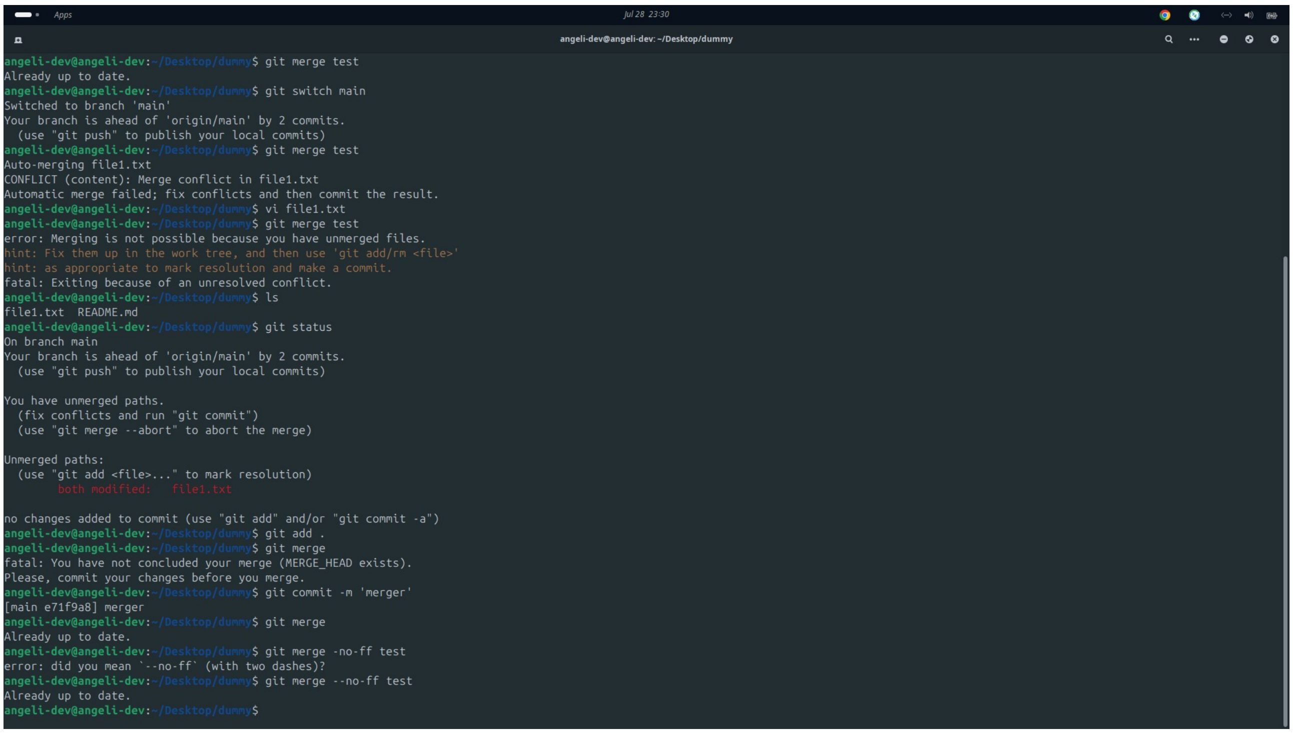
9. Create a branch called "test". Make some changes in the master branch. Let there be some changes in the working directory and some in the staging area. Make some changes in the test branch as well. Issue the command to show the differences for a. Working directory vs Staging area

b. Working directory vs Local Repository c. Staging area vs Local Repository d. Between two commits e. Between two tags f. Local vs Remote Repository g. Master branch vs test branch



```
file1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ vi file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ git add file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ git commit -m 'a commit'
[main 135edfb] a commit
 1 file changed, 1 insertion(+)
angeli-dev@angeli-dev:~/Desktop/dummy$ git branch test
fatal: a branch named 'test' already exists
angeli-dev@angeli-dev:~/Desktop/dummy$ git diff
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
file1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
file1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ vi file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ git diff
diff --git a/file1.txt b/file1.txt
index 18b1481..98db18b 100644
--- a/file1.txt
+++ b/file1.txt
@@ -1 +1,3 @@
 A file from the master branch
+
+Another change is been made here
angeli-dev@angeli-dev:~/Desktop/dummy$ git diff HEAD
diff --git a/file1.txt b/file1.txt
index 18b1481..98db18b 100644
--- a/file1.txt
+++ b/file1.txt
@@ -1 +1,3 @@
 A file from the master branch
+
+Another change is been made here
angeli-dev@angeli-dev:~/Desktop/dummy$ git diff --staged
angeli-dev@angeli-dev:~/Desktop/dummy$ git difftool

This message is displayed because 'diff.tool' is not configured.
See 'git difftool --tool-help' or 'git help config' for more details.
'git difftool' will now attempt to use one of the following tools:
meld opendiff kdiff3 tkdiff xxdiff kompare gvimdiff diffuse diffmerge ecmerge p4merge araxis bc codecompare smerge emerge vimdiff nvimdiff

Viewing (1/1): 'file1.txt'
Launch 'bc' [Y/n]? y
The diff tool bc is not available as 'bcompare'
fatal: external diff died, stopping at file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$
```

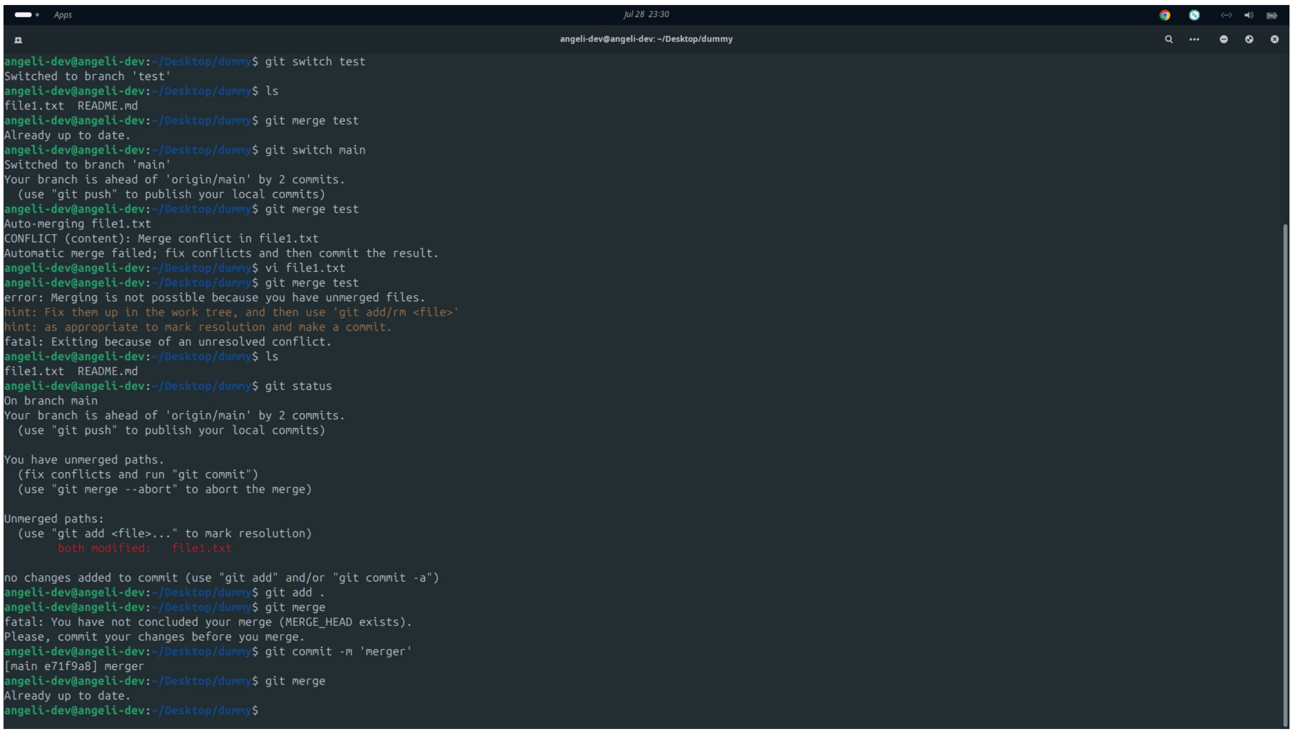
10. Merge the changes from test branch to master branch. a. FastForward merge b. Disabling FastForward merge c. What is the difference between option (a) and option (b)



```
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge test
Already up to date.
angeli-dev@angeli-dev:~/Desktop/dummy$ git switch main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge test
Auto-merging file1.txt
CONFLICT (content): Merge conflict in file1.txt
Automatic merge failed; fix conflicts and then commit the result.
angeli-dev@angeli-dev:~/Desktop/dummy$ vi file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge test
error: Merging is not possible because you have unmerged files.
hint: Fix them up in the work tree, and then use 'git add/rm <file>'
hint: as appropriate to mark resolution and make a commit.
fatal: Exiting because of an unresolved conflict.
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
file1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   file1.txt

no changes added to commit (use "git add" and/or "git commit -a")
angeli-dev@angeli-dev:~/Desktop/dummy$ git add .
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge
fatal: You have not concluded your merge (MERGE_HEAD exists).
Please, commit your changes before you merge.
angeli-dev@angeli-dev:~/Desktop/dummy$ git commit -m 'merger'
[main e71f9a8] merger
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge
Already up to date.
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge -no-ff test
error: did you mean `--no-ff` (with two dashes)?
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge --no-ff test
Already up to date.
angeli-dev@angeli-dev:~/Desktop/dummy$
```

c. Fast forward merge: Git attempts to combine changes directly into the target branch without creating a new commit.
No fast forward merge: Forces Git to create a new merge commit, even if a fast-forward is possible.

11. Create a merge conflict situation. Resolve the conflict and merge the changes between the branches.



```
angeli-dev@angeli-dev:~/Desktop/dummy$ git switch test
Switched to branch 'test'
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
file1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge test
Already up to date.
angeli-dev@angeli-dev:~/Desktop/dummy$ git switch main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge test
Auto-merging file1.txt
CONFLICT (content): Merge conflict in file1.txt
Automatic merge failed; fix conflicts and then commit the result.
angeli-dev@angeli-dev:~/Desktop/dummy$ vi file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge test
error: Merging is not possible because you have unmerged files.
hint: Fix them up in the work tree, and then use 'git add/rm <file>'
hint: as appropriate to mark resolution and make a commit.
fatal: Exiting because of an unresolved conflict.
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
file1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   file1.txt

no changes added to commit (use "git add" and/or "git commit -a")
angeli-dev@angeli-dev:~/Desktop/dummy$ git add .
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge
fatal: You have not concluded your merge (MERGE_HEAD exists).
Please, commit your changes before you merge.
angeli-dev@angeli-dev:~/Desktop/dummy$ git commit -m 'merger'
[main e71f9a8] merger
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge
Already up to date.
angeli-dev@angeli-dev:~/Desktop/dummy$
```

12. What is the difference between merge and rebase, demonstrate with an example. Explain it

Merge: In this method git creates a new commit to represent the combination of two branches, it combines changes from one branch to another.

Rebase: In this method git moves a branch’s commits to a new base, it rewrites the project’s history by creating new commits based on the original one.

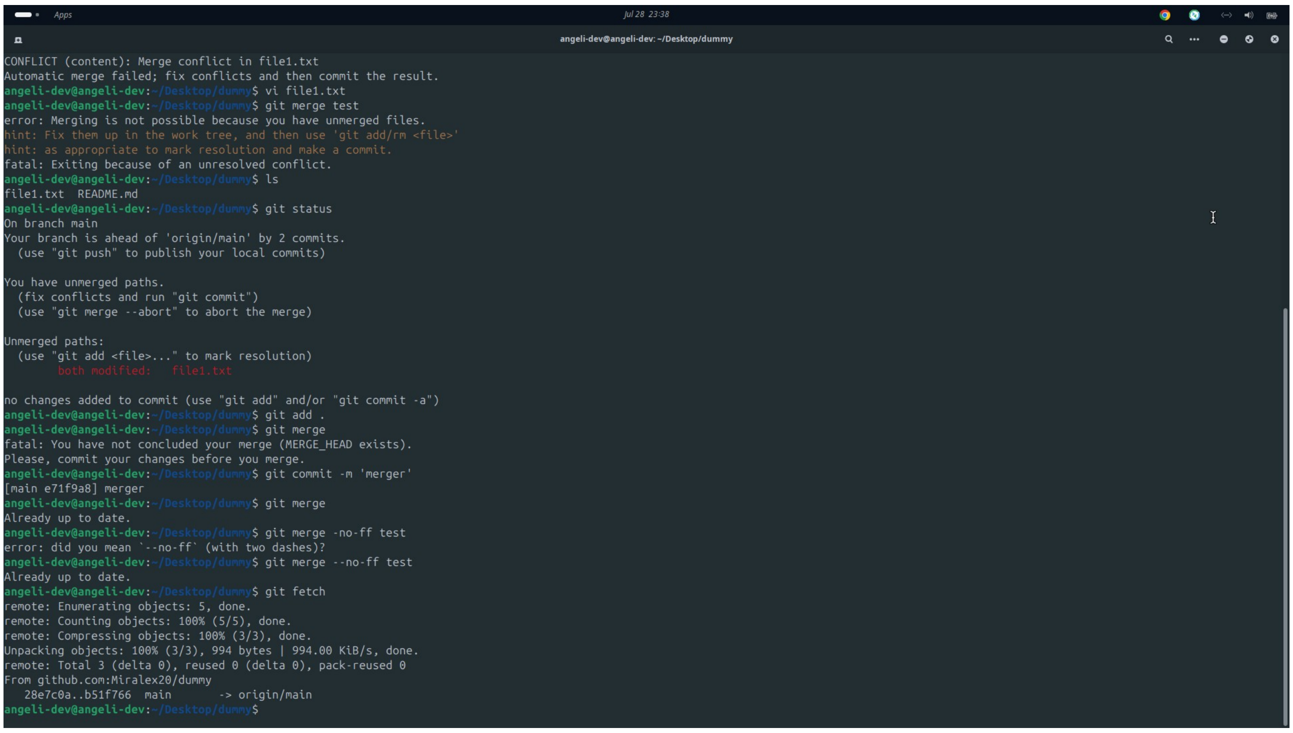
13. With an example, demonstrate fetch, clone and pull. What is the use case for these operations. Are they same or different? Explain.

Git clone:
 Creates a local copy of a remote repository, downloads all files and the complete commit history from the remote repository to your local machine.\

Git fetch:
Updates local information about remote branches, downloads new commits and branches from the remote repository without merging them into your local branches.

Git pull:
Updates your local repository with changes from the remote repository, combines the other mentioned two, It downloads new commits and branches from the remote repository and then merges them into your current branch.

14. Create a new repository in Github/Gitlab, with a README file. While pushing to the remote repository, if the remote branch is ahead of the local repository (new file is added in remote repository, which is not there in local repository) and pull is failing, how do you solve this problem?



```
CONFLICT (content): Merge conflict in file1.txt
Automatic merge failed; fix conflicts and then commit the result.
angeli-dev@angeli-dev:~/Desktop/dummy$ vi file1.txt
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge test
error: Merging is not possible because you have unmerged files.
hint: Fix them up in the work tree, and then use 'git add/rm <file>'
hint: as appropriate to mark resolution and make a commit.
fatal: Exiting because of an unresolved conflict.
angeli-dev@angeli-dev:~/Desktop/dummy$ ls
file1.txt  README.md
angeli-dev@angeli-dev:~/Desktop/dummy$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
  (use "git push" to publish your local commits)

You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   file1.txt

no changes added to commit (use "git add" and/or "git commit -a")
angeli-dev@angeli-dev:~/Desktop/dummy$ git add .
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge
fatal: You have not concluded your merge (MERGE_HEAD exists).
Please, commit your changes before you merge.
angeli-dev@angeli-dev:~/Desktop/dummy$ git commit -m 'merger'
[main e71f9a8] merger
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge
Already up to date.
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge -no-ff test
error: did you mean `--no-ff` (with two dashes)?
angeli-dev@angeli-dev:~/Desktop/dummy$ git merge --no-ff test
Already up to date.
angeli-dev@angeli-dev:~/Desktop/dummy$ git fetch
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
Unpacking objects: 100% (3/3), 994 bytes | 994.00 KiB/s, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
From github.com:Miralex20/dummy
   28e7c0a..b51f766  main       -> origin/main
angeli-dev@angeli-dev:~/Desktop/dummy$
```