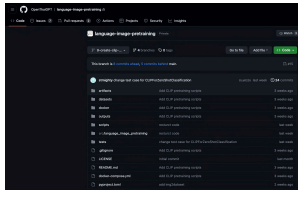
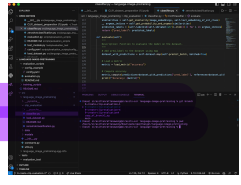
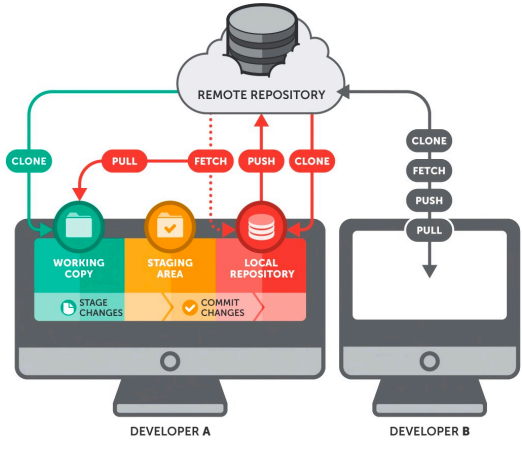
vocab

- git remote -> A remote in Git is a common repository that all team members use to exchange their changes.
- git local -> store data(and code) on your local machine







1. How to create new branch

git checkout -b <branchname>

(this will be only on your local not remote)

if you want to push it on your remote repository

git push -u origin <branchname>

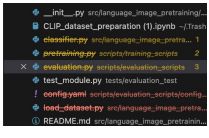
- you need to ensure that you are in branch main (generally we don't create branch in branch)

- this will take you to the branch you created (it will copy everything from main to the branch)

(you can ensure that you are in the branch by using git status, you will see the branch you are at the top)

- if you don't setup something when you go back to main branch the new code you write in branch will not be copied to branch main even on your local machine (but it is not lost your code is still the same when you access your branch. So, when you create the new branch make change on that branch not main to avoid confusion.

When you go back to main it will be sth like this if you make any change on git branch don't change anything on main to avoid mismatch until you want to merge branch and main



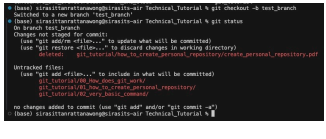
2. How to access branch you created.

git checkout <branchname>

git checkout main / git checkout master -> go back to main/master

3. See What Branch You're On

- Run this command:
  - **git status**



4. List All Branches

NOTE: The current local branch will be marked with an asterisk (*).

- To see **local branches**, run this command:
  - **git branch**
- To see **remote branches**, run this command:
  - **git branch -r**
- To see **all local and remote branches**, run this command:
  - **git branch -a**

5. When you want to push any branches to your github (remote)
you can

- git add
- git commit -m
- git push

} it will be put in only local not remote (if that branch isn't on remote yet)

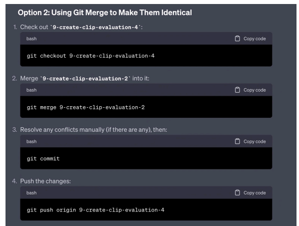
warning make sure that you are in the branch not main

remark use git push -u origin <branchname> to push the branch to remote-git

6.git merge -> Copy from a branch to another

firstly, go to main branch then

(Example of copy from 9-create-clip-evaluation-2 to 9-create-clip-evaluation-4)



Option 2: Using Git Merge to Make Them Identical

1. Check out `9-create-clip-evaluation-4`:

```bash
git checkout 9-create-clip-evaluation-4
```

2. Merge `9-create-clip-evaluation-2` into it:

```bash
git merge 9-create-clip-evaluation-2
```

3. Resolve any conflicts manually (if there are any), then:

```bash
git commit
```

4. Push the changes:

```bash
git push origin 9-create-clip-evaluation-4
```

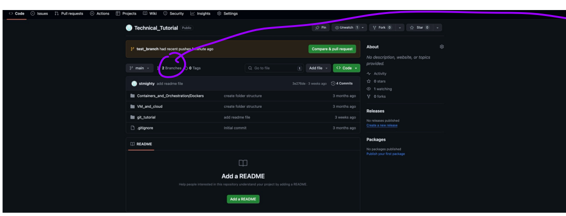
add valuation branch 2 to branch 4

[illegible] detail [illegible] 04

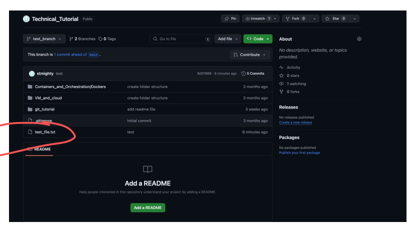
6.2 when you want to merge branch to main (pull request)

0. preparation

- create new branch
- test by adding a new file
- add commit push
- push -u origin test_branch main

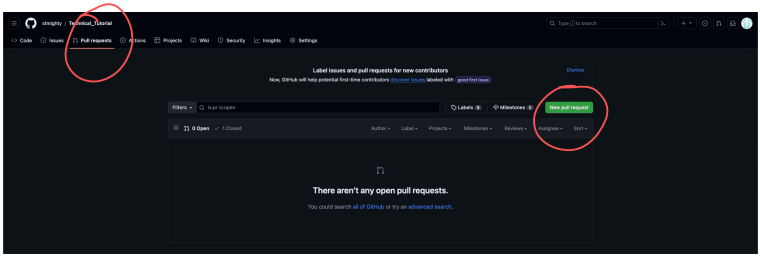


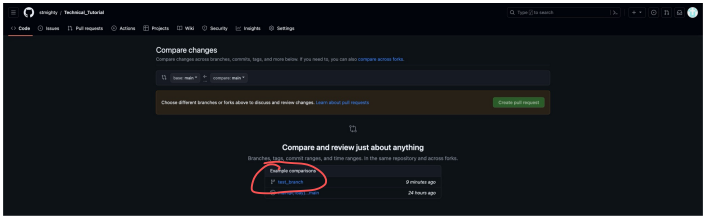
now in main branch

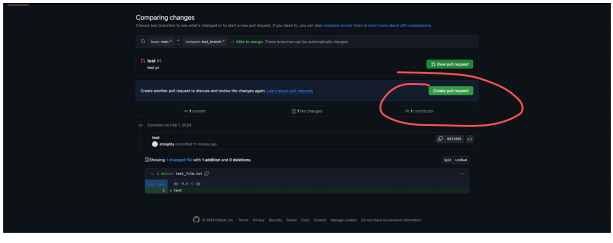


now in test_branch

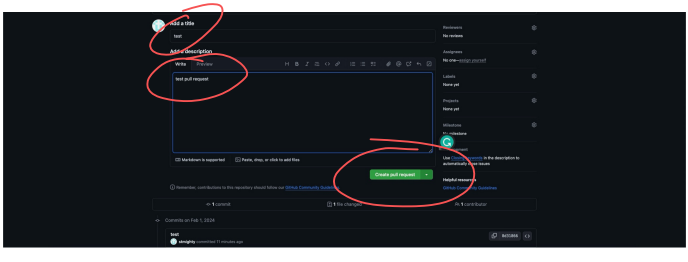
pull request

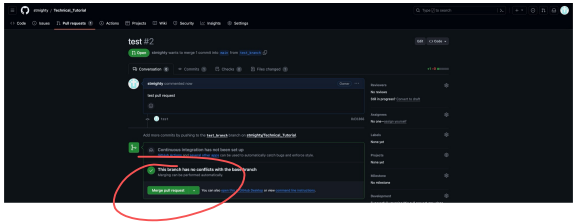


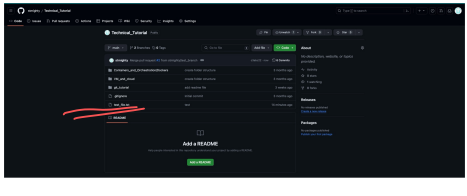




** แต่เวลาทำงานจริง เราแค่
Pr ไปให้คนอื่น เป็นคน merge (คนที่มี permission)
ถ้าเราชำนาญแล้วเรามี permission ก็ merge เองได้ หรือจะให้
คนอื่น review ก่อนก็ได้ โดยเราเลือกได้ว่าจะ pr ไปให้ใคร







7. Git pull : if you work as a team project you can pull the most recent code from remote repository using git pull command.

We should remember, a **git pull** is a combination of **git fetch** and a **git merge.**

There are different variations of 'git pull' which is discussed below:

- **`git pull origin master`** fetches commits from the master branch of the origin remote (into the local origin/master branch), and then it merges origin/master into the branch you currently have checked out.
- **`git pull`** only works if the branch you have checked out is tracking an upstream branch. For example, if the branch you have checked out tracks origin/master, `git pull` is equivalent to `git pull origin master`
- **`git pull origin/master`** is invalid unless you have a remote called "origin/master". However, `git merge origin/master` is valid. (It is the second step of `git pull origin master` as described above.)

ไม่ว่าอยู่ branch ไหน
ก็ pull จาก origin main ได้
(normal use case)

8. Git fetch -> git fetch origin

Fetch

```
$ git fetch origin
```

git fetch really only downloads new data from a remote repository - but it doesn't integrate any of this new data into your working files. Fetch is great for getting a fresh view on all the things that happened in a remote repository.
Due to it's "harmless" nature, you can rest assured: fetch will never manipulate, destroy, or screw up anything. This means you can never fetch often enough.

ChatGPT

Sure, let's go through an example to illustrate the difference between `git fetch` and `git pull`, and how they both retrieve updates from a remote repository.

Imagine you have a project repository on GitHub (remote repository) and a copy of this repository on your local machine (local repository). You're working on a branch called `main`. Another team member makes a change to the `main` branch in the remote repository by adding a new file called `newfile.txt` and commits this change.

Scenario with `git fetch`

1. You run `git fetch` in your local repository. This command contacts the remote repository and downloads any new data, including the commit that added `newfile.txt`.
2. After running `git fetch`, your local repository knows about `newfile.txt`, but it's not yet in your local working directory or in your local `main` branch. The commit is in a remote-tracking branch called `origin/main`.
3. To incorporate `newfile.txt` into your local `main` branch, you need to merge `origin/main` into `main`. You can do this with `git merge origin/main` or by checking out to `main` and then merging.

Scenario with `git pull`

1. You run `git pull` while on your local `main` branch. `git pull` does two things: it first runs `git fetch`, which downloads the new commit (with `newfile.txt`) from the remote repository, just like before.
2. Immediately after fetching, `git pull` performs a `git merge` of the fetched branch (`origin/main`) into your current branch (`main`). This means `newfile.txt` is automatically merged into your local `main` branch.
3. After `git pull` completes, your local `main` branch now contains `newfile.txt`. You didn't need to manually merge the changes as `git pull` did it for you.

Summary

- `git fetch`: Retrieves updates (like new commits) but does not automatically integrate them into your local working directory or branches. You see the updates, but you have to manually merge them.
- `git pull`: Retrieves updates and automatically integrates (merges) them into your current branch. It's a combination of `git fetch` followed by `git merge`.

In both cases, the updates from the remote repository are retrieved (downloaded), but `git pull` goes a step further by also merging these updates into your current working branch.

9. To undo git add and commit you can use

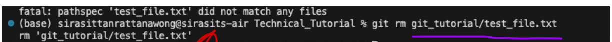
1. `git reset --soft HEAD^` (undo `git commit`)
2. `git reset` (undo `git add`)

อยากแค่ลบก็ได้ แล้ว add commit push
แบบ: ใช้ git rm ก็ได้

10. Working as a team, if you want to delete files from remote repository or you want to rename files or folders you can do it by

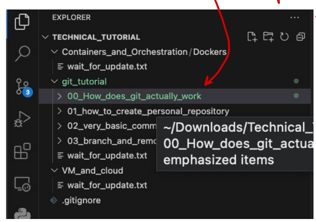
1. Git pull to your local * ต้อง... ก่อน
2. checkout new branch
3. remove or/and rename

rename ใน git มันก็จะ replace ชื่อไฟล์ใหม่ไปเลย ไม่ต้องลบไฟล์เก่า)



git rm <file>

remove



case example

- delete trash file
- delete unnecessary or broken file

method

1. cd to that directory
2. git rm

   for example git rm .DS_Store (this will also delete trash file on your local)

3. git add commit push

11. Conflict of git merge

12. การผูก issue