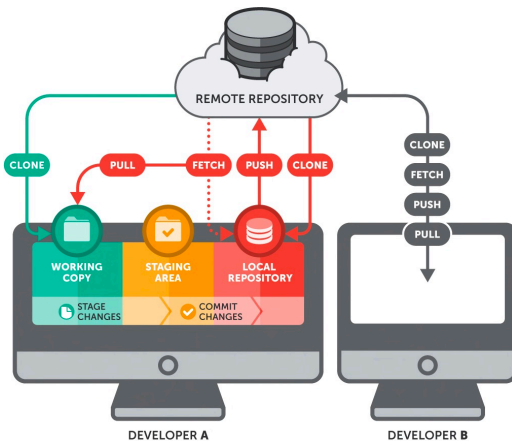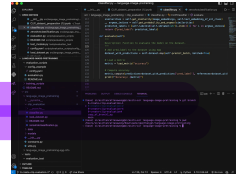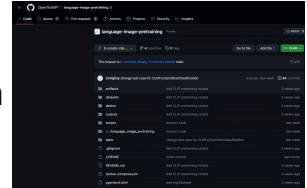vocab
- git remote -> A remote in Git is a common repository that all team members use to exchange their changes.
- git local -> store data(and code) on your local machine

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)
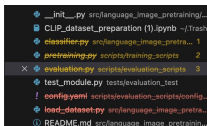   - this will take you to the branch you created (it will copy everything from main to the branch)
        (you can ensure that you are in the branch by using git status, you will see the branch you are at   the top)
        - if you don't setup something when you go back to main branch the new code you 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

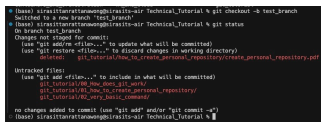2. How to access branch you created.

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   } it will be put in only local not remote
- git push   (if that branch isn't on remote yet)
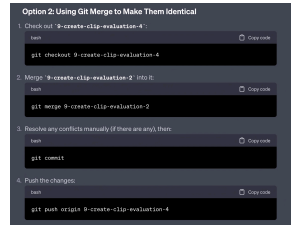
warning make sure that you are in the branch not main
remark use git push -u origin <branchname> to push the branch to remote-git

6.git merge -> Copy from a branch to another
firstly, go to main branch then
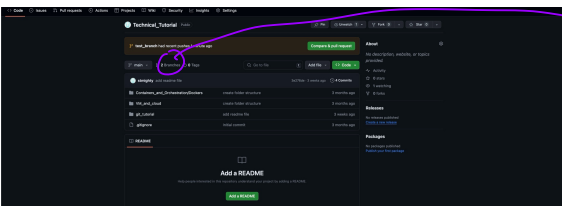(Example of copy from 9-create-clip-evaluation-2 to 9-create-clip-evaluation-4)

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
```

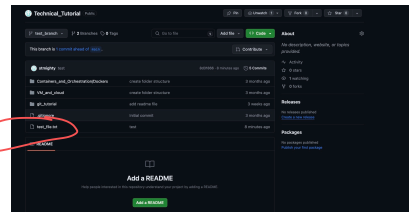add valaๅๅจน branch2 to branch4
1ู่ซั่ๅ1ัใ่ผ้ันๅ่ลุฃ ๅ detail 1ั่ oๅ

6.2 when you want to merge branch to main (pull request)
0. preparation
- create new branch
- test by adding a new file
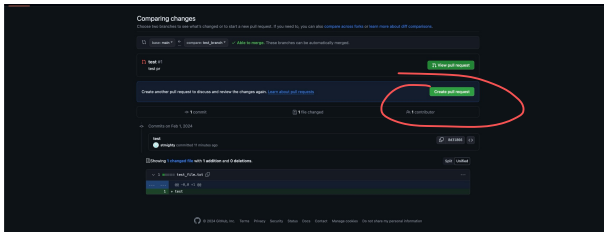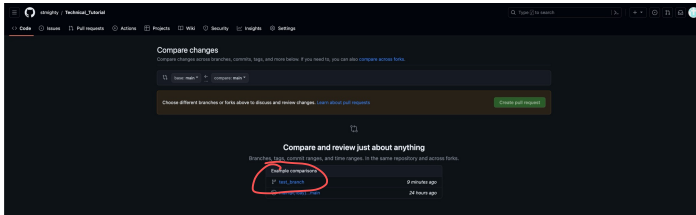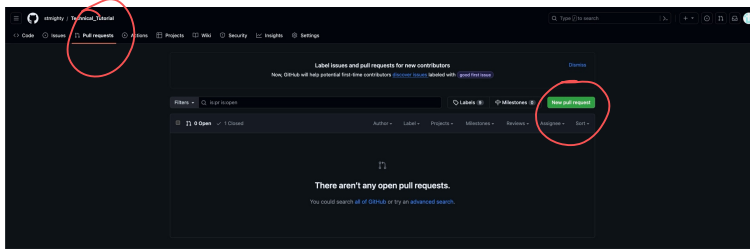- add commit push
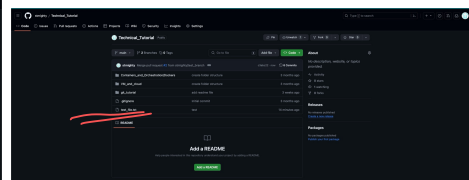- push -u origin test-branch main
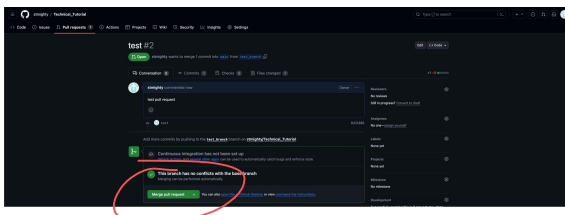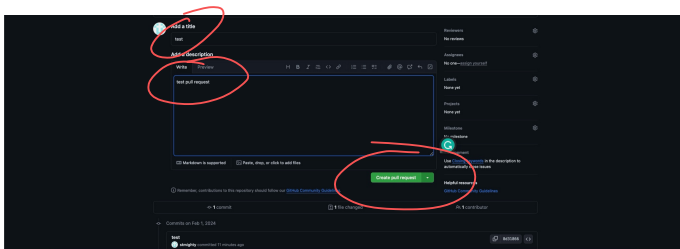

now in main branch

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.)

(handwritten note, orange box) เราใช้อยู่ 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.

9. To undo git add and commit you can use

git reset HEAD~1

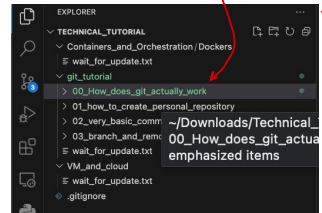วิธีๆ แต่ดูแล้วกว่าได้ เหลือ add commit push

แบบว่า ไง 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 *ดึงมา-อันนี้ก่อน*
2. checkout new branch
3. remove or/and rename

rename ถ้าๆ git อันนำเอา replace ชื่อใหม่ได้ไปเลย ไม่ต้อง อันใหม่(มา)

```
fatal: pathspec 'test_file.txt' did not match any files
(base) sirasittanrattanawong@sirasits-air Technical_Tutorial % git rm git_tutorial/test_file.txt
rm 'git_tutorial/test_file.txt'
```

git rm <file>

remove

case example
 - delete trash file
 - delete unnecessary or broken file

method
 1. cd to that directory
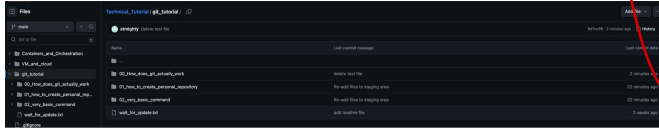 2. git rm
    for example   git rm . DS_Store (this will also delete trash file on your local)
 3. git add commit push

4.1 Git add commit push
4.2 Git push -u origin <branch name>

5. Now, you can pull requst and merge to main branch

In fact you might just delete then add commit push

push -u origin

work for merging as well

11. Conflict of git merge

12. การผูก issue