**Sử dụng GIT và GITHUB**

|  |  |
| --- | --- |
| **Tác vụ** | **Lệnh và giải thích** |
| Tạo 1 folder | mkdir my\_git\_project  cd my\_git\_project |
| [Initialize]  Khởi tạo git cho folder | **git init** |
| [Track]  Kiểm tra tình trạng thay đổi | **git status** |
| [Stage]  Track các thay đổi (folders hoặc files) | git add my\_file myfile2 myfile3  **git add .** |
| [Push]  Chấp thuận và cập nhật các thay đổi | **git commit -m "First Commit"** |
|  |  |
| Xem các thay đổi trong những lần tiếp theo | **git diff**  git diff my\_file |
| [Stage] | git add my\_file myfile2  git add –u |
| Có thể undo git add | **git rm --cached mistake\_file**  rm stands for remove  --cached, we ask Git to untrack the file, but let it remain in the file system |
| [Push] | git commit -m "Made changes to two files" |
| Có thể undo git commit | git commit --amend -m "New Message"  change the commit message of the last commit  git reset --soft HEAD~1  --soft option undoes a commit, but lets the changes you made in that commit remain staged for you to review.  **git revert HEAD~1**  to go back one commit using revert  **git push –f origin master**  Postfixing -f is a dangerous move, as it rewrites the remote without confirming  it. Make sure you double check your local changes before going for an -f push. |
| Xem lại lịch sử commit | **git log –oneline** |
| Xem thông tin của 1 commit cụ thể | **git show b6bd481** |
|  |  |
| Tạo git ignore | **.gitignore**  configuration/  some\_file.m  \*.exe |
|  |  |
| Đồng bộ local repository với github | **git remote add origin** [**https://github.com/sdaityari/my\_git\_project.git**](https://github.com/sdaityari/my_git_project.git)  git push -u origin master  -u option stands for “upstream” |
|  |  |
| Tạo nhánh project | git branch  git branch –a  (Tạo nhánh để thử nghiệm mà không ảnh hưởng đến bản chính. Nếu bản thử nghiệm fails, quay lại bản chính còn nếu success có thể merge kết quả vào bản chính) |
| Tạo 1 branch và sau đó chuyển sang branch | **git branch test\_branch**  (Tạo 1 branch test tuy nhiên vẫn ở current branch)git checkout test\_branch |
| Tạo 1 branch và đồng thời chuyển sang branch | **git checkout -b new\_test\_branch** |
| Tạo 1 branch từ 1 commit cụ thể và chuyển sang branch đó | **git checkout -b old\_commit\_branch cafb55d** |
| Đổi tên current branch thành renamed\_branch | **git branch -m renamed\_branch** |
|  |  |
| Xóa 1 branch | **git branch -D new\_test\_branch**  Don't Delete Branches Unless You Have To |
|  |  |
| Marge 1 branch new\_feature vào 1 branch master | **git checkout master**  **git merge new\_feature** |
|  |
| **git checkout new\_feature**  **git merge another\_feature** |
|  |
|  |  |
| Làm việc trong 1 team trước tiên là clone project từ remote xuống local | **git clone** [**https://github**](https://github)**.com/sdaityari/my\_git\_project.git my\_project** |
| Xác minh lại original remote points từ URL | **git remote –v** |
|  |  |
| Đóng góp vào remote bằng git push | git push  git push remote\_name  **git push remote\_name branch\_name**  **git push remote\_name local\_branch:remote\_branch** |
|  |  |
| Cập nhật project của bạn với remote bằng git pull | **git fetch remote\_name**  To download the changes that have appeared in the remote  **git merge origin/master**  update your local branch you need to merge it with the appropriate  branch from the remote. For instance, if you’re planning to update the local master branch with the remote’s master branch  **git pull origin master**  a shorter way of updating the local branch by downloading and  merging a remote branch is by using pull.  git pull  git pull remote\_name  **git pull remote\_name branch\_name**  **git pull remote\_name local\_branch:remote\_branch** |
|  |  |
| Reject | It involves pushing new code to a remote branch that’s been updated since your last synchronization. In this case, your push would be rejected—with the message that “it is non-fast-forward”. This simply means that, since changes were made to both the remote and your local copy, Git is not able to determine how to merge them.  A conflict arises when your current branch and the branch to be merged have diverged, and there are commits in your current branch that aren’t present in the  other branch, and vice versa. Git isn’t able to determine which changes to keep, so  it raises a conflict to ask the user to review the changes. The last common commit  between the two branches—which is also the point where they diverged—is called  the base commit.  When Git merges the two branches, it looks at the changes in each branch since the  base commit. When there are unambiguous differences—like changes to different  files, and sometimes different parts of the same file—the changes are applied.  However, if there are changes to the same parts of the same file, and Git can’t determine which changes to keep, it raises a conflict.  **git merge friend\_branch** [🡪 error due to conflict]  **git status** [🡪 check why]  **git add sample.py** [do necessary in files that create conflicts then add them]  **git commit -m "Concluded merge with friend\_branch"** [🡪 merge] |
| Nếu merge fail có thể từ bỏ merge | **Git merge –abort**  **Git status** |