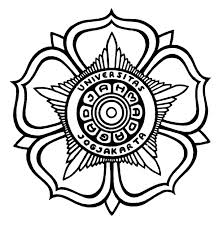
**TUGAS KL WEB**



**Disusun oleh:**

|  |
| --- |
| **Nama : Hanafi Sri Tanjung**  **NIM : 13/344656/SV/03171**  **Kelas : KOMSI A** |

**KOMPUTER DAN SISTEM INFORMASI SV**

**UNIVERSITAS GADJAH MADA**

**2014**

* 1. **Got 15 minutes and want to learn Git?**

Prompt terminal sedang dalam direktori yang diberi nama misalkan “octobox”. Untuk menginisialisasi sebuah repository Git, ketik perintah berikut :

git init

* 1. **Checking the Status**

Direktori "octobox" sekarang memiliki sebuah repositori kosong di/.git /. Repositori adalah direktori tersembunyi dimana Git beroperasi. Untuk melihat keadaan proyek kita saat ini, ketik perintah berikut :

git status

* 1. **Adding & Committing**

Membuat sebuah file bernama octocat.txt dalam repositori octobox. Untuk melihat bagaimana status repositori berubah, ketik perintah berikut :

git status

* 1. **Adding Changes**

Perhatikan bagaimana Git mengatakan octocat.txt " untracked "? Itu berarti bahwa Git melihat bahwa octocat.txt adalah file baru.

Untuk memberitahu Git untuk mulai melacak perubahan yang dibuat untuk octocat.txt, kita perlu menambahkan ini ke staging area dengan menggunakan :

git add octocat.txt

* 1. **Checking for Changes**

Git sekarang adalah memantau berkas octocat.txt kita. Mari kita jalankan git status kembali lagi untuk melihat dimana kedudukan kita:

git status

* 1. **Committing**

Perhatikan bagaimana Git mengatakan changes to be committed? File tercantum di Staging Area, dan mereka tidak dalam repositori kita. Kita dapat menambahkan atau menghapus file dari stage sebelum kita menyimpannya dalam repositori.

Untuk menyimpan perubahan bertahap, kita jalankan perintah commit dengan pesan yang menggambarkan apa yang telah kita rubah. Mari kita lakukan sekarang dengan mengetik:

git commit -m "Add cute octocat story"

* 1. **Adding All Changes**

Kita dapat menggunakan wildcard jika kita ingin menambahkan banyak file dari jenis yang sama. Perhatikan bahwa kita telah menambahkan sekelompok file .txt ke direktori.

Taruh beberapa di sebuah direktori yang dinamakan "octofamily" dan yang lain berakhir di root direktori "octobox". Kita dapat menambahkan semua file baru menggunakan wildcard dengan git add. Jangan lupa tanda kutip!

git add '\*.txt'

* 1. **Committing All Changes**

Menambahkan semua file teks ke staging area. Jalankan git status kembali untuk melihat apa yang kita commit.

git commit -m 'Add all the octocat txt files'

* 1. **History**

Jadi kita telah membuat beberapa commit. Sekarang kita melihat isi mereka untuk melihat apa yang kita rubah.

Git log sebagai sebuah jurnal yang mengingat semua perubahan yang kami telah kita commit sejauh ini, dalam urutan kita commit mereka. Coba jalankannya sekarang:

git log

* 1. **Remote Repositories**

Perintah ini mengambil remote name dan URL repository, yang dalam kasus ini <https://github.com/try-git/try_git.git>.

git remote add origin https://github.com/try-git/try\_git.git

* 1. **Pushing Remotely**

Remote name adalah asal dan nama default dari branch lokal master. -U memberitahu Git untuk mengingat parameter, sehingga waktu berikutnya kita hanya dapat menjalankan git push dan Git akan tahu apa yang harus dilakukan.

git push -u origin master

* 1. **Pulling Remotely**

Kita dapat memeriksa perubahan pada repositori GitHub kita dan menurunkan perubahan baru dengan menjalankan:

git pull origin master

* 1. **Differences**

Dalam kasus ini kita ingin diff dari recent commit, kita dapat merujuk kepada menggunakan HEAD pointer.

git diff HEAD

* 1. **Staged Differences**

Kegunaan lain diff adalah untuk melihat perubahan dalam file yang telah ditunjukkan. Ingat, staged file adalah file yang telah kita beritahu git yang siap untuk di commit.

Kita menggunakan git add ke tahap octofamily/octodog.txt.

git add octofamily/octodog.txt

* 1. **Staged Differences (cont'd)**

Menjalankan git diff dengan --staged untuk melihat perubahan yang kita tampilkan. Kita harus melihat octodog.txt itu diciptakan.

git diff --staged

* 1. **Resetting the Stage**

Kita dapat mengeluarkan file dengan menggunakan perintah git reset. Menghapus octofamily/octodog.txt.

git reset octofamily/octodog.txt

* 1. **Undo**

File dapat berubah kembali ke bagaimana mereka berada di commit terakhir dengan menggunakan perintah: checkout git -- <target>. Menyingkirkan semua perubahan sejak terakhir commit untuk octocat.txt

git checkout -- octocat.txt

* 1. **Branching Out**

Kita ingin menghapus semua octocats, jadi kita membuat branch disebut clean\_up, di mana kita akan melakukan semua pekerjaan:

git branch clean\_up

* 1. **Switching Branches**

Sekarang jika kita mengetikkan git branch kita akan melihat dua branch lokal: branch utama bernama master dan branch baru yang bernama clean\_up.

Kita dapat menggunakan git checkout <branch> command. Coba sekarang untuk beralih ke branch clean\_up :

git checkout clean\_up

* 1. **Removing All The Things**

Jadi kita berada di branch clean\_up. Kita akhirnya dapat menghapus semua octocats tersebut dengan menggunakan perintah git rm yang tidak hanya menghapus file yang sebenarnya dari disk, tetapi juga akan tahap penghapusan file bagi kita.

Kita akan ingin menggunakan wildcard lagi untuk mendapatkan semua octocats dalam satu sapuan.

git rm '\*.txt'

* 1. **Commiting Branch Changes**

Sekarang kita telah menghapus semua cats, kita harus mengcommit perubahan kita.

Menjalankan git status untuk memeriksa perubahan commit kita.

git commit -m "Remove all the cats"

* 1. **Switching Back to master**

Kita hanya perlu beralih kembali ke branch master sehingga kita dapat menyalin (atau menggabungkan) perubahan dari branch clean\_up kembali ke branch master.

git checkout master

* 1. **Preparing to Merge**

Kita sudah di branch master, jadi kita hanya perlu memberitahu Git untuk menggabungkan branch clean\_up ke dalamnya:

git merge clean\_up

* 1. **Keeping Things Clean**

Kita dapat menggunakan git branch -d <branch name>  untuk mengatakan menghapus branch. Menghapus branch clean\_up.

git branch -d clean\_up

* 1. **The Final Push**

Di sini kita berada, pada langkah terakhir. Semua yang tersisa bagi kita untuk lakukan sekarang adalah untuk mendorong segala sesuatu yang telah dikerjakan pada remote repository, dan selesai!

git push