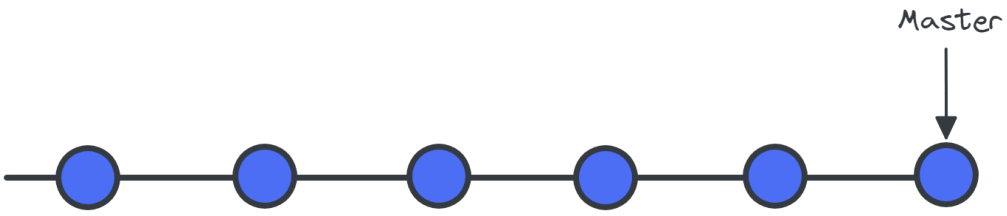
Git & Version Control

Git adalah suatu sistem kontrol version dari suatu text document. Git dibuat oleh Linus Torvalds pada tahun 2005, dan telah dikelola oleh Junio Hamano sejak saat itu.

Git digunakan untuk:
- Melacak perubahan kode
- Melacak siapa yang membuat perubahan
- Kolaborasi dalam mengerjakan kode

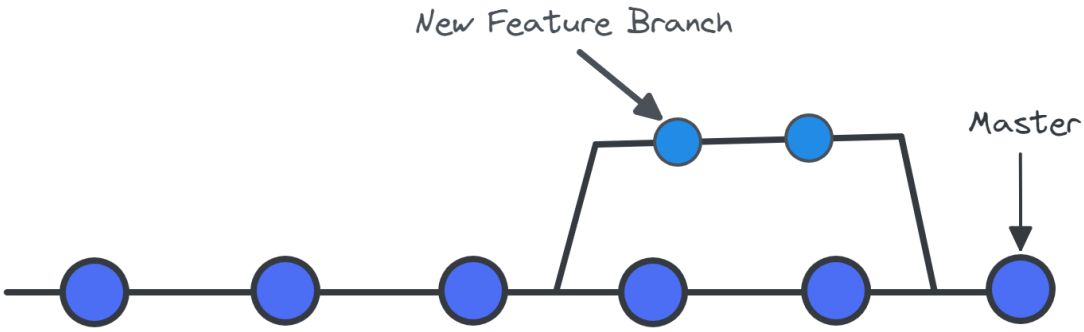
Apa yang Git bisa lakukan?
- Mengelola projects dengan membuat Repositori
- Mengkloning projects untuk mengerjakan salinan kode di sistem lokal
- Mengontrol dan melacak perubahan dengan Staging dan Commiting
- Branch dan Merge untuk memungkinkan pekerjaan pada bagian dan versi projects yang berbeda
- Pull versi terbaru dari projects ke salinan lokal
- Push pembaruan lokal ke projects utama

Bagaimana Git bekerja:
Git menyimpan file dan riwayat perubahan di repositori lokal. Setiap kali kita menyimpan perubahan yang telah dibuat, Git membuat commit. Commit adalah snapshot dari file saat ini. Commit ini terkait satu sama lain. Dengan ini, git memungkinkan kita untuk kembali ke commit sebelumnya, membandingkan perubahan, dan melihat progress dari code yang kita kerjakan. Commit diidentifikasi oleh hash unik yang digunakan untuk membandingkan dan mengembalikan perubahan yang telah dibuat.



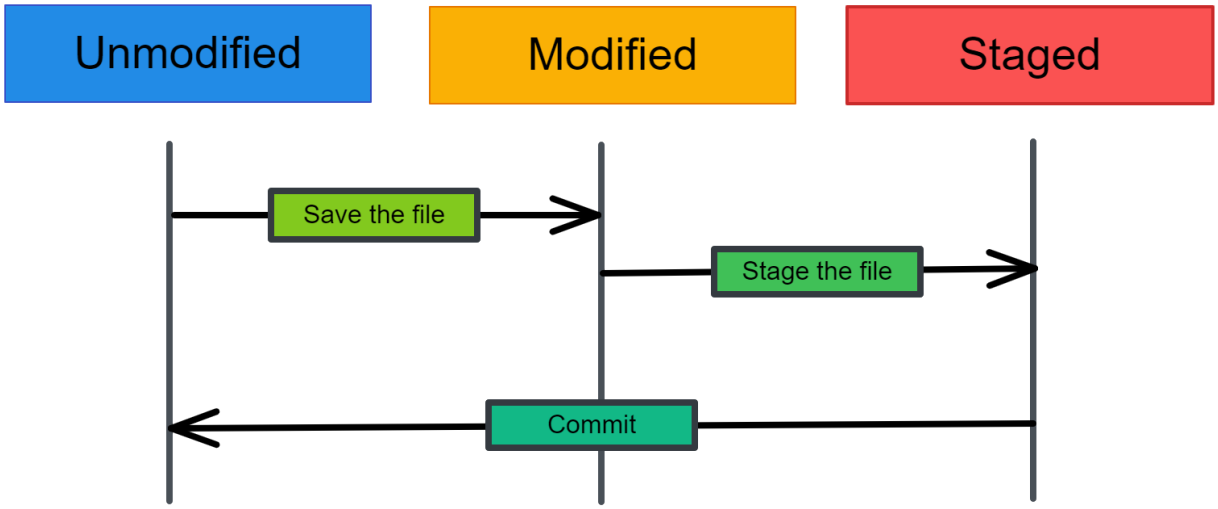
Branch

Branch adalah salinan dari source code yang bekerja paralel dengan versi utama. Untuk menyimpan perubahan yang dibuat, gabungkan branch ke versi utama (master/main). Branch memungkinkan kita bekerja secara bersama-sama secara async. Setiap orang memiliki tugasnya masing-masing, dan dengan menggunakan branch, mereka dapat mengerjakan fitur baru tanpa campur tangan rekan satu tim lainnya. Setelah tugas selesai, kita dapat menggabungkan fitur baru dengan versi utama (master/main).



Commits

Ada tiga status file di Git: dimodifikasi (modified), staged, dan commit. Saat kita membuat perubahan dalam file, perubahan akan disimpan di direktori lokal. Perubahan di lokal bukan bagian dari history yang disimpan Git. Untuk membuat commit, kita harus terlebih dahulu melakukan stage pada file yang diubah. Kita dapat menambahkan atau menghapus perubahan di area staged dan kemudian kemas perubahan ini sebagai commit dengan pesan yang menjelaskan perubahan yang terjadi.



Install Git di Local Computer

Install Git pada local komputer, sesuai dengan OS yang Anda pakai, step by step installasi dapat dilihat di link berikut - https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

Check Versi git melalui Local Terminal / Powershell

Setelah melakukan instalasi, kita dapat menjalankan script shell dibawah ini untuk memastikan apakah git sudah terinstall di local computer kita.

```
git --version
```

Jika git telah terinstall, makan command diatas akan menghasilkan contoh output seperti:
Untuk MacOS:

```
git version 2.17.2 (Apple Git-113)
```

Untuk Windows:

```
git version 2.30.2.windows.1
```

Setup Config

Sekarang beri tahu Git siapa Anda. Ini penting untuk git, karena setiap commit Git menggunakan informasi ini:

```
git config --global user.name "Angga Pradikta"
git config --global user.email "email@example.com"
```

Git Repository (Repo)

Git Repository adalah penyimpanan virtual untuk project kita. Git Repository memungkinkan kita untuk menyimpan versi kode kita, yang dapat diakses bila diperlukan.

Membuat Git Repo

Untuk membuat repo baru, kita akan menggunakan perintah git init. git init adalah perintah satu kali yang digunakan selama pengaturan awal repo baru. Menjalankan perintah ini akan membuat subdirektori .git baru di direktori kerja Anda saat ini. Ini juga akan membuat branch utama baru.

```
cd /path/to/your/existing/code
git init
```

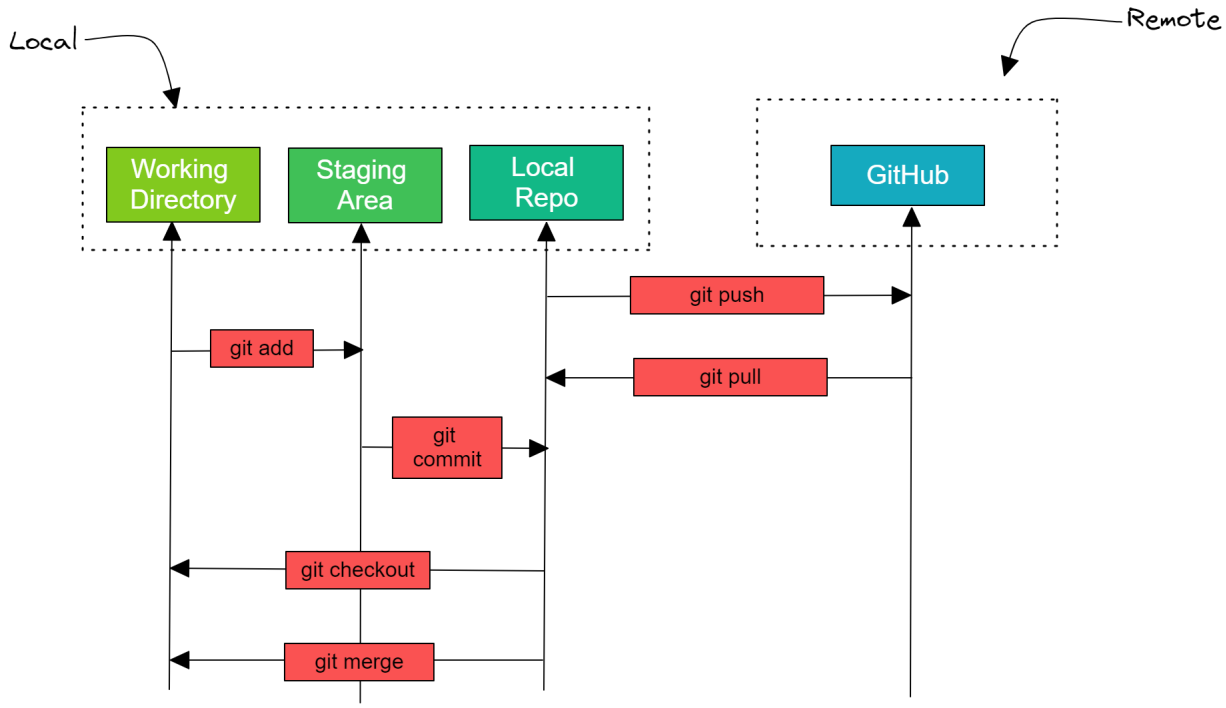
Signup / Signin ke akun Github dan Buat Repository

Masuk atau buat akun Github menggunakan email Anda. Buat repository dengan nama, misalnya git-tutorial. https://github.com/

Github Authentication

```
gh auth login
```

Github authentication bisa dijalankan ketika kita sudah mempunyai github cli terinstall, untuk menginstall github client, bisa menggunakan link dibawah
https://github.com/cli/cli



Clone Repository ke Local Komputer

Dari repo github yang telah Anda bikin, klik code dan salin URL yang tersedia. Pada komputer Anda, buka Terminal (Mac) atau Powershell (Win) masuk ke direktori yang ingin Anda tuju dan gunakan git clone untuk menyalin repo ke local komputer Anda

```
git clone https://github.com/USERNAME_ANDA/git-tutorial.git
cd git-tutorial
```

Check Branch & Buat Branch Baru

Gunakan git branch untuk check branch apa yang tersedia, lalu buat branch baru dengan git branch, setelah itu masuk ke branch baru tersebut dengan git checkout. Sangat tidak disarankan untuk mengubah secara langsung main branch.

```
git branch tutorial-1
git checkout tutorial-1
```

Buat File Baru, Commit Perubahannya dan Push Perubahan

Buat file baru didalam branch baru tersebut

```
touch tutorial.txt
vi tutorial.txt
```

Ketik i

Masukkan text misal: Git Tutorial Digital Skola

Ketik :wq untuk menyimpan dan keluar dari file

Atau gunakan

```
echo "Git Tutorial Digital Skola" > tutorial.txt
```

Lalu lakukan commit ke repo Anda dengan menggunakan.

```
git add .
git status
git commit -m "commit pertama"
git push origin tutorial-1
```

Buat Pull/Merge Request

Buat pull atau merge request untuk menggabungkan perubahan dari file pada branch tutorial-1 ke main branch.

Mengabaikan Files

Buat file baru di local komputer misalnya: ignore_this.txt. Lalu masukkan test kedalam file tersebut

```
echo "please ignore me" > ignore_this.txt
```

masukkan nama file kedalam .gitignore.

Commit, push dan merge seperti langkah diatas

Check Log dan Perubahan Apa saja yang dilakukan

```
git checkout
```

Track Perubahan dari Remote Server

```
git branch --set-upstream-to=origin/<branch> test-1
git pull
```

Check Log dan Perubahan Apa saja yang dilakukan

```
git log
git checkout
```

Mengembalikan Perubahan

Sebelum Merge

```
git revert HEAD
```

Sesudah Merge

```
git log , untuk check bash code merge commit
git revert ba0cd9c -m 2
```

Lalu commit, push dan merge

Menghapus Branch

```
git branch -D nama_branch
```

Membuat Repo dari Local

```
Git init
```

Add remote & Push Changes

```
git remote add origin https://github.com/myusername/mynewrepository.git
git push -u origin master
```

How to Reproduce Conflict & Solve it

Merge conflict terjadi ketika beberapa orang membuat perubahan berbeda pada baris yang sama dari file yang sama, atau ketika satu orang mengedit file dan orang lain menghapus file yang sama.

Reproduce conflicts

```
mkdir git-repo
cd git-repo
git init
touch my_code.sh
git add my_code.sh
echo "echo Hello" > my_code.sh
git commit -m 'initial'

git checkout -b new_branch
echo "echo \"Hello World\"" > my_code.sh
git add .
git commit -m 'first commit on new_branch'
git checkout master
echo "echo \"Hello World!\"" > my_code.sh
git commit -m 'second commit on main
git merge new_branch
```

https://jonathanmh.com/how-to-create-a-git-merge-conflict/
https://www.atlassian.com/git/tutorials/using-branches/merge-conflicts

Solve merge conflict

Pastikan kita berada di working repo project kita.

```
cd ~/<repo_directory>
```

Contohnya, jika repo kita bernama my-repository, maka scriptnya akan terlihat seperti ini:

```
cd ~/my-repository
```

Pull versi paling baru repo project kita.

```
git pull
```

Check out source branch.

```
git checkout <feature_branch>
```

Pull branch tujuan ke source branch. Pada saat ini, pulling ke destination akan mencoba untuk melakukan merge dengan source branch dan menampilkan semua conflicts.

```
git pull origin <destination_branch>
```

Ketika kita mencoba melakukan merge dua branch locally, When you merge two branches with conflicts locally, you'll get conflict markers in the file when you open your editor.

Open the file to resolve the conflict. You can do this using the command line or you can navigate to the file.

Resolve the conflict by doing the following:

1. Remove the change designations added by Git
2. Correct the content
3. Save the file

Add and commit the change.

```
git add <filename>
git commit -m 'commit message'
```

Push the change to the remote.

```
git push origin <feature_branch>
```

Clone Private Repo

Untuk clone private repo, kita harus memasukkan SSH keys dari laptop kita. Check ssh key apakah sudah tergenerate di komputer kita, ketik command berikut di terminal / shell:

```
cat ~/.ssh/id_rsa.pub
```

Jika sudah ada kita tinggal copy konten yang tercetak di layar terminal dan masukkan ke github → Settings → SSH & GPG Keys → Add SSH.

Jika belum, maka kita men-generate SSH keys terlebih dahulu dengan command berikut:

```
ssh-keygen -o -t rsa -C "ssh@github.com"
```

Tekan enter sampai proses terakhir, lalu lakukan langkah sebelumnya.

Git Cheat Sheet

Dasar - dasar Git

Command	Descriptions
git init <nama_directory>	Membuat repositori (repo) Git kosong di direktori tertentu.
git clone <link_repo>	Kloning repo terletak di <link_repo> ke computer lokal. Repo asli bisa terletak di sistem file lokal atau pada mesin jarak jauh melalui HTTP atau SSH.
git config -- global user.name "nama_anda"	Tentukan nama penulis yang akan digunakan untuk semua commit dalam

	repo saat ini.
git add <nama_directory>	Simpan di stage semua perubahan <nama_directory> untuk commit berikutnya. Ganti <nama_directory> dengan <nama_file> untuk mengubah ke file tertentu.
git commit -m "tulis message"	Commit snapshot yang telah disimpan di stage, tuliskan message sebagai pesan komit.
git status	Mencantumkan file mana yang di staged, di unstaged, dan yang tidak terlacak.
git log	Tampilkan seluruh riwayat commit.
git diff	Memperlihatkan perubahan tanpa label (unstaged) antara indeks di local dan direktori kerja
git push <remote-name> <branch-name>	kirim commit lokal ke branch repositori jarak jauh (remote repo).
git checkout -b <branch-name>	membuat branch baru dan beralih ke branch baru.
git remote -v	Melihat semua remote repo di akun tertentu
git pull	menggabungkan commit ke direktori lokal dari repositori jarak jauh.
git branch –d <branch-name>	Menghapus branch
git merge <branch-name>	setelah menyelesaikan merge conflict, command ini menggabungkan branch yang dipilih ke branch saat ini.