KelasIT.com

PANDUAN GIT & GITHUB
UNTUK PEMULA

Ringkas, Praktis, Langsung Bisa Dipakai

Edisi 2026 • eBook Panduan Developer Pemula

Hifzon, M.Kom

BAB 1

Apa itu Git & GitHub?

Git adalah sistem version control: alat untuk mencatat setiap perubahan pada kode, sehingga kamu bisa melihat riwayat, membandingkan versi, dan kembali ke versi sebelumnya kapan saja. Git berjalan lokal di komputer kamu.

GitHub adalah layanan online untuk menyimpan repository Git di cloud, sekaligus tempat berkolaborasi dengan orang lain lewat fitur seperti Pull Request, Issue, dan Code Review. Git adalah alatnya, GitHub adalah tempat nongkrongnya.

Instalasi Singkat

- Windows/Mac/Linux: unduh dari git-scm.com, lalu install seperti biasa.
- Cek instalasi: jalankan `git --version` di terminal.
- Set identitas sekali saja: `git config --global user.name "Nama"` dan `user.email`.

TIPS: Kenapa Penting
Tanpa version control, kamu terpaksa menyimpan banyak folder "project_final_fix2_beneran.zip". Git menggantikan semua itu dengan riwayat yang rapi.

BAB 2

Konsep Dasar yang Wajib Dipahami

- **Repository (repo)** — folder proyek yang riwayat perubahannya dilacak oleh Git.
- **Commit** — "snapshot" perubahan pada titik waktu tertentu, disertai pesan penjelas.
- **Staging area** — area sementara untuk memilih file mana saja yang akan dimasukkan ke commit berikutnya.
- **Branch** — jalur pengembangan terpisah, biasanya dipakai untuk mengerjakan fitur tanpa mengganggu kode utama.
- **Remote** — salinan repository yang tersimpan di server lain, misalnya GitHub.

Alur Dasar Git

```
1. Edit file di komputer
2. git add    -> masukkan ke staging area
3. git commit -> simpan snapshot + pesan
4. git push   -> kirim ke GitHub (remote)
```

TIPS: Analogi Sederhana

Anggap staging area seperti keranjang belanja: kamu pilih dulu item yang mau di-checkout (commit), tidak semua barang di toko ikut terbawa.

BAB 3

Perintah Git Sehari-hari

Ini kumpulan perintah yang akan paling sering kamu pakai setiap hari.

```
git init                  # mulai repo baru di folder ini
git status                # lihat perubahan yang belum di-commit
git add nama_file         # tambahkan file ke staging
git add .                 # tambahkan semua perubahan
git commit -m "pesan"     # simpan snapshot dengan pesan
git log                   # lihat riwayat commit
git diff                  # lihat detail perubahan baris kode
```

Tips Menulis Pesan Commit

- Gunakan kalimat singkat dan jelas: "Perbaiki bug login" lebih baik dari "update".
- Satu commit sebaiknya fokus pada satu perubahan logis.
- Gunakan awalan konsisten kalau tim punya konvensi, misalnya `fix:`, `feat:`, `docs:`.

BAB 4

Bekerja dengan GitHub (Remote)

Setelah repo lokal siap, hubungkan ke GitHub supaya kode tersimpan online dan bisa diakses tim.

```
git clone <url>              # salin repo dari GitHub ke lokal
git remote add origin <url>  # hubungkan repo lokal ke GitHub
git push origin main         # kirim commit ke GitHub
git pull origin main         # ambil update terbaru dari GitHub
```

Clone vs Init

- Pakai `git clone` kalau repo sudah ada di GitHub dan kamu mau mulai kerja di komputer sendiri.
- Pakai `git init` kalau kamu mulai proyek baru dari nol di komputer lokal.

TIPS: Selalu Pull Dulu
Biasakan `git pull` sebelum mulai kerja, supaya kamu bekerja dengan kode paling baru dan menghindari conflict yang tidak perlu.

BAB 5

Branching & Merging

Branch memungkinkan kamu bereksperimen atau mengerjakan fitur baru tanpa mengganggu kode utama (biasanya branch `main`).

```
git branch                  # lihat daftar branch
git branch fitur-login      # buat branch baru
git checkout fitur-login    # pindah ke branch tsb
git checkout -b fitur-login # buat + pindah sekaligus
git merge fitur-login       # gabungkan ke branch aktif
```

Kapan Membuat Branch Baru?

- Setiap mengerjakan fitur baru.
- Saat memperbaiki bug tertentu (bug-fix branch).
- Saat ingin mencoba pendekatan berbeda tanpa risiko merusak kode utama.

BAB 6

Mengatasi Merge Conflict

Conflict terjadi saat Git tidak bisa otomatis menggabungkan perubahan karena dua branch mengubah baris yang sama. Ini normal dan bisa diselesaikan dengan tenang.

Langkah Menyelesaikan Conflict

- Buka file yang konflik — Git menandai bagian bermasalah dengan `<<<<<<<`, `=======`, `>>>>>>>`.
- Putuskan versi mana yang dipakai, atau gabungkan keduanya secara manual.
- Hapus tanda penanda konflik setelah selesai diedit.
- Jalankan `git add` pada file tersebut, lalu `git commit` untuk menyelesaikan merge.

TIPS: Jangan Panik
Conflict bukan tanda kesalahan besar — itu bagian normal dari kerja tim. Baca konteksnya pelan-pelan sebelum memutuskan versi mana yang benar.

BAB 7

Kolaborasi via Pull Request

Pull Request (PR) adalah cara mengajukan perubahan dari branch kamu supaya direview dan digabungkan ke branch utama oleh tim.

Alur Kerja Umum

```
1. git checkout -b fitur-baru
2. ...edit kode & commit...
3. git push origin fitur-baru
4. Buka GitHub -> buat Pull Request
5. Tim review & diskusi di PR
6. Setelah disetujui -> merge ke main
```

Tips Membuat PR yang Enak Direview

- Beri judul dan deskripsi PR yang jelas: apa yang berubah dan kenapa.
- Usahakan PR tidak terlalu besar — lebih mudah direview jika fokus satu topik.
- Balas komentar reviewer dengan commit tambahan, jangan menghapus riwayat diskusi.

BAB 8

Kesalahan Umum & Solusinya

- **Salah commit message** — perbaiki dengan `git commit --amend -m "pesan baru"` (sebelum push).
- **Lupa menambahkan file ke commit** — `git add` file yang lupa, lalu `git commit --amend`.
- **Ingin membatalkan perubahan yang belum di-commit** — gunakan `git checkout -- nama_file`.
- **Ingin mundur ke commit sebelumnya (aman)** — gunakan `git revert <hash>`, bukan menghapus riwayat.
- **Push ditolak karena ketinggalan update** — jalankan `git pull` dulu, selesaikan conflict jika ada, baru push lagi.

TIPS: Hindari git reset --hard Sembarangan

Perintah ini menghapus perubahan secara permanen. Pastikan kamu benar-benar yakin sebelum memakainya, terutama di branch bersama.

BAB 9

Cheat Sheet Ringkas

`git init`	`git branch`
`git clone <url>`	`git checkout -b <branch>`
`git status`	`git merge <branch>`
`git add .`	`git pull origin main`
`git commit -m "pesan"`	`git push origin main`
`git log`	`git revert <hash>`
`git diff`	`git commit --amend -m "..."`

Simpan cheat sheet ini di dekat mejamu. Semakin sering dipakai, perintah-perintah ini akan hafal dengan sendirinya.

Selamat belajar Git & GitHub — sedikit demi sedikit, lama-lama jadi kebiasaan!