Afapedia Studio

Cheat Sheet Git untuk Developer

Perintah Git yang dipakai setiap hari, alur branch dan pull request, cara menyelesaikan konflik, dan cara membatalkan kesalahan tanpa panik.

Edisi November 2026 · Berlaku untuk Git 2.23 ke atas (perintah switch dan restore) · Gratis dari afapedia.com

Aturan aman: sebelum perintah yang mengubah riwayat, cek dulu dengan `git status` dan `git log --oneline`. Hindari `reset --hard` dan `push --force` di branch yang dipakai bersama.

1 Setup sekali

```
git config --global user.name "Nama Kamu"
git config --global user.email "email@kamu.com"
git config --global init.defaultBranch main
git config --global pull.rebase false
```

2 Memulai repositori

`git init`	Mulai repo baru di folder ini
`git clone URL`	Salin repo dari GitHub/GitLab
`git remote -v`	Lihat alamat remote
`git remote add origin URL`	Hubungkan ke remote

3 Alur harian

`git status`	Apa yang berubah
`git diff`	Lihat perubahan yang belum di-stage
`git diff --staged`	Lihat yang akan di-commit
`git add file`	Stage file tertentu
`git add -p`	Pilih perubahan per bagian
`git commit -m "..."`	Simpan titik riwayat
`git pull`	Ambil dan gabungkan perubahan remote
`git push`	Kirim commit ke remote

4 Branch

`git branch`	Daftar branch lokal
`git switch -c fitur-x`	Buat dan pindah ke branch baru
`git switch main`	Pindah branch
`git merge fitur-x`	Gabungkan ke branch aktif
`git branch -d fitur-x`	Hapus branch yang sudah digabung
`git push -u origin fitur-x`	Kirim branch baru ke remote
`git fetch --prune`	Sinkron daftar branch remote

5 Alur pull request

```
git switch main && git pull
git switch -c perbaiki-validasi-telepon
# ... edit, test, commit kecil-kecil ...
git push -u origin perbaiki-validasi-telepon
# buka PR di GitHub, minta review
# setelah merge:
git switch main && git pull
git branch -d perbaiki-validasi-telepon
```

6 Melihat riwayat

`git log --oneline --graph`	Riwayat ringkas bergambar
`git log -p file`	Riwayat perubahan satu file
`git show HASH`	Isi satu commit
`git blame file`	Siapa mengubah baris mana
`git reflog`	Jejak semua posisi HEAD, penyelamat saat "hilang"

7 Membatalkan kesalahan

SITUASI	PERINTAH
Buang perubahan file (belum add)	`git restore file`
Keluarkan dari staging	`git restore --staged file`
Ubah pesan commit terakhir*	`git commit --amend`
Batalkan commit terakhir, simpan perubahan*	`git reset --soft HEAD~1`
Batalkan commit yang sudah di-push	`git revert HASH`
Kembalikan file ke versi commit lama	`git restore --source HASH file`

* Hanya untuk commit yang belum di-push.

8 Simpan sementara (stash)

`git stash`	Simpan perubahan, kembalikan ke bersih
`git stash -u`	Termasuk file baru
`git stash list`	Daftar simpanan
`git stash pop`	Kembalikan simpanan terakhir

9 Menyelesaikan konflik

```
<<<<<<< HEAD
Harga: Rp50.000
=======
Harga: Rp55.000
>>>>>>> fitur-harga-baru
```

1. Pilih atau gabungkan isi yang benar.
2. Hapus tanda `<<<`, `===`, `>>>`.
3. `git add file`, lalu `git commit`.
4. Ragu? `git merge --abort` untuk kembali ke sebelum merge.

10 .gitignore untuk proyek web

```
.env
/vendor
/node_modules
/public/build
/storage/*.key
*.log
.DS_Store
.idea/
.vscode/*
!.vscode/extensions.json
```

.env terlanjur ter-push? Anggap bocor: ganti password dan API key-nya dulu.

`git rm --cached .env` menghapusnya dari pelacakan tanpa menghapus file lokal.

11 Pesan commit yang baik

Hindari: `update`, `fix`, `final2`

Lebih baik:

- `Perbaiki validasi nomor telepon di form daftar`
- `Tambah ekspor laporan penjualan ke CSV`
- `Pindahkan kirim email ke queue`

Kalimat perintah, singkat, menjelaskan apa yang berubah. Alasan ditulis di baris setelah baris kosong.

12 Kebiasaan yang mencegah masalah

- `git pull` sebelum mulai bekerja.
- Satu branch untuk satu fitur atau perbaikan.
- Commit kecil dan sering, PR kecil cepat direview.
- Lindungi branch `main`: wajib review dan CI hijau.
- Buat branch sebelum memberi tugas ke AI agent, agar perubahannya mudah ditinjau atau dibuang.

13 Rebase, cherry-pick, dan tag

`git rebase main`	Pindahkan commit branch-mu ke atas main terbaru; hanya untuk branch pribadi
`git rebase --abort`	Batalkan rebase yang bermasalah
`git cherry-pick HASH`	Ambil satu commit dari branch lain, misalnya hotfix
`git tag v1.2.0`	Tandai versi rilis
`git push --tags`	Kirim tag ke remote
`git push --force-with-lease`	Push setelah rebase, lebih aman daripada `--force`

14 Darurat: "commit saya hilang"

1. Jangan panik dan jangan menjalankan perintah lain dulu.
2. `git reflog` untuk melihat posisi HEAD sebelumnya.
3. `git switch -c penyelamatan HASH` untuk membuat branch dari posisi itu.
4. Periksa isinya, lalu gabungkan kembali.

Commit yang sudah pernah dibuat hampir selalu masih bisa ditemukan lewat reflog selama beberapa minggu.

© 2026 Afapedia Studio · Boleh dibagikan, tidak untuk dijual