# TEKNOLOGI PERKANTORAN

# MODUL

# *PENGENALAN* *GIT*

1. Pokok Bahasan Praktikum
2. Pengenalan Git
3. Cara Kerja Git
4. Tujuan Instruksional Praktikum
5. Mahasiswa memahami cara kerja GIT
6. Mahasiswa dapat menggunakan GIT untuk menyimpan pekerjaan
7. Durasi

60 menit

# MATERI

## Memulai GIT

### Pengenalan Version Control

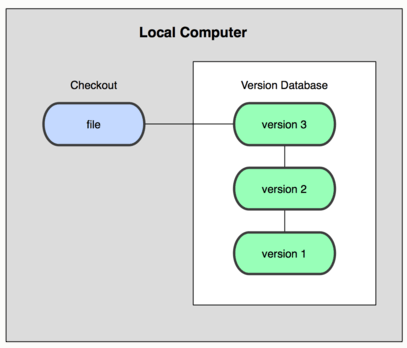
Version control adalah sebuah sistem yang mencatat setiap perubahan terhadap sebuah berkas atau kumpulan berkas sehingga pada suatu saat anda dapat kembali kepada salah satu versi dari berkas tersebut.

Jika anda adalah seorang desainer grafis atau desainer web dan anda ingin menyimpan setiap versi dari gambar atau layout yang anda buat (kemungkinan besar anda pasti ingin melakukannya), maka Version Control System (VCS) merupakan sebuah solusi bijak untuk digunakan. Sistem ini memungkinkan anda untuk mengembalikan berkas anda pada kondisi/keadaan sebelumnya, mengembalikan seluruh proyek pada keadaan sebelumnya, membandingkan perubahan setiap saat, melihat siapa yang terakhir melakukan perubahan terbaru pada suatu objek sehingga berpotensi menimbulkan masalah, siapa yang menerbitkan isu, dan lainnya. Dengan menggunakan VCS dapat berarti jika anda telah mengacaukan atau kehilangan berkas, anda dapat dengan mudah mengembalikannya.

### Version Control System Lokal

Kebanyakan orang melakukan pengontrolan versi dengan cara menyalin berkas-berkas pada direktori lain (mungkin dengan memberikan penanggalan pada direktori tersebut, jika mereka rajin). Metode seperti ini sangat umum karena sangat sederhana, namun cenderung rawan terhadap kesalahan. Anda akan sangat mudah lupa dimana direktori anda sedang berada, selain itu dapat pula terjadi ketidak sengajaan penulisan pada berkas yang salah atau menyalin pada berkas yang bukan anda maksudkan.

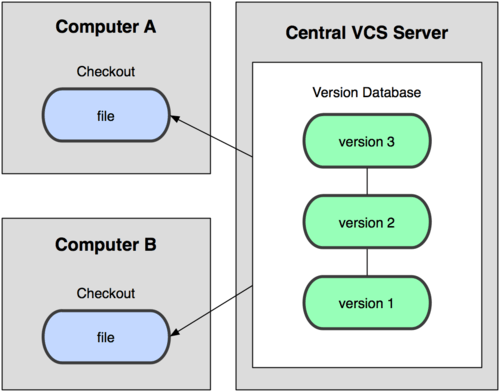
Untuk mengatasi permasalahan ini, dikemkembangkan berbagai VCS lokal yang memiliki sebuah basis data sederhana untuk menyimpan semua perubahan pada berkas yang berada dalam cakupan revision control.



Contoh VCS lokal adalah rcs. *Revision Control System* (RCS) mampu mengelola revisi dari banyak *file*. RCS mengotomasi penyimpanan, pengambilan, pencatatan, pemeriksaan, dan penggabungan dari sebuah revisi. Berguna untuk teks yang sering direvisi seperti *source code*, *program*, dokumentasi, grafik, jurnal, dan surat.

### Version Control Systems Terpusat

Permasalahan berikutnya yang dihadapi adalah para pengembang perlu melakukan kolaborasi dengan pengembang pada sistem lainnya. Untuk mengatasi permasalahan ini maka dibangunlah Centralized Version Control Systems (CVCSs). Sistem ini, diantaranya CVS, Subversion, dan Perforce, memiliki sebuah server untuk menyimpan setiap versi berkas, dan beberapa klien yang dapat melakukan checkout berkas dari server pusat. Untuk beberapa tahun, sistem seperti ini menjadi standard untuk version control.



Sistem seperti ini memiliki beberapa kelebihan, terutama jika dibandingkan dengan VCS lokal. Misalnya, setiap orang pada tingkat tertentu mengetahui apa yang orang lain lakukan pada proyek. Administrator memiliki kendali yang mantap atas siapa yang dapat melakukan apa; dan adalah jauh lebih mudah untuk mengelola sebuah CVCS dibandingkan menangani database lokal pada setiap client.

Walau demikian, sistem dengan tatanan seperti ini memiliki kelemahan serius. Kelemahan nyata yang direpresesntasikan oleh sistem dengan server terpusat. Jika server mati untuk beberapa jam, maka tidak ada seorangpun yang bisa berkolaborasi atau menyimpan perubahan terhadap apa yang mereka telah kerjakan. Jika harddisk yang menyimpan basisdata mengalami kerusakan, dan salinan yang beran belum tersimpan, anda akan kehilangan setiap perubahan dari proyek kecuali snapshot yang dimiliki oleh setiap kolaborator pada komputernya masing-masing. VCS lokal juga mengalami nasib yang sama jika anda menyimpan seluruh history perubahan proyek pada satu tempat, anda mempunyai resiko kehilangan semuanya.

### Version Control System Terdistribusi

Inilah saatnya bagi Distributed Version Control Systems untuk mengambil tempat. dalam sebuah DVCS (seperti Git, Mercurial, Bazaar atau Darcs), klien tidak hanya melakukan checkout untuk snapshot terakhir setiap berkas, namun mereka (klien) memiliki salinan penuh dari repositori tersebut. Jadi, jika server mati, dan sistem berkolaborasi melalui server tersebut, maka klien manapun dapat mengirimkan salinan repositori tersebut kembali ke server. Setiap checkout pada DVCS merupakan sebuah backup dari keseluruhan data (lihat Gambar 1-3).



Lebih jauh lagi, kebanyakan sistem seperti ini mampu menangani sejumlah remote repository dengan baik, jadi Anda dapat melakukan kolaborasi dengan berbagai kelompok kolaborator dalam berbagai cara secara bersama-sama pada suatu proyek. Hal ini memungkinkan Anda untuk menyusun beberapa jenis alur kerja yang tidak mungkin dilakukan pada sistem terpusat, seperti hierarchical model.

## Dasar-Dasar GIT

### Instalasi GIT

Silahkan buka website resminya Git ( [git-scm.com](https://git-scm.com/" \t "_blank)). Kemudian unduh Git sesuai dengan arsitektur komputer kita. Kalau menggunakan 64bit, unduh yang 64bit. Begitu juga kalau menggunakan 32bit. Langkah-langkahnya adalah sebagai berikut :

1. Silahkan klik 2x file instaler Git yang sudah diunduh.
2.  klik Next > untuk melanjutkan.
3. Selanjutnya menentukan lokasi instalasi. Biarkan saja apa adanya, kemudian klik *Next >*.
4. Selanjutnya pemilihan komponen, biarkan saja seperti ini kemudian klik *Next>*.



1. Selanjutnya pemlilihan direktori start menu, klik *Next >*.
2. Selanjutnya pengaturan *PATH Environment*. Pilih yang tengah agar erintah git dapat di kenali di *Command Prompt* (CMD). Setelah itu klik *Next >*.



1. Selanjutnya konversi *line ending*. Biarkan saja seperti ini, kemudian klik *Next >*.

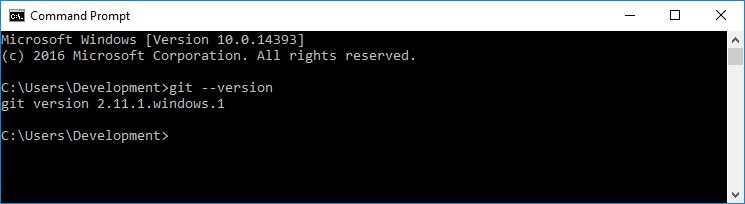


1. Selanjutnya pemilihan emulator terminal. Pilih saja yang bawah, kemudian klik *Next >*.



1. Selanjutnya pemilihan opsi ekstra. Klik saja *Next >*.
2. Selanjutnya pemilihan opsi ekspreimental, langsung saja klik *Install* untuk memaulai instalasi dan tunggu beberapa saat, instalasi sedang dilakukan.
3. Setelah selesai, kita bisa langsung klik *Finish*.

Selamat, Git sudah terinstal di Windows. Untuk mencobanya, silahkan buka CMD atau PowerShell, kemudian ketik perintah git --version.



### Membuat Repository di Github

Terlebih dahulu, Anda perlu membuat akun Github dengan menggunakan akun email anda. Kemudian silahkan buka Github, kemudian buat sebuah repository dengan nama **belajar-git** seperti berikut ini. Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Maka sekarang kita punya repository kosong di Github.

Graphical user interface, text, application, email

Description automatically generated

Jangan diapa-apakan dulu. Silahkan buka kembali repository lokal yang pernah kita buat, yaitu belajar-git. Kita akan upload ke Github.

Sebelum kita bisa upload semua revisi yang ada di repository lokal, kita harus menambahkan remote-nya terlebih dahulu. Remote dapat kita tambahkan dengan perintah seperti ini:

Setelah itu, silahkan ketik perintah git remote -v untuk melihat remote apa saja yang sudah ditambahkan.

Text

Description automatically generated

sekarang kita sudah menambahkan remote di dalam repository lokal. Selanjutnya kita bisa melakukan push atau mengirim revisi ke repository remote (Github) dengan perintah git push <nama remote> <nama cabang tujuan>. Contoh :

Text

Description automatically generated

Sekarang lihat ke repository Github Anda, pasti semuanya sudah ter-upload disana.

A screenshot of a computer

Description automatically generated

### Github Desktop

Github Desktop adalah aplikasi git client yang digunakan untuk menyederhanakan penggunaan GIT khususnya pada repository central Github. Github Desktop dibangun dengan tujuan berikut ini:

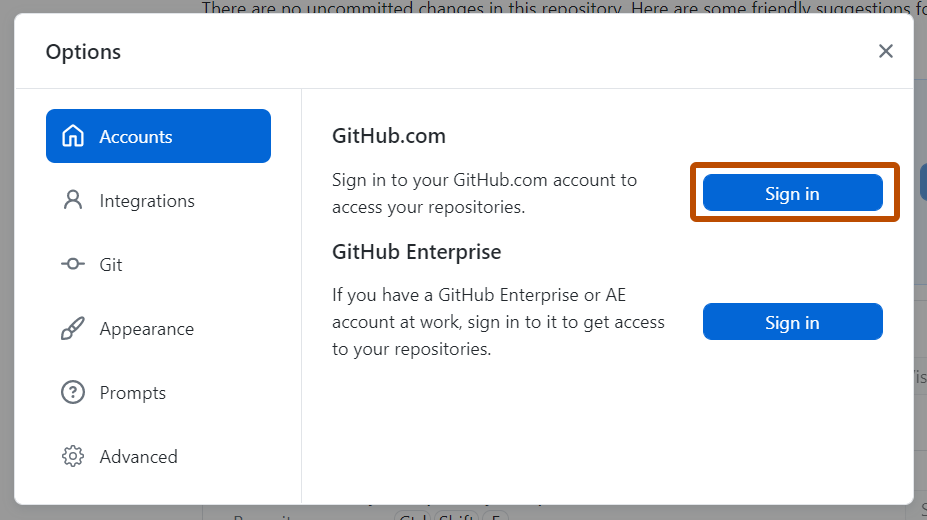
* GitHub Desktop mengurangi kompleksitas dalam menggunakan Git dan menyederhanakan alur kerja Git dan GitHub\*.
* Menyederhanakan alurkerja Github bagi pemula dan team developer untuk tetap dapat berkolaborasi.

#### Instalasi dan Autentikasi

Proses intalasi dapat dilakukan dengan mendowload aplikasi dari link berikut :

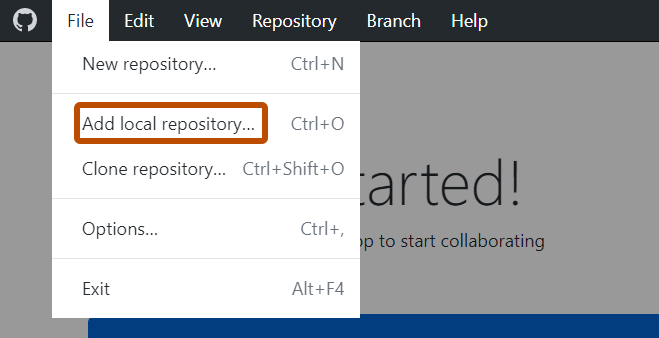
<https://desktop.github.com/>

Sebelum dapat menggunakan Github, Anda perlu melakukan autentikasi terlebiih dahulu.

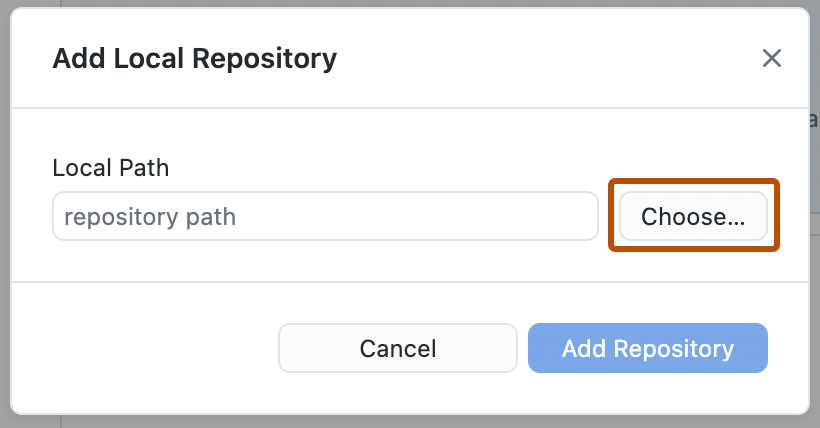


#### Add repositories

* 1. Anda juga dapat menambahkan repository yang sudah ada pada computer anda dengan menekan menu **File** dan **Add Local Repository**.



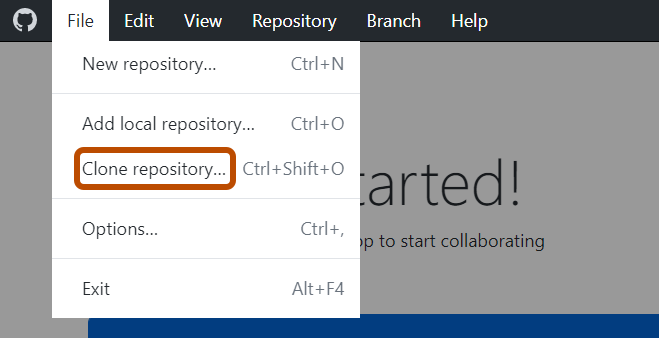
* 1. Pilih menu Add Local Repository, dan pilih Choose, untuk memilih lokasi tempat repository berada.



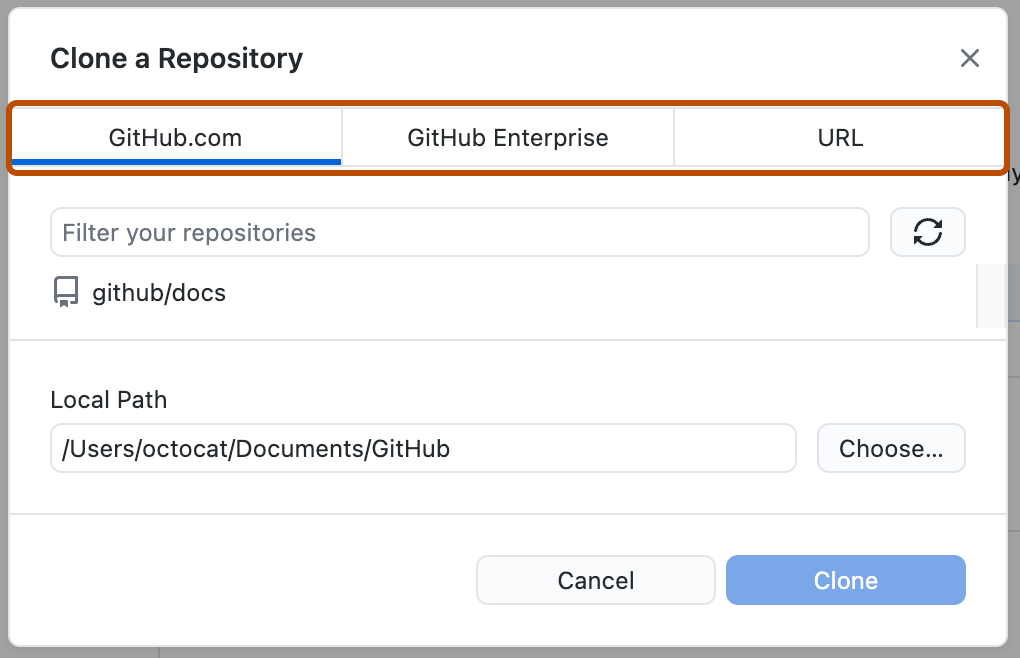
* 1. Tekan Add repository.

#### Cloning repositories

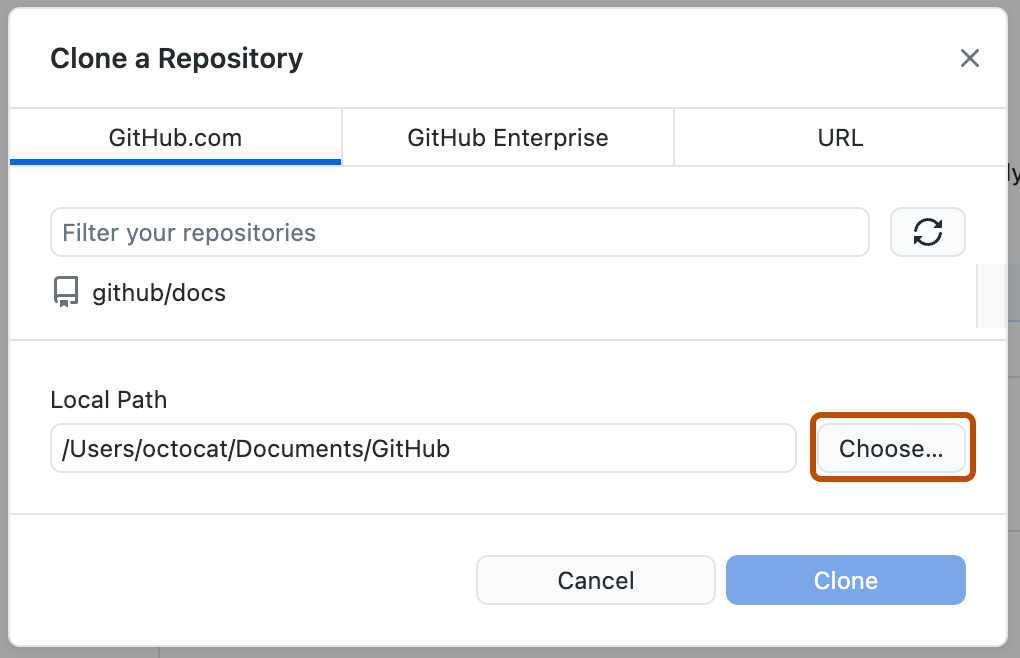
1. Proses Clone dapat dilakukan dengan menekan menu berikut



1. Pilih tab yang sesuai dengan lokasi repository berada.



1. Pilih lokasi folder tempat repository akan di clone.



#### Meng-Commit perubahan ke repositories

Mirip dengan menyimpan file yang telah diedit, komit mencatat perubahan pada satu atau lebih file di cabang Anda. Git memberikan ID unik, yang disebut SHA atau hash, yang mengidentifikasi:

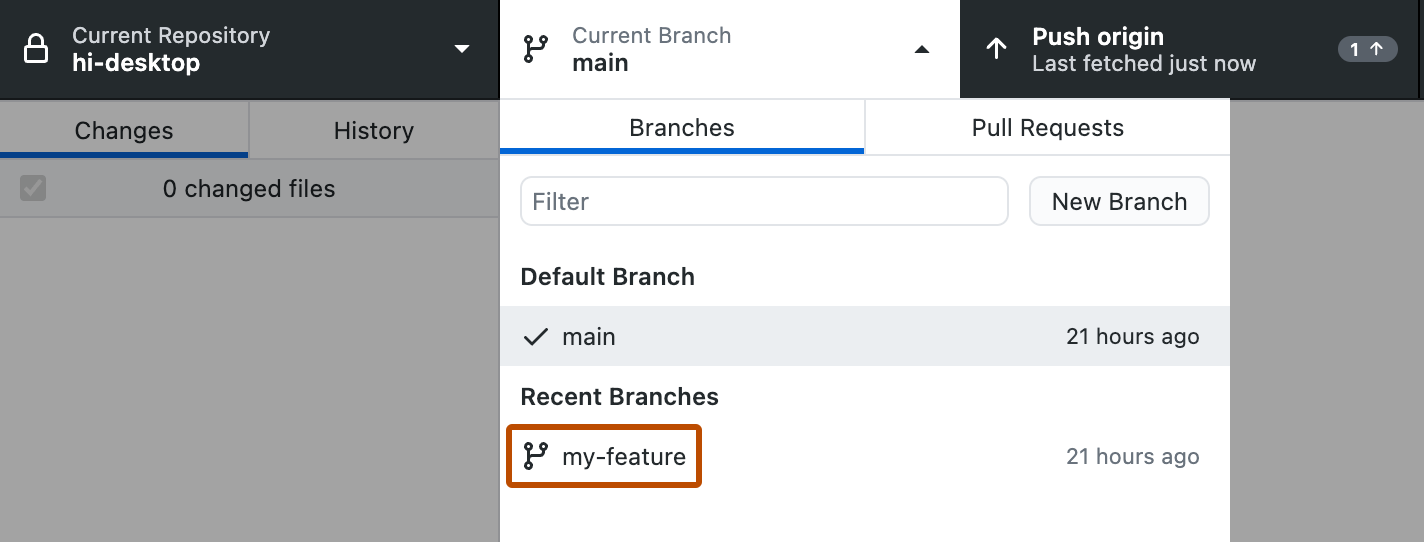
* Perubahan spesifik
* Kapan perubahan dilakukan
* Siapa yang menciptakan perubahan

Saat Anda membuat komit, Anda harus menyertakan pesan komit yang menjelaskan perubahan secara singkat. Anda juga dapat menambahkan rekan penulis pada setiap komitmen yang Anda kerjakan. Jika komit yang Anda buat di GitHub Desktop dikaitkan dengan akun yang salah di GitHub, perbarui alamat email di konfigurasi Git Anda menggunakan GitHub Desktop.

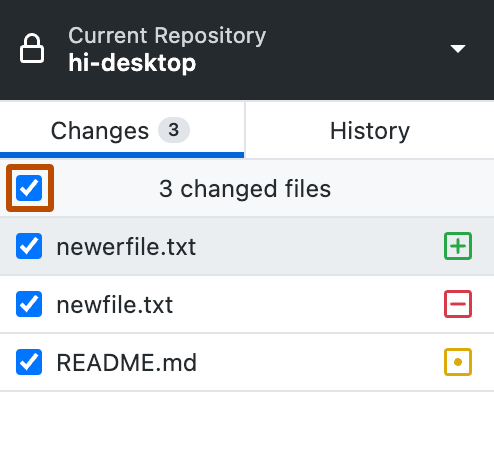
Administrator repositori dapat mengaktifkan kumpulan aturan untuk cabang guna menerapkan konvensi tertentu saat melakukan. Misalnya, kumpulan aturan mungkin memerlukan penandatanganan komit, atau nomor masalah harus direferensikan di awal pesan komit. GitHub Desktop akan menampilkan peringatan dan mencegah penerapan jika penerapan tidak mengikuti aturan.

Untuk langkah-langkah melakukan commit, adalah sebagai berikut :

1. Memilih atau membuat branch dan melakukan perubahan
2. Buat branch baru, atau pilih branch yang sudah ada dengan mengklik **Current Branch** Ini pada toolbar dan pilih branch dari daftar.

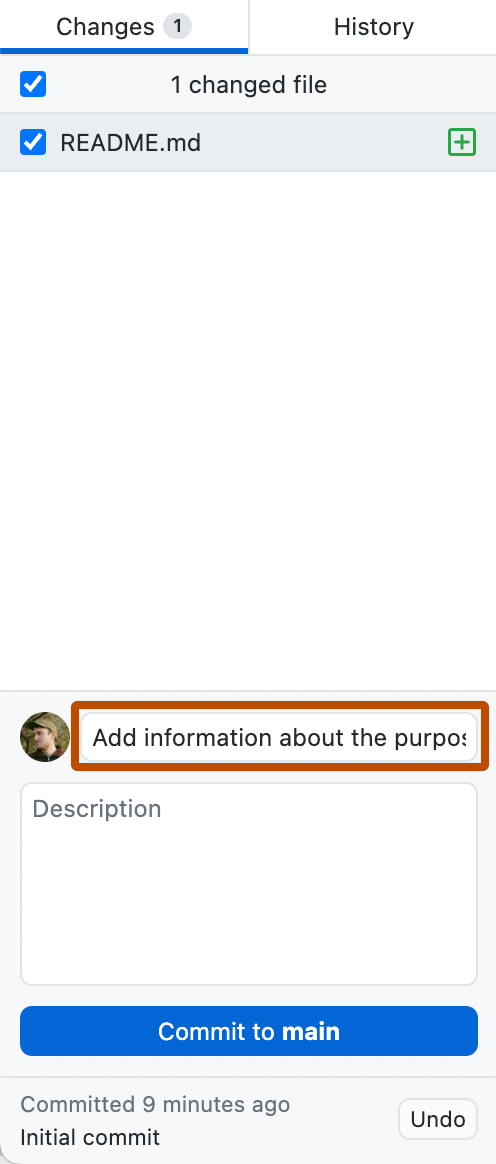


1. Gunakan editor teks favorit Anda untuk membuat perubahan yang diperlukan pada file Anda.
2. Pilih perubahan untuk disertakan dalam commit
3. Saat Anda membuat perubahan pada file di editor teks dan menyimpannya secara lokal, Anda juga akan melihat perubahan di GitHub Desktop pada beberapa menu berikut pada tab "Changes" di sidebar kiri:

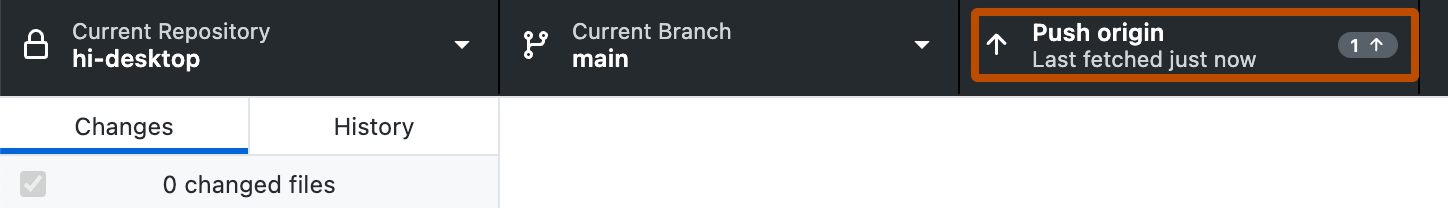


* 1. Ikon merah menunjukkan file yang dihapus.
  2. Ikon kuning menunjukkan file yang dimodifikasi.
  3. Ikon hijau menunjukkan file yang ditambahkan.
  4. Untuk mengakses perubahan yang disimpan, klik Perubahan yang Disimpan.
  5. Untuk menambahkan semua perubahan di semua file ke satu penerapan, tetap pilih kotak centang di bagian atas daftar.

1. Setelah perubahan telah dilakukan dan Anda memilih file untuk disertakan dalam komit Anda, tulis pesan komit Anda. Di bagian bawah daftar perubahan, di panel Summary, ketikkan pesan commit yang singkat dan bermakna. Secara opsional, Anda dapat menambahkan lebih banyak informasi tentang perubahan di panel Description.



1. Di bawah bidang Deskripsi, klik **Commit to BRANCH**.
2. Jika branch yang Anda coba komit protected, Github Desktop akan memberi peringatan kepada Anda. Berikut adalah beberapa solusi yang dapat dilakukan :
   1. Untuk memindahkan perubahan Anda, klik change branch.
   2. Untuk menerapkan perubahan Anda ke branch yang dilindungi, klik Commit to BRANCH.
3. Setelah commit berhasil, Anda dapat Meng-Push perubahan ke Github repository. Untuk mempush perubahan lokal Anda ke remote repositori, di menu repositori, klik push origin.



Catatan: GitHub Desktop akan menolak push jika melebihi batas tertentu.

* Push berisi file besar berukuran lebih dari 100 MiB.
* Ukuran total push lebih dari 2 GiB.

1. Jika ada komit diremote branch yang tidak Anda miliki di local branch, GitHub Desktop akan meminta Anda mengambil komit baru dari remote branch. Di panel "New Commit on Remote", klik Fetch.
2. REFERENSI

<https://git-scm.com/book/id/v1/Memulai-Git-Tentang-Version-Control>

<https://docs.github.com/en/desktop/contributing-and-collaborating-using-github-desktop/making-changes-in-a-branch/committing-and-reviewing-changes-to-your-project-in-github-desktop>

<https://docs.github.com/en/desktop/contributing-and-collaborating-using-github-desktop/making-changes-in-a-branch/committing-and-reviewing-changes-to-your-project-in-github-desktop>