

LAPORAN
“Langkah-langkah Pengoperasian pada GITHUB”



DISUSUN OLEH:
Fatmawati Harmain (531421075)

DOSEN PENGAMPUH:
Moh. Syafrin Tuloli, ST.,MT.

PROGRAM STUDI SISTEM INFORMASI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS NEGERI GORONTALO
2023

BAB 1 PENDAHULUAN

A. Version Control System (VCS)

Version Control System (VCS) adalah sebuah sistem yang melakukan *source code management* (SCM) untuk mengelola perubahan di setiap dokumen, program komputer, website, dan kumpulan pemrograman lainnya. Setiap melakukan revisi, *file* yang telah lalu tidak akan dibuang dan akan disimpan dengan nama yang berbeda. Sedangkan yang terbaru akan disimpan dengan nama; misal “Skripsi (Revisi Ke-2)”. Kegiatan tersebut akan dilakukan secara terus menerus hingga dalam satu folder skripsi terdapat *file* Ms. Word dengan kuantitas yang banyak. Tujuan tersebut tidak lain adalah untuk menyimpan *history* pekerjaan mahasiswa. Hingga akhirnya *file* terakhir selesai dengan nama “Skripsi Alhamdulillah Wis Udah”.

Konsep pekerjaan tersebut dianggap tidak efisien oleh banyak developer karena kapasitas penyimpanan akan membengkak. VCS disini berfungsi untuk membantu penyimpanan berupa *history* tanpa menyimpan *file* baru, yang tersimpan hanya perubahan data. Sehingga kapasitas penyimpanan *file* menjadi ringan. setiap perubahan data secara manual akan menghasilkan lebih banyak *file*. Sedangkan pada VCS mengusung konsep menyimpan rekaman perubahan dengan satu *file* saja.

B. GIT

Git merupakan *software* berbasis *Version Control System* (VCS) yang bertugas untuk mencatat perubahan seluruh *file* atau *repository* suatu *project*. Developer *software* biasa menggunakan Git untuk *distributed revision* (VCS terdistribusi), hal ini bertujuan untuk menyimpan *database* tidak hanya ke satu tempat. Namun semua orang yang terlibat dalam penyusunan kode dapat menyimpan *database* ini. Prosedur yang diterapkan ini dapat membantu antar divisi *project* untuk memantau dan menghubungkan (*merge*) antar ekstensi yang berbeda dengan mudah. Sehingga aplikasi yang dibuat oleh sebuah tim *project* dapat berfungsi tanpa menghubungkan secara manual.

Terdapat istilah *commit* pada Git yang berfungsi untuk menyimpan riwayat perubahan data pada *file*. Melalui *commit*, developer dapat kembali ke *source code* sebelumnya dengan istilah *checkout*. Untuk mengoperasikan Git, kamu perlu menginstall *software* terlebih dahulu sehingga

pekerjaan ini dapat dilakukan secara *offline* (tidak terkoneksi internet). *Software* ini juga tersedia secara gratis melalui web unduhan resminya di Git Downloading.

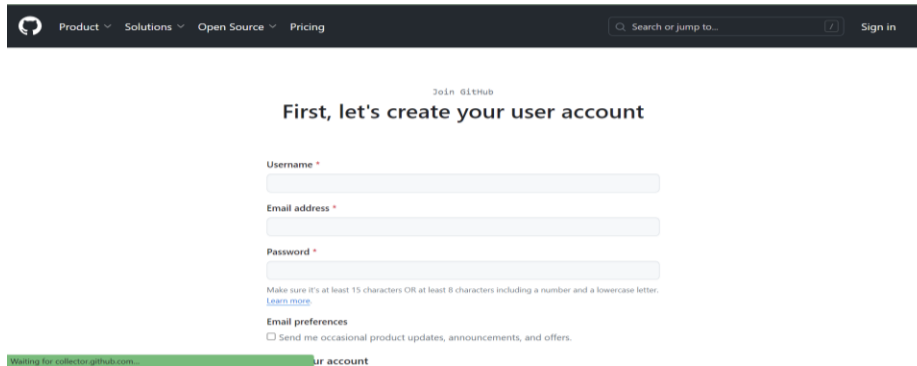
C. Github

GitHub merupakan layanan *cloud* yang berguna untuk menyimpan dan mengelola sebuah *project* yang dinamakan *repository* (repo git). Cara kerja pada GitHub harus terkoneksi pada internet sehingga tidak perlu meng-*install* sebuah *software* ke dalam perangkat keras. Hal ini memberikan keringanan penyimpanan komputer yang kita gunakan karena *file project* tersimpan oleh *cloud* GitHub. Konsep kerja GitHub pada dasarnya sama dengan Git yaitu dapat menulis *source code* secara individu atau tim. *User interface* yang tersedia pada GitHub lebih menarik dan mudah dipahami oleh pengguna awal. Pekerjaan secara tim, pengguna juga bisa melihat siapa penulis kode dan tanggal berapa kode tersebut dibuat.

Terdapat fitur lain pada GitHub yaitu kita dapat membaca berbagai *blog* dan *feed* yang dibuat oleh sesama pengguna. Hal ini dimanfaatkan oleh pengguna seluruh dunia untuk saling berbagi ide pemrograman dan berdiskusi dalam menyelesaikan masalah. Tentunya postingan yang ada pada GitHub berkaitan dengan pemrograman. Sehingga Github telah menjadi forum diskusi para *programmer* seperti halnya media sosial. Semenjak GitHub diakuisisi oleh Microsoft di tahun 2018, platform ini berkembang semakin baik dan unggul. Sehingga mayoritas *programmer* lebih mengenal GitHub dalam program VCS daripada pesaingnya seperti GitLab dan Atlassian BitBucket.

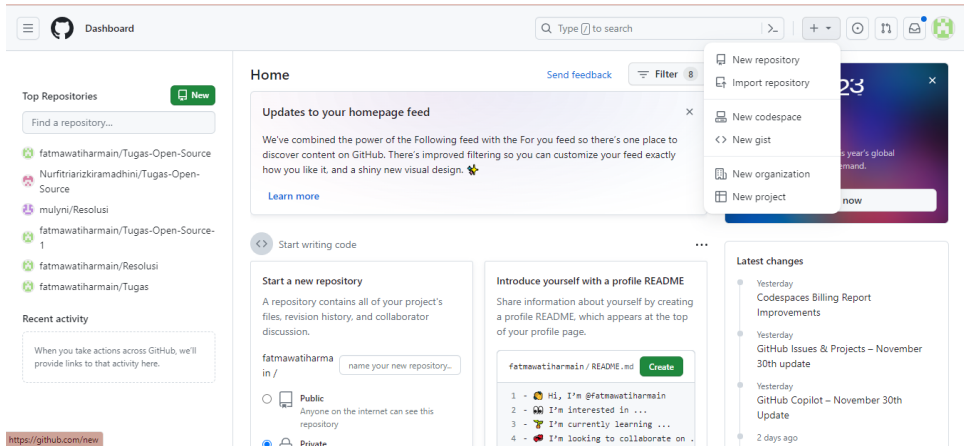
BAB II PEMBAHASAN

1. Membuat akun di GitHub



The screenshot shows the GitHub sign-up page. At the top, there's a navigation bar with links for Product, Solutions, Open Source, and Pricing. Below this, a search bar and a 'Sign in' link are visible. The main heading is 'First, let's create your user account'. The form includes fields for Username, Email address, and Password. A note specifies password requirements: 'Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter.' There's a link to 'Learn more' and a checkbox for 'Email preferences' with the text 'Send me occasional product updates, announcements, and offers.'

2. Buat Repository dengan mengklik fitur +



3. Tulis nama repository, menuliskan description dan tampilkan kepada publik lalu mencentang add a README file

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Owner * fatmawatiharmain / Repository name * Tugas Open Source

✓ Your new repository will be created as **Tugas-Open-Source-**.
The repository name can only contain ASCII letters, digits, and the characters -, ., and _.

Great repository names are short and memorable. Need inspiration? How about **miniature-octo-parakeet** ?

Description (optional)

Repository untuk menyimpan tugas open source

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

Setelah selesai klik Create repository

description (optional)

Repository untuk menyimpan tugas open source

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:

☐ Add a README file
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

You are creating a public repository in your personal account.

Create repository

4. Buat file baru klik add file kemudian Create new file

fatmawatiharmain / Tugas-Open-Source

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Tugas-Open-Source Public

Pin Unwatch 1 Fork 2 Star 0

main 1 branch 0 tags

Go to file Add file Code

fatmawatiharmain Merge pull request #5 from fatmawatiharmain/Coba-Fitur-2

15 commits

Initial commit yesterday

Repository untuk menyimpan tugas open source

Readme Activity

5. Lakukan pengeditan pada file

fatmawatiharmain / Tugas-Open-Source

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

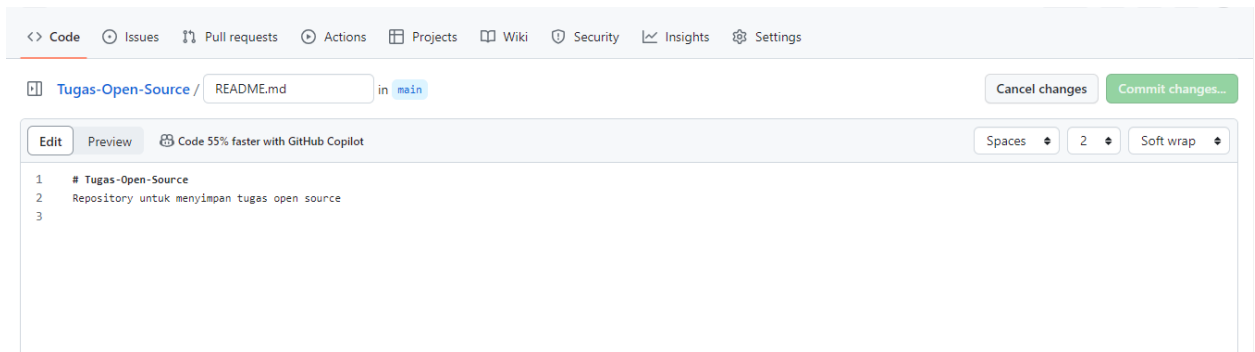
main Tugas-Open-Source / Resolusi.txt

fatmawatiharmain Menghapus dan menambahkan Resolusi.txt bcf4428 · 19 hours ago History

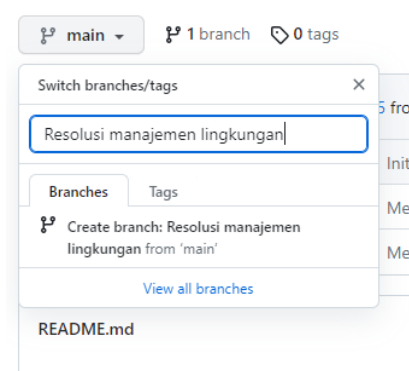
Code Blame 5 lines (5 loc) · 170 Bytes Code 55% faster with GitHub Copilot

```
1 Resolusi memanaajemen keuangan
2 - membuat anggaran bulanan
3 - mengurangi pengeluaran tidak perlu
4 - mengembangkan darurat keuangan
5 - menetapkan Tujuan Keuangan Jangka Panjang
```

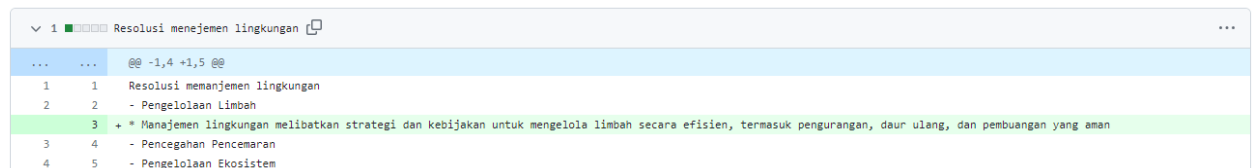
6. Edit README, kemudian klik Commit changes



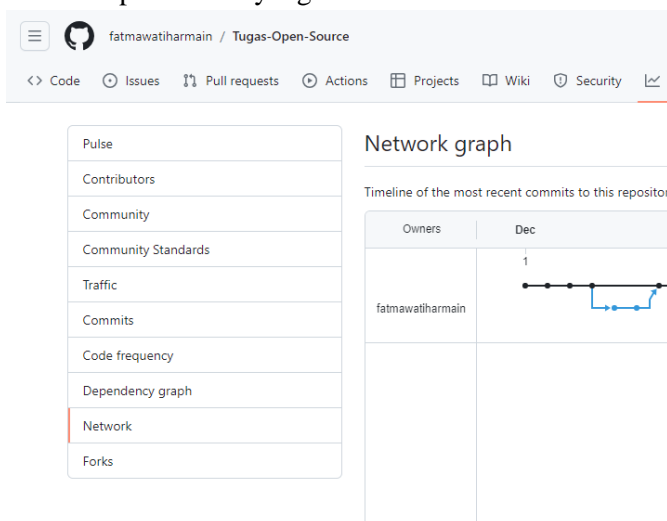
7. Buat file lalu simpan sebagai branch baru



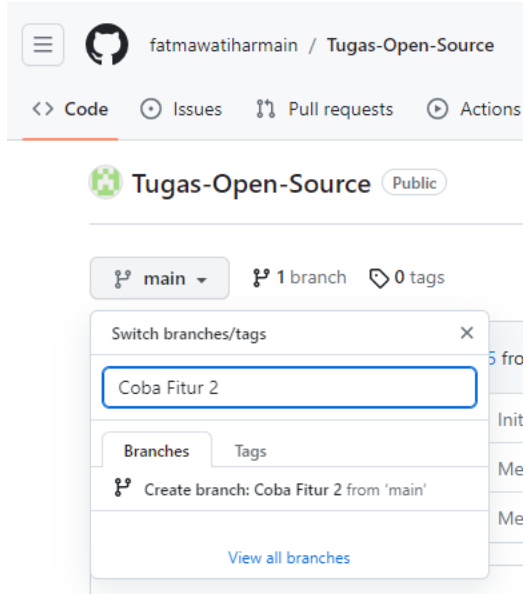
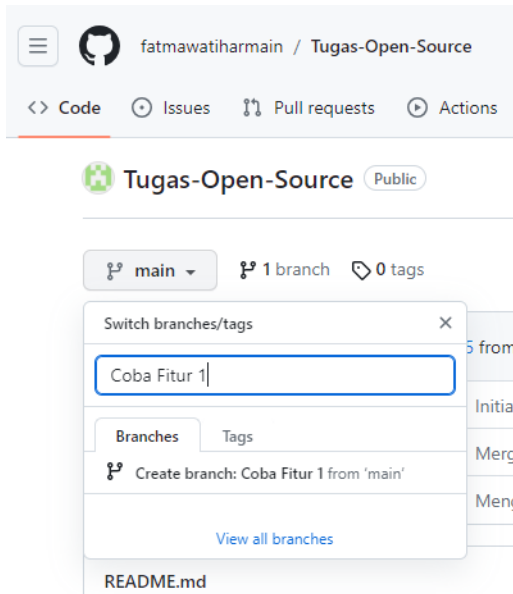
8. Setelah itu lakukan pengeditan pada branch



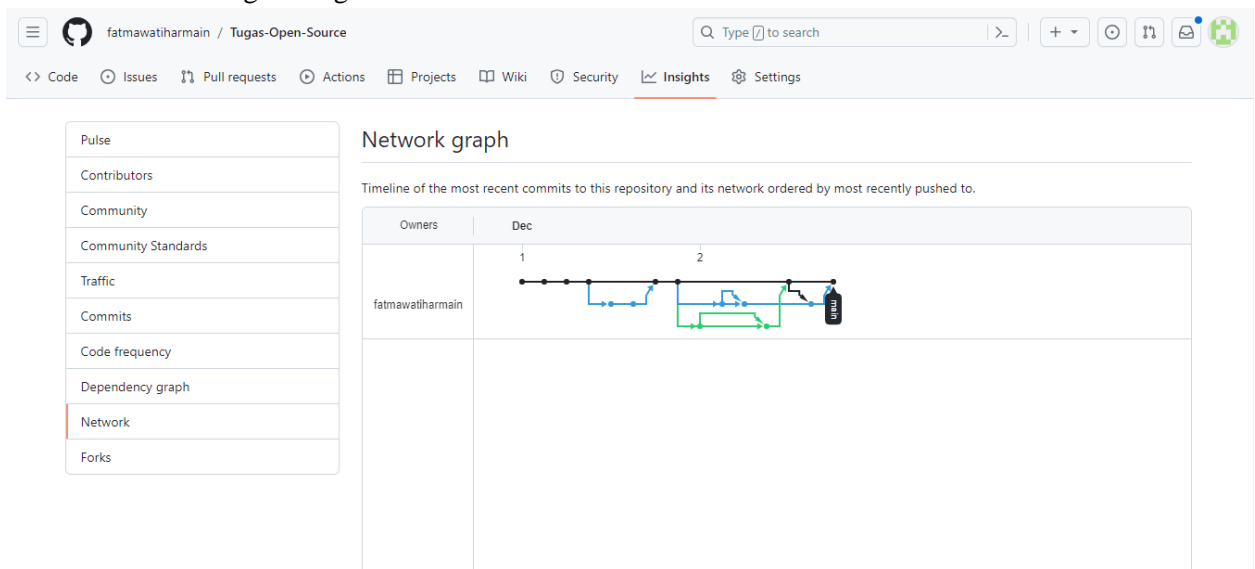
9. Lakukan pull request pada file baru ke branch master lalu confirm marge dan branch baru. Setelah selesai hapus branch yang telah di buat



10. Membuat dua branch baru kemudian bagikan ke teman untuk dilakukan perubahan



11. Tunggu full request dari teman
12. Setelah menerima dua full request dari teman, terima salah satunya agar bergabung pada branch master
13. Terima full request yang kedua, maka akan terjadi konflik
14. Selesaikan konflik agar merge bisa dilakukan.



BAB III

PENUTUP

1.3 KESIMPULAN

Dari hasil penggunaan Git dan GitHub selama periode tertentu, dapat disimpulkan bahwa kedua alat ini memberikan dampak positif pada pengembangan perangkat lunak dan kolaborasi tim. Git, sebagai sistem kontrol versi, memberikan fleksibilitas dan keamanan dalam melacak perubahan kode, sedangkan GitHub sebagai platform kolaborasi membantu meningkatkan transparansi, kualitas, dan manajemen proyek secara keseluruhan.

Selama penggunaan, kami mencapai peningkatan efisiensi dalam manajemen perubahan kode dan integrasi kontribusi tim. Fitur-fitur GitHub, seperti pull requests, code review, dan issue tracking, telah memfasilitasi dialog konstruktif dan meningkatkan kualitas kode. Meskipun demikian, kami menyadari adanya tantangan terutama terkait dengan manajemen merge conflicts dan strategi branching yang lebih baik.

1.3 SARAN

Dalam penyusunan Laporan ini penyusun menyadari masih banyak kekurangan, perlunya tambahan referensi yang lebih banyak lagi. Maka dari itu penyusun sangat membutuhkan kirit dan saran yang sifatnya membangun untuk kesempurnaan laporan ini. Serta semoga laporan ini dapat bermanfaat bagi penyusun dan pembaca dapat menambah pengetahuan mengenai penggunaan Git dan Github.