



UNIVERSITAS SEMARANG  
FAKULTAS TEKNOLOGI INFORMASI DAN KOMUNIKASI  
TEKNIK INFORMATIKA

---

# Open Source Systems

---

Modul Praktikum Mahasiswa

*Oleh:*  
Alauddin Maulana Hirzan, S. Kom., M. Kom  
NIDN. 0607069401

# Daftar Isi

<b>Pendahuluan</b>	<b>2</b>
0.1 Menenal <i>Version Control System</i> dan GIT . . . . .	2
0.2 Keamanan dan Integritas GIT . . . . .	3
0.3 Cara Kerja GIT . . . . .	3
0.4 Respository Penyimpanan . . . . .	4
<b>Persiapan Praktikum</b>	<b>5</b>
0.5 Perangkat Keras . . . . .	5
0.6 Perangkat Lunak . . . . .	5
0.7 Instalasi . . . . .	6
0.8 Akun GitHub . . . . .	7
0.8.1 Daftar Akun . . . . .	7
0.8.2 <i>Personal Token Access</i> . . . . .	7
<b>1 Praktikum 1</b>	<b>10</b>
1.1 Membuat Folder GIT . . . . .	10
1.2 Tutorial . . . . .	10
<b>2 Praktikum 2</b>	<b>16</b>
2.1 Kloning dan Update Repositori . . . . .	16
2.2 Tutorial . . . . .	16
2.3 Tugas Mandiri . . . . .	18
<b>3 Praktikum 3</b>	<b>20</b>
3.1 Percabangan Repositori . . . . .	20
3.2 Tutorial . . . . .	20
3.3 Tugas Mandiri . . . . .	22
<b>4 Praktikum 4</b>	<b>23</b>
4.1 Penggabungan dan Penghapusan Cabang Repositori . . . . .	23
4.2 Tutorial . . . . .	23
<b>5 Praktikum 5</b>	<b>26</b>
5.1 <i>Undo</i> dan <i>Stash</i> . . . . .	26
5.2 Tutorial . . . . .	26
<b>6 Praktikum 6</b>	<b>29</b>
6.1 Tagging . . . . .	29

6.2	Tutorial . . . . .	29
<b>7</b>	<b>Praktikum 7</b>	<b>33</b>
7.1	Commit Squashing . . . . .	33
7.2	Tutorial . . . . .	33
<b>8</b>	<b>Praktikum 8</b>	<b>37</b>
8.1	Reset Commit . . . . .	37
8.2	Tutorial . . . . .	37
<b>9</b>	<b>Praktikum 9</b>	<b>39</b>
9.1	Instalasi Linux Ubuntu . . . . .	39
9.2	Tutorial . . . . .	39
<b>10</b>	<b>Praktikum 10</b>	<b>48</b>
10.1	Operasi Perintah Linux Ubuntu . . . . .	48
10.2	Tutorial . . . . .	48

# Daftar Gambar

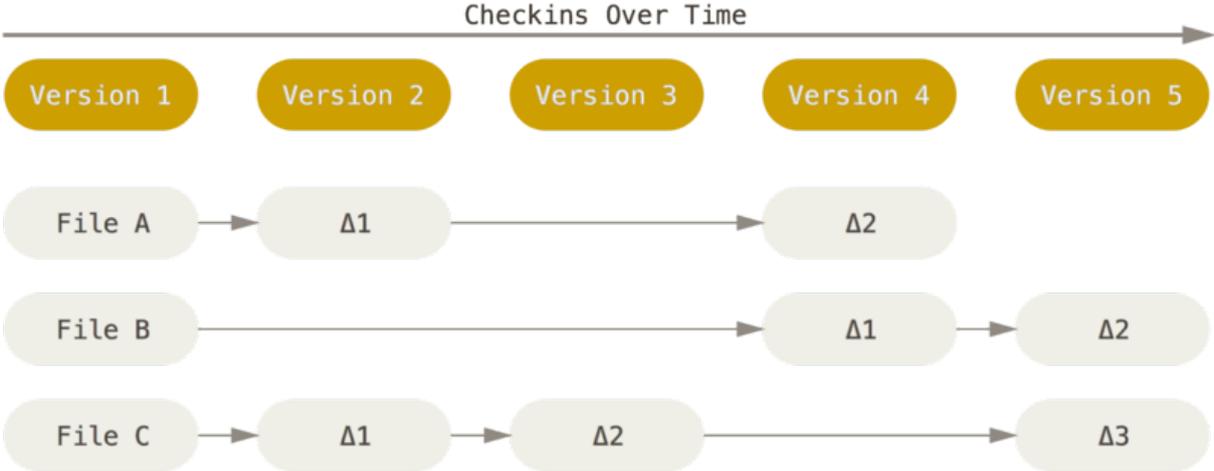
- 1 *Delta-based Control* . . . . . 2
- 2 *Snapshot-based Control* . . . . . 3
- 3 Tiga Tahapan GIT . . . . . 4
- 4 Tahap Pemilihan Editor . . . . . 6
- 5 Git Bash . . . . . 6

# Pendahuluan

## 0.1 Mengenal *Version Control System* dan GIT

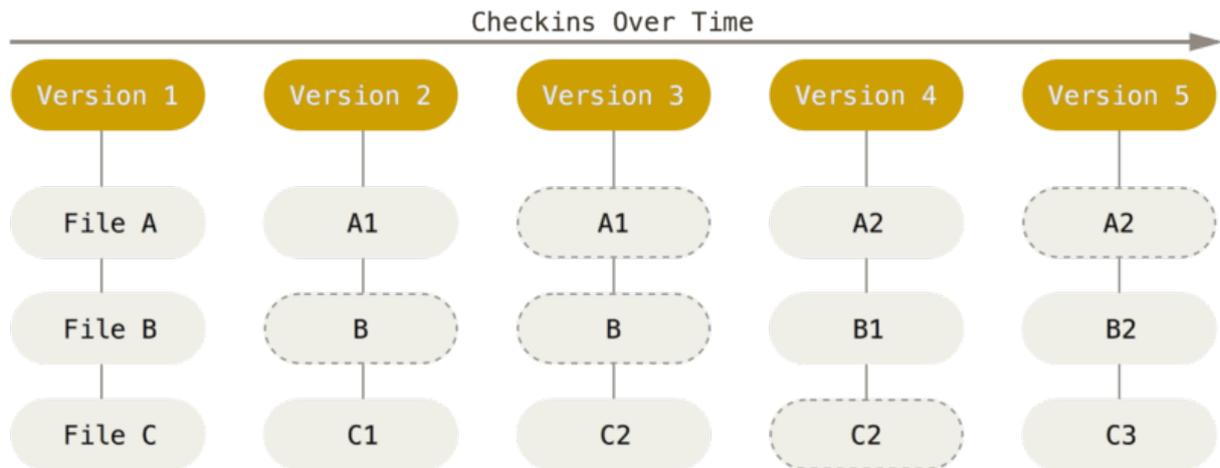
*Version Control* di dalam bidang *Software Engineering* merupakan sebuah teknik yang dilakukan untuk mengendalikan versi yang akan dirilis dalam pengembangan aplikasi. Sehingga dengan adanya teknik ini, semua perubahan yang terjadi di dalam aplikasi tersebut dapat tercatat dan termonitor dengan baik. Terdapat beberapa *software* yang dapat digunakan seperti GIT dan SVN. Keduanya memiliki tugas yang sama namun cara mencapai tugas itu berbeda. Dalam praktikum kali ini, mata kuliah ini menganjurkan untuk menggunakan GIT sebagai alat pengendalian versi.

Apa itu GIT? GIT adalah aplikasi pengendali versi yang menyimpan dan menganalisis informasi dengan cara yang berbeda dengan yang lainnya. GIT menerapkan konsep *content-aware* sehingga dia tidak akan menambahkan, mengubah, mengunggah file yang kosong atau tidak mengalami perubahan. Ciri khas ini dimiliki oleh GIT namun tidak dimiliki oleh aplikasi lain. Aplikasi lain seperti CSV, Subversion, Perforce, Bazaar menggunakan pendekatan *time-based changes* atau melihat perubahan file dari waktu ke waktu. Atau bisa dikenal sebagai *delta-based version control*. Berikut ini adalah perbandingan antara *Delta-based* dengan *Snapshot-based* GIT:



Gambar 1: *Delta-based Control*

Model yang ditunjukkan oleh Gambar 1 adalah cara yang digunakan oleh aplikasi lainnya. Dengan membandingkan perbedaan antara satu file dengan yang lainnya (*diff* atau *delta*), VCS akan mengubah file tersebut. Jika di file A terdapat perubahan kedua, maka perubahan tersebut akan diunggah. Jika tidak, maka file akan dibiarkan.



Gambar 2: *Snapshot-based Control*

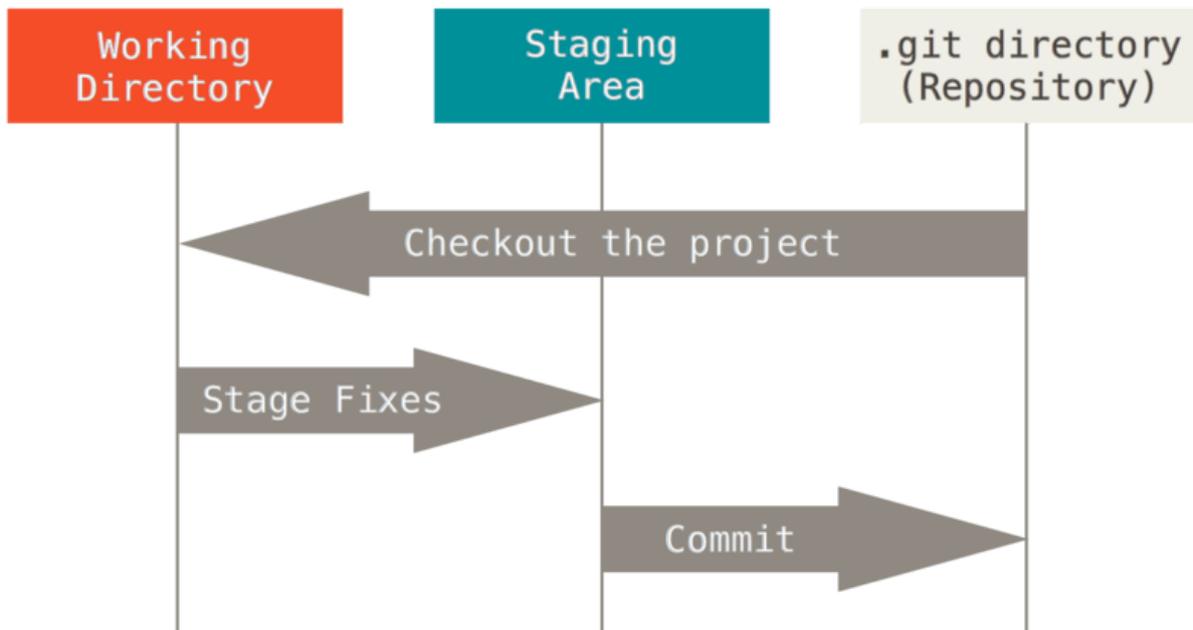
Sedangkan GIT yang menggunakan metode *snapshot*, menggunakan cara yang berbeda dalam mengendalikan versi dari masing-masing file. Dengan menggunakan GIT setiap perintah *commit* yang diberikan, GIT akan merekam status file yang ada di folder tersebut dan menunggunghannya. Sehingga disetiap sesi *commit* versi, file-file yang dibutuhkan tetap ada. Sehingga muncul fitur seperti *branching* yang memungkinkan perubahan tanpa menyentuh file utama.

## 0.2 Keamanan dan Integritas GIT

GIT memiliki perlindungan serta integritas yang sangat tinggi. Pengguna dapat mengunggah file mereka melalui jalur yang aman melalui *Secure Shell / SSH*. Selain itu, GIT juga melakukan monitoring kualitas atau perubahan file berdasarkan *checksum hash* menggunakan SHA-1. Ditambah, untuk memastikan orang-orang yang mengunggah adalah orang asli bukan penuri, GIT juga mampu menggunakan kunci kriptografi sebagai identifikasi *committer*.

## 0.3 Cara Kerja GIT

GIT memiliki cara kerja yang sangat sederhana, hanya menggunakan **Tiga Tahapan** saja. Berikut adalah ilustrasi cara kerja GIT:



Gambar 3: Tiga Tahapan GIT

Menurut Gambar 3, GIT dimulai dengan melakukan *checkout* pada proyek yang ada di dalam folder. Hal ini bisa dicapat dengan melakukan perintah *git init* untuk memulai GIT, atau dengan mengambil proyek yang sudah ada melalui perintah *git clone*. Folder *.git* ini berisi informasi-informasi penting berkaitan dengan proyek tersebut. Meski aman untuk dihapus, namun rekam jejak file termasuk link upload akan hilang.

Setelah melakukan inialisasi atau klon, pengguna dapat melakukan perubahan file tersebut. Segala bentuk perubahan inilah yang nantinya akan masuk ke dalam *Staging Area* melalui perintah *git add -A*. Semua file yang masuk di tahap ini kemudian diberi label menggunakan perintah *git commit* sehingga memudahkan pengguna dalam membaca versi yang diunggah. *Commit* ini disimpan dalam folder *.git*. File-file siap diunggah ke *Repository*

## 0.4 Respository Penyimpanan

*Repository* merupakan tempat penyimpanan kode-kode atau gambar atau file lain (dengan batas ukuran tertentu) secara daring sehingga bisa diakses oleh orang lain. Hampir mirip dengan *Cloud Storage* pada biasanya, tetapi hanya bisa diakses oleh pemilik atau *contributor* yang sudah dipercaya (dan diberi akses). Di dunia internet ada beberapa repository yang bisa dipakai seperti <https://github.com> atau <https://gitlab.com>. Keduanya memiliki tugas yang sama sebagai repository, namun memiliki kebijakan yang berbeda dalam hal prinsip *open source*. Dalam hal fitur, *GitHub* lebih banyak kebebasan dibandingkan *GitLab*.

# Persiapan Praktikum

Agar praktikum dapat berjalan dengan lancar, mahasiswa diwajibkan memenuhi persyaratan berikut baik dalam bentuk perangkat keras maupun lunak:

## 0.5 Perangkat Keras

GIT SCM tidak memerlukan spesifikasi yang tinggi untuk dioperasikan sehingga dapat digunakan diperangkat sekelas *Raspberry Pi* dengan spesifikasi paling rendah. Namun wajib memiliki

- Layar Monitor
- Mouse
- Keyboard

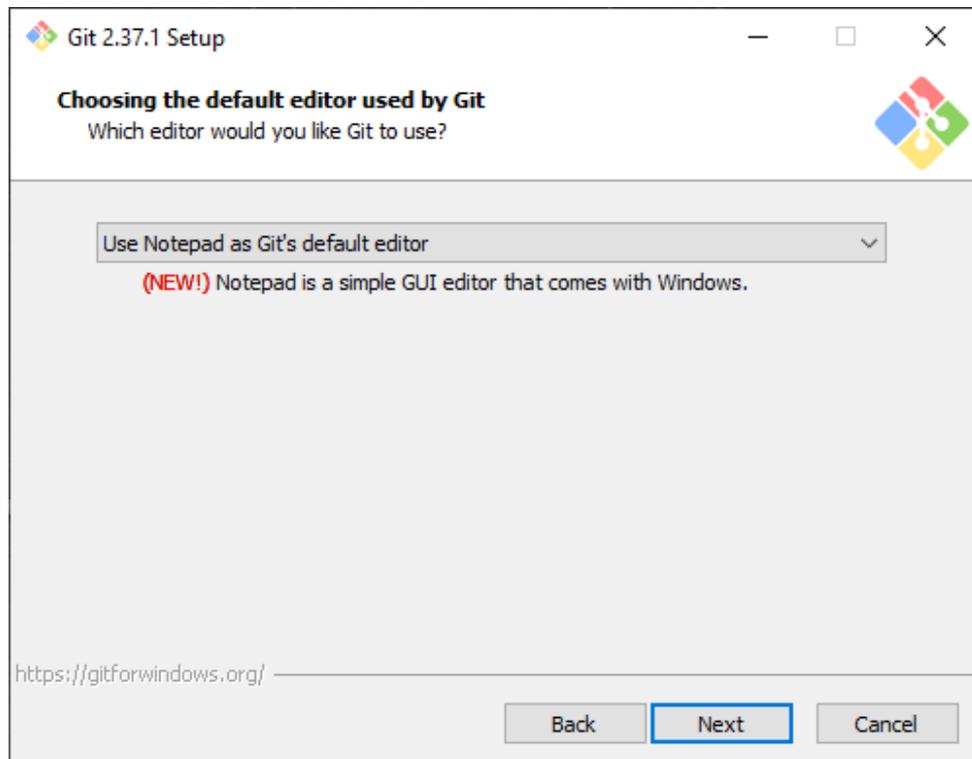
## 0.6 Perangkat Lunak

Perangkat lunak berikut ini wajib diinstall oleh mahasiswa demi lancarnya praktikum:

- Praktikum 1 - 8
  - Pengguna Windows
    1. Git-SCM via <https://git-scm.com/> (Gratis)
    2. GPG4Win via <https://gpg4win.org/download.html> (Gratis, pilih \$0 ketika mengunduh)
    3. Notepad (sudah termasuk)
  - Pengguna Linux
    1. git
    2. gpg
    3. nano atau vim atau editor lain
- Praktikum 9 - 10
  - VirtualBox

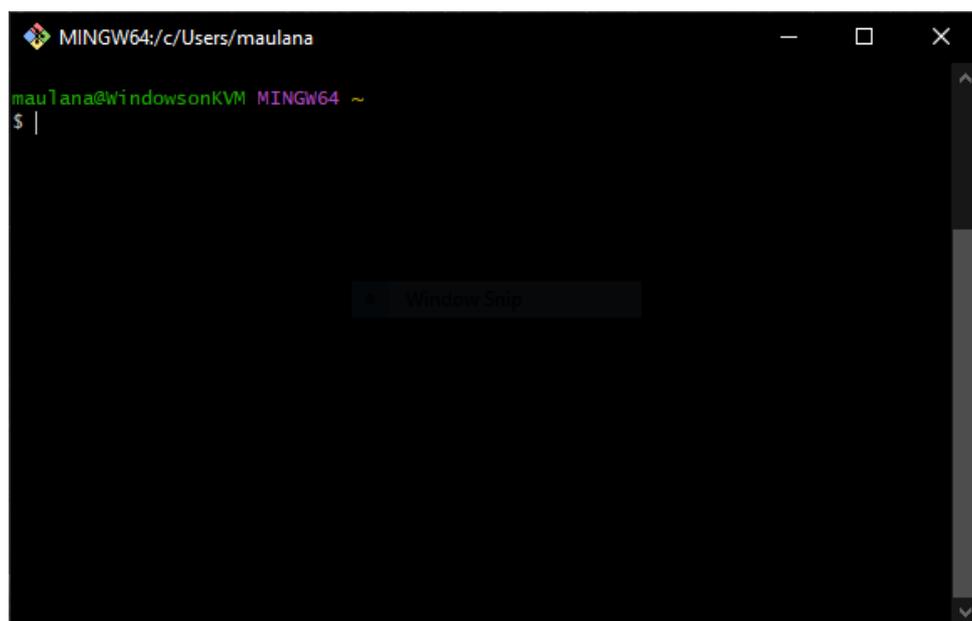
## 0.7 Instalasi

Dalam melakukan instalasi GIT, ada satu langkah yang harus diperhatikan. Langkah ini akan menentukan editor apa yang akan dipakai git di Windows.



Gambar 4: Tahap Pemilihan Editor

Pastikan ketika memasang GIT-SCM untuk memilih *Notepad* sebagai editor default Windows. Bagian akhir ketika praktikum, mahasiswa sebaiknya menggunakan *Git Bash* dan bukan menggunakan CMD maupun Powershell seperti Gambar 5.

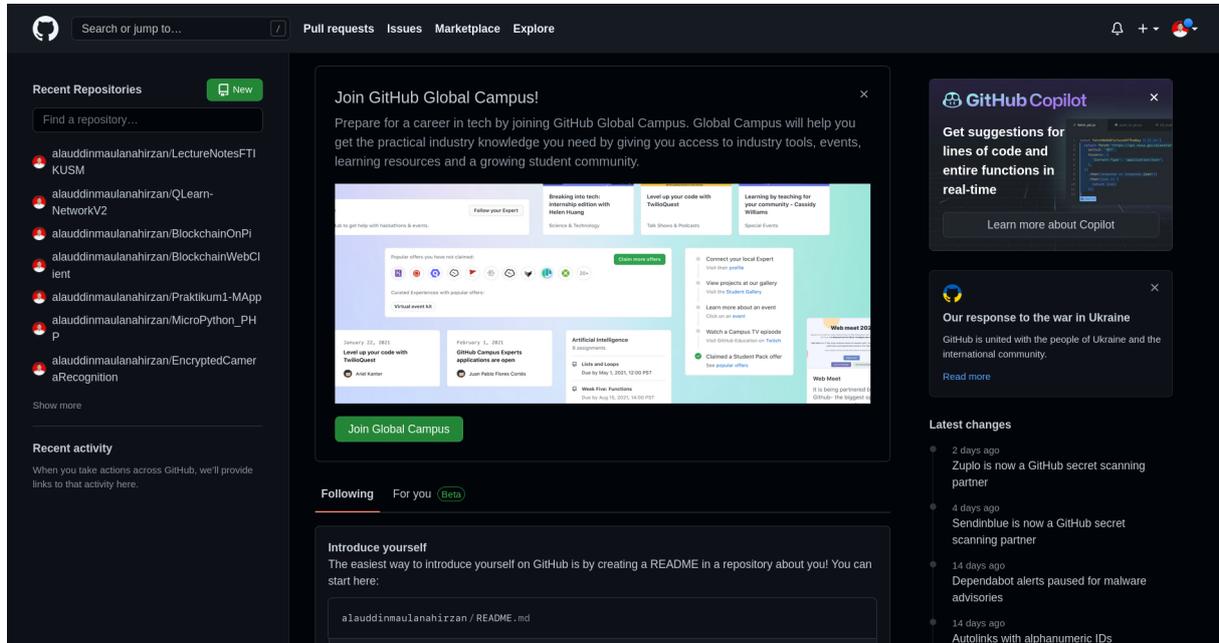


Gambar 5: Git Bash

## 0.8 Akun GitHub

### 0.8.1 Daftar Akun

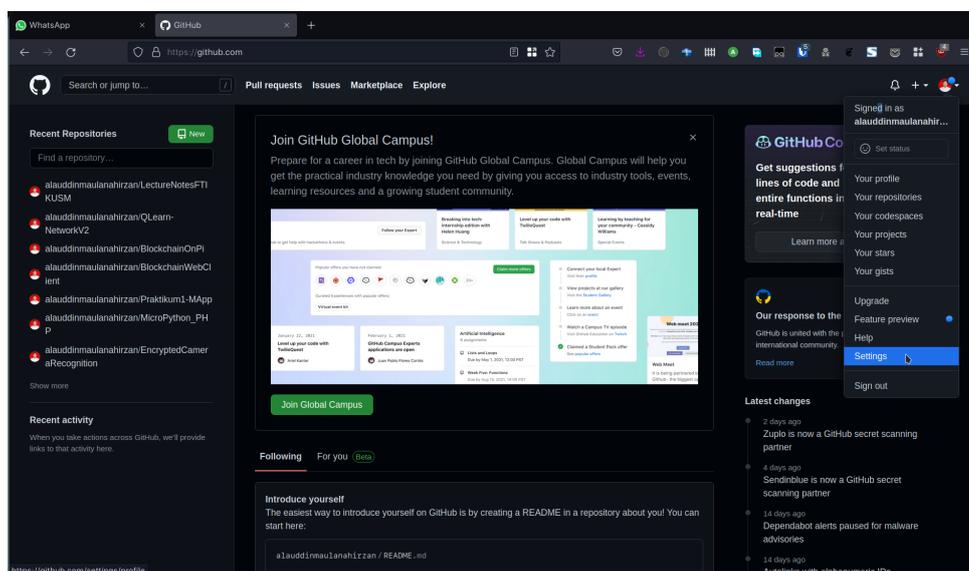
Tahapan ini sangat penting karena git tidak bisa beroperasi tanpa akun di *GitHub*. Buka website <https://github.com/signup>. Isi semua form yang diperlukan, dan pastikan untuk **konfirmasi e-mail** agar akun GitHub dapat digunakan.



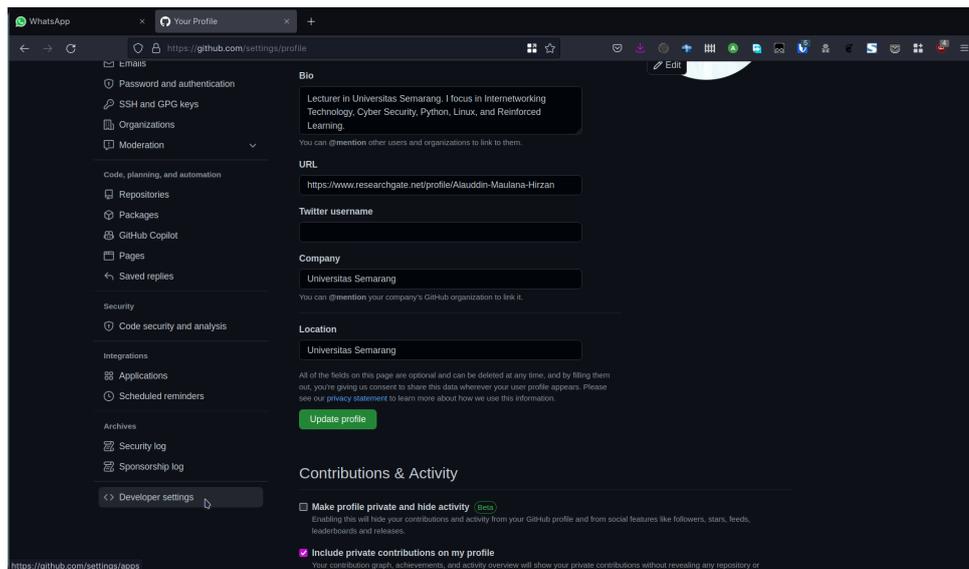
### 0.8.2 Personal Token Access

Tahap ini sangat penting karena demi keamanan pemilik GitHub untuk menggunakan *Token* ketika mengunggah kodenya. Cukup dengan cara mudah:

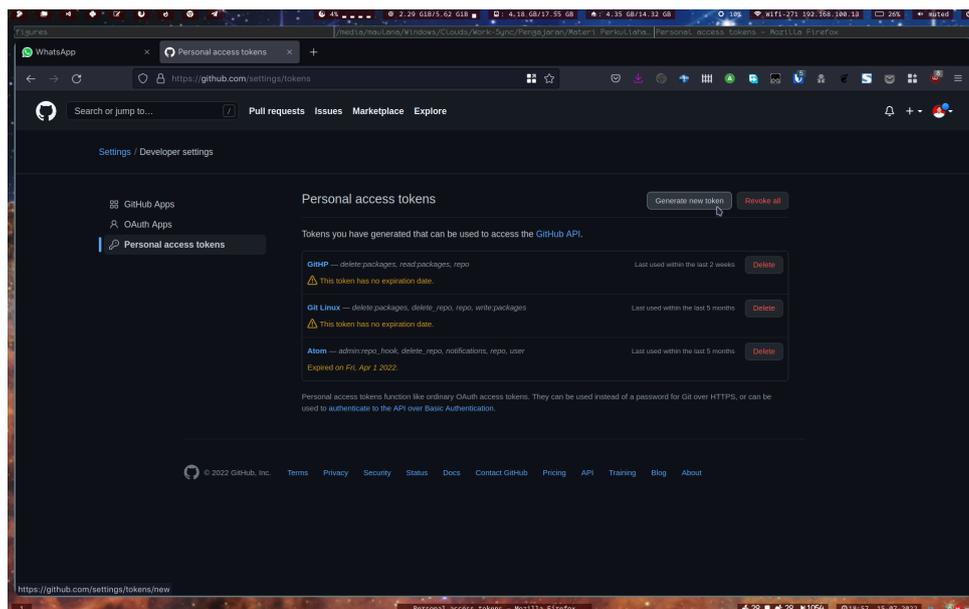
1. Klik **Foto Profil** di **Pojok Kanan Atas**, Pilih **Settings**



2. Klik **Developer Settings** di bagian bawah kiri

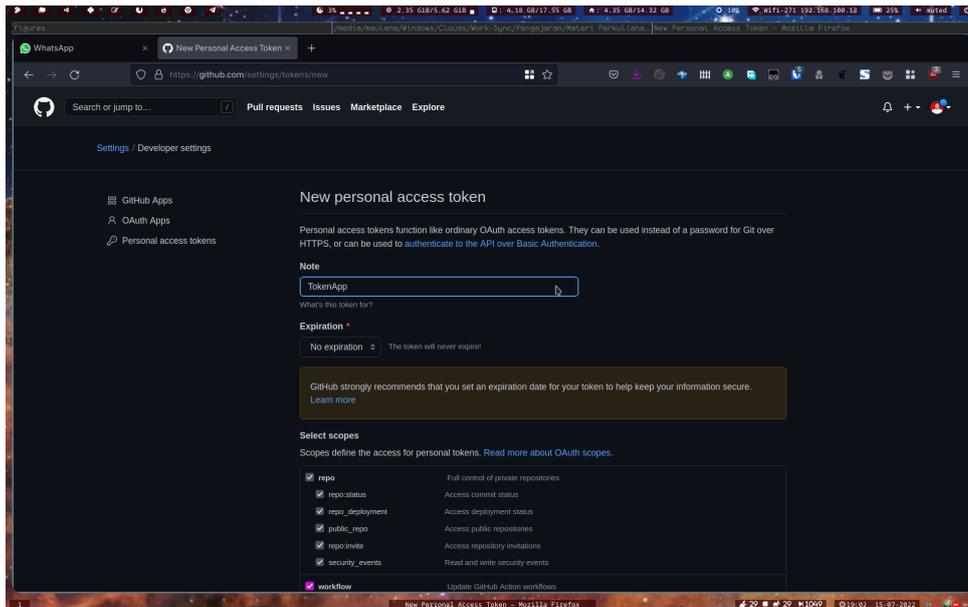


3. Pilih **Personal access token**, lalu klik **Generate new token**

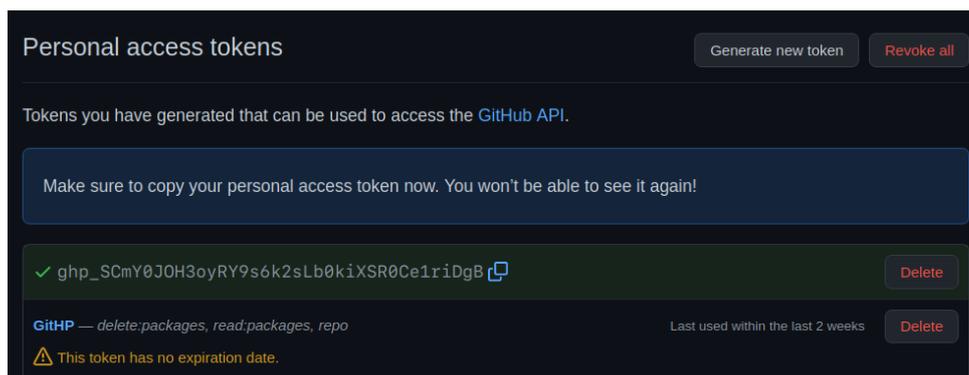


4. **GitHub** akan meminta **Password**. Masukkan **Password** akun.

5. **Beri nama token**. Ubah **Expiration** ke **No expiration** dan centang **semua scope**. Jika sudah klik **Generate token** yang ada di bawah halaman.



6. GitHub akan membuat kode unik yang harus disimpan dan tidak boleh dimasukkan ke folder git. Simpan baik baik.



7. Kopi kode tersebut karena tidak akan muncul lagi.

# Bab 1

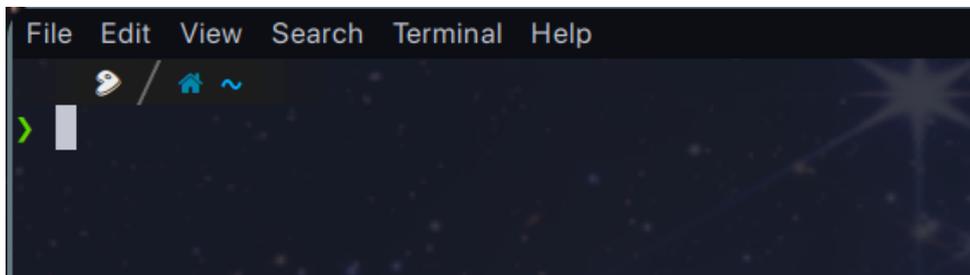
## Praktikum 1

### 1.1 Membuat Folder GIT

Di bagian ini mahasiswa diajarkan bagaimana melakukan pembuatan projek git, menambahkan file ke git, melakukan pelabelan, dan mengunggah ke repositori. Mahasiswa diwajibkan untuk menyelesaikan **Persiapan Praktikum** sebelum masuk ke tahapan ini.

### 1.2 Tutorial

1. Buka **Git Bash** jika menggunakan **Windows** atau **Terminal** jika menggunakan **Linux**. Keduanya sama-sama memiliki antarmuka dan perintah yang sama.



2. Sebelum masuk ke pembuatan projek, sebaiknya aplikasi **git** dikonfigurasi terlebih dahulu dengan perintah berikut:

```
Perintah Terminal  
git config --global user.name "Your Name"  
git config --global user.email "youremail@yourdomain.com"  
(Keterangan : Mengeset Nama dan E-Mail dalam config)
```

3. Kemudian cek dengan perintah:

```
Perintah Terminal  
git config --list  
(Keterangan : Melihat daftar konfigurasi)
```

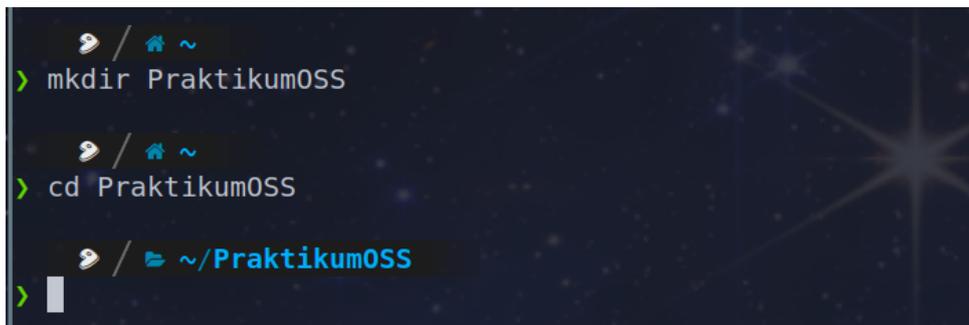
4. Konfigurasi ini dapat dilihat di folder **%USERPROFILE%/.gitconfig** (**Windows**) atau di **\$HOME/.gitconfig** (**Linux**)

```
GNU nano 6.2
[user]
name = Alauddin Maulana Hirzan
email = maulanahirzan@proton.me
```

5. Buatlah sebuah folder dan masuk ke dalam folder tersebut dengan menggunakan perintah berikut:

**Perintah Terminal**

```
mkdir PraktikumOSS
cd PraktikumOSS
```

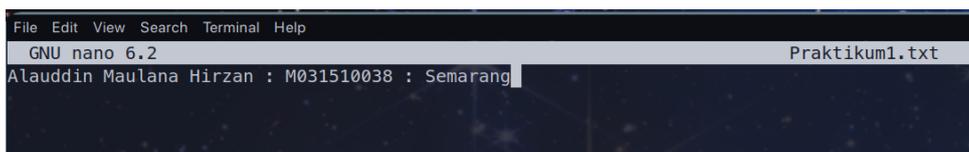


```
> mkdir PraktikumOSS
> cd PraktikumOSS
>
```

6. Buatlah **Sebuah File** dengan menggunakan **Notepad** atau **Editor Lain** dengan isi sebagai berikut, dan simpan sebagai **Praktikum1.txt** di dalam folder **PraktikumOSS**.

**Perintah Terminal**

```
notepad Praktikum1.txt (Hanya Windows)
nano Praktikum1.txt (Hanya Linux)
-----
<Nama> : <NIM> : <Asal> (Isi Praktikum1.txt)
```



```
File Edit View Search Terminal Help
GNU nano 6.2 Praktikum1.txt
Alauddin Maulana Hirzan : M031510038 : Semarang
```

7. Setelah file tersimpan dengan baik, waktunya membuat projek di dalam folder tersebut. Gunakan perintah berikut untuk mengubah folder tersebut menjadi folder GIT:

**Perintah Terminal**

```
git init
(Keterangan : Mengubah Folder biasa menjadi Folder GIT)
```

```
~/PraktikumOSS
> git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/maulana/PraktikumOSS/.git/

~/PraktikumOSS / on git master ?1
>
```

8. Folder ini sudah menjadi folder GIT, namun belum bisa mengunggah ke repositori. Untuk dapat mengunggah, gunakan perintah berikut:

**Perintah Terminal**

```
git add -Av
(Keterangan : Tambahkan semua file dan tampilkan log)
```

```
~/PraktikumOSS / on git master ?1
> git add -Av
add 'Praktikum1.txt'
```

9. Berikutnya adalah memberikan label atas file-file yang ingin diunggah. Kita sebut ini sebagai **COMMIT**. Gunakan perintah berikut untuk melabeli unggahan (bukan FILE):

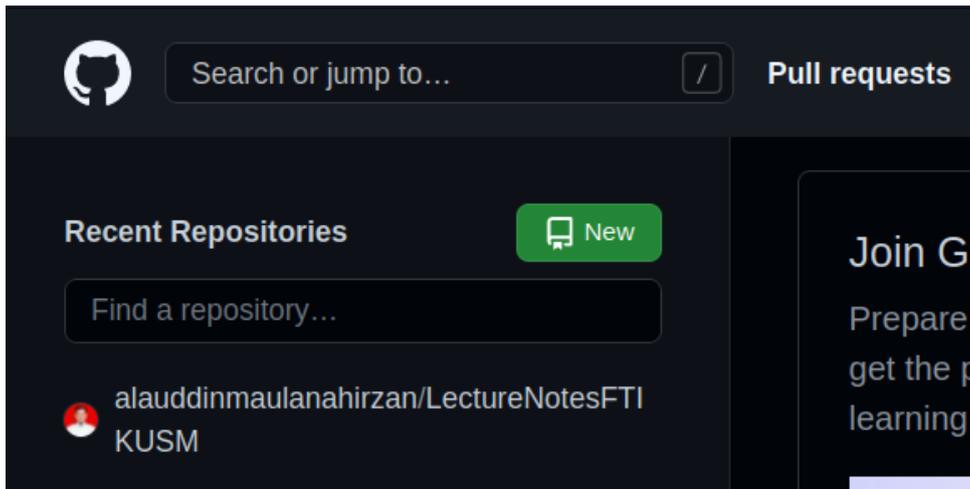
**Perintah Terminal**

```
git commit -m "Versi 0.0.1 - File Baru"
(Keterangan : Membuat commit atau label untuk unggahan yang akan dilakukan)
```

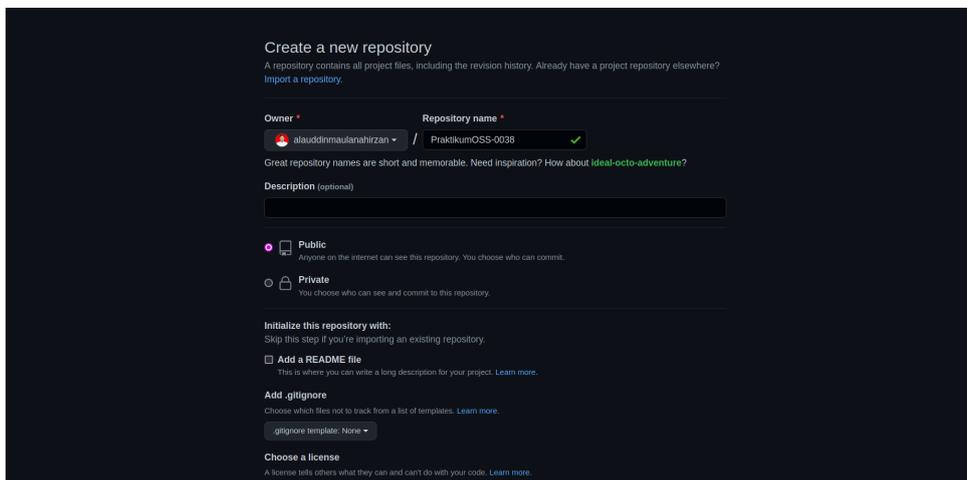
```
~/PraktikumOSS / on git master +1
> git commit -m "Versi 0.0.1 - File Baru"
[master (root-commit) a34d223] Versi 0.0.1 - File Baru
1 file changed, 1 insertion(+)
create mode 100644 Praktikum1.txt

~/PraktikumOSS / on git master
>
```

10. Namun untuk mengunggah folder ini, kita memutuskan yang namanya **Repository** (bukan akun) yang ada di dalam GitHub. Buka [github.com](https://github.com) dan buatlah Repositori baru dengan meng-klik **New**



11. Beri nama Repositori dengan **PraktikumOSS-[4 DIGIT NIM Terakhir]**. Untuk membedakan mahasiswa satu dengan yang lainnya.



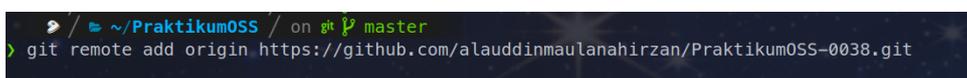
12. Kemudian klik **Create Repository** yang ada di bagian bawah.
13. Berikutnya kita akan diarahkan ke halaman repositori tersebut. Lalu **Kopi** alamat yang ada di sana.



Kembali ke **Git Bash** atau terminal dan masukkan perintah berikut:

**Perintah Terminal**

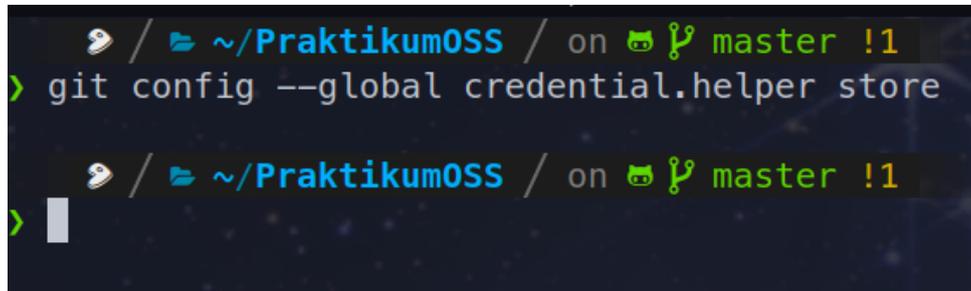
```
git remote add origin [alamat dari langkah sebelumnya]
(Keterangan : Mengeset [alamat] sebagai tujuan unggahan)
```



14. Sebelum melakukan pengunggahan, sebaiknya mengkonfigurasi GIT agar menyimpan *credentials* sehingga tidak perlu lagi memasukkan token ketika login. Masukkan perintah berikut:

Perintah Terminal

```
git config --global credential.helper store  
(Keterangan : Menyimpan username dan token ketika mengunggah)
```

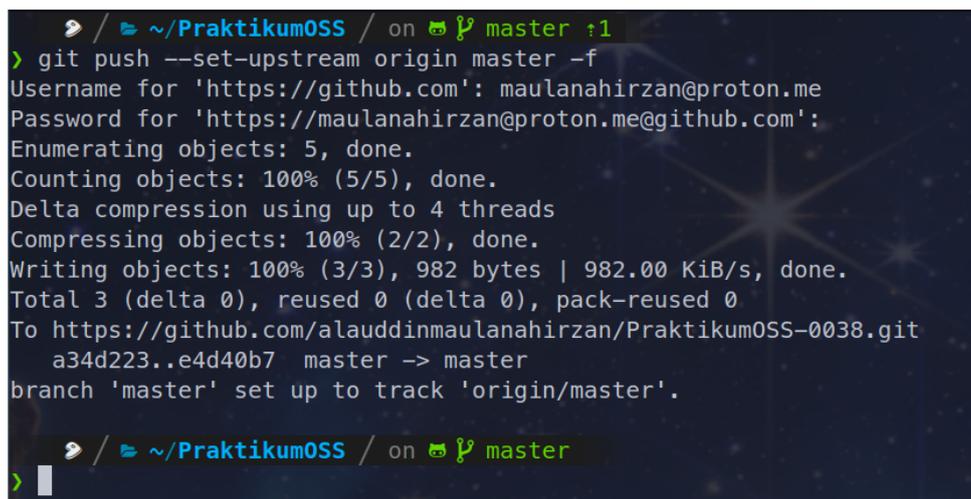


```
~/PraktikumOSS / on master !1  
> git config --global credential.helper store  
  
~/PraktikumOSS / on master !1  
>
```

15. Kini GIT siap melakukan pengunggahan. Karena masih pertama kali gunakan perintah berikut: (Catatan: **PASSWORD** diisi dengan **TOKEN**, dan **INPUT TIDAK AKAN TERLIHAT**)

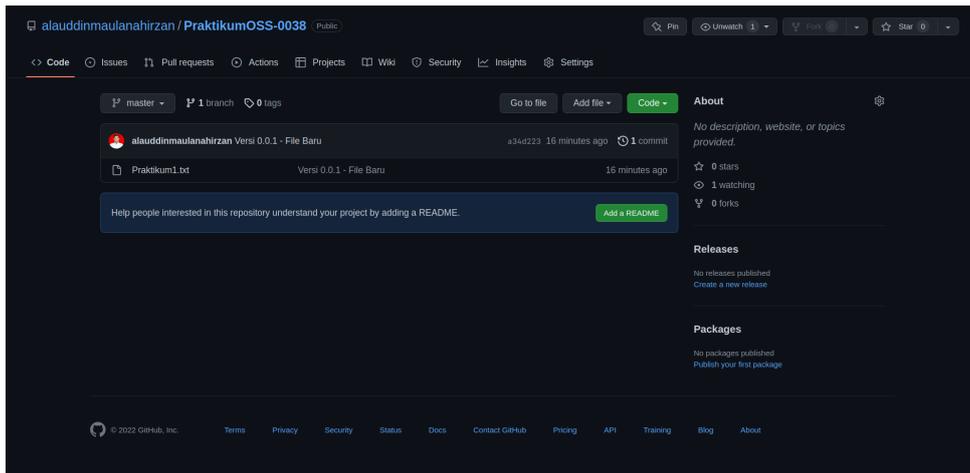
Perintah Terminal

```
git push --set-upstream origin main -f  
  
atau  
  
git push --set-upstream origin master -f  
(Keterangan : Perintah ini akan mengeset origin sebagai tujuan dan -f akan memaksa unggahan)
```

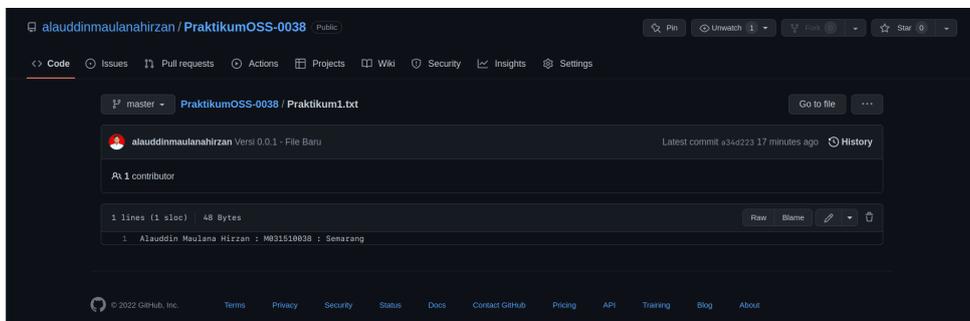


```
~/PraktikumOSS / on master :1  
> git push --set-upstream origin master -f  
Username for 'https://github.com': maulanahirzan@proton.me  
Password for 'https://maulanahirzan@proton.me@github.com':  
Enumerating objects: 5, done.  
Counting objects: 100% (5/5), done.  
Delta compression using up to 4 threads  
Compressing objects: 100% (2/2), done.  
Writing objects: 100% (3/3), 982 bytes | 982.00 KiB/s, done.  
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
To https://github.com/alauddinmaulanahirzan/PraktikumOSS-0038.git  
a34d223..e4d40b7 master -> master  
branch 'master' set up to track 'origin/master'.  
  
~/PraktikumOSS / on master  
>
```

16. Folder sudah terunggah, buka repositori tadi untuk melihat hasilnya.



17. Dan jika dibuka berisi



18. Kumpulkan hasil yang sudah dibuat ke e-Learning yang sudah disediakan

19. Praktikum 1 Selesai

# Bab 2

## Praktikum 2

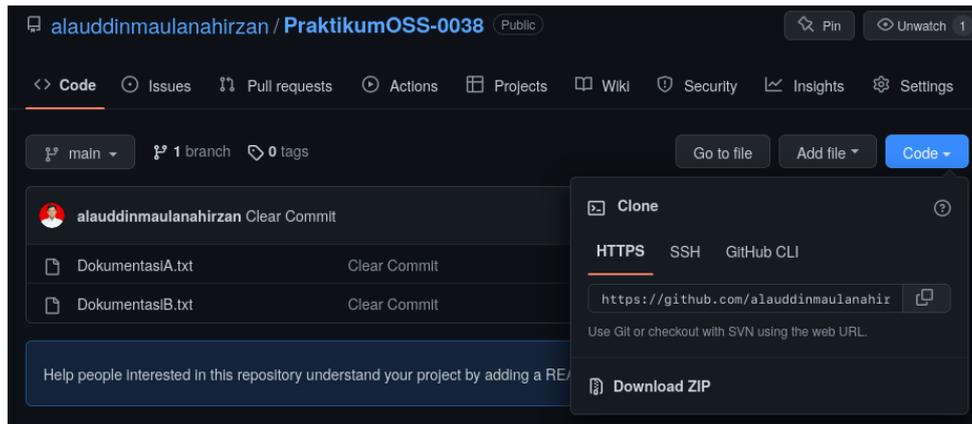
### 2.1 Kloning dan Update Repositori

Di bagian ini mahasiswa diajarkan bagaimana melakukan kloning repositori yang sudah dibuat sebelumnya, kemudian melakukan perubahan yang ada di dalam file, dan mengunggahnya kembali ke repositori. Mahasiswa diwajibkan untuk menyelesaikan **Praktikum 1** sebelum masuk ke tahapan ini.

### 2.2 Tutorial

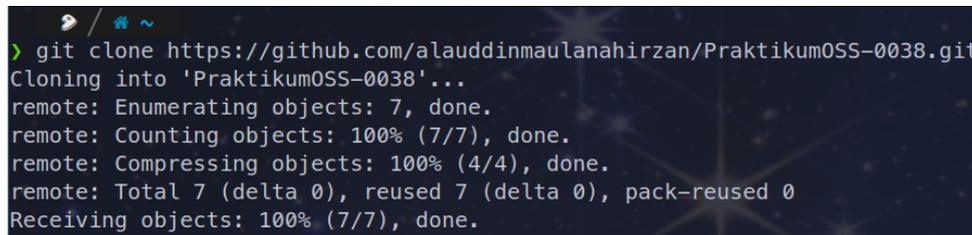
1. Buka **Git Bash** atau **Terminal**
2. Hapus folder **PraktikumOSS-[NIM]**
  - Hapus folder **PraktikumOSS-[NIM]** menggunakan **File Explorer** jika menggunakan Windows
  - Jika penghapusan dilakukan secara manual melalui perintah seperti berikut:  
**Perintah Terminal**

```
rm -rf PraktikumOSS-[NIM] (Jika Linux)
rmdir PraktikumOSS-[NIM] (Jika Windows)
```
3. Di tempat saat ini (**Bukan PraktikumOSS-[NIM]**), masukkan perintah berikut untuk melakukan kloning repositori: (url repository didapatkan melalui halaman Repository tersebut)



**Perintah Terminal**

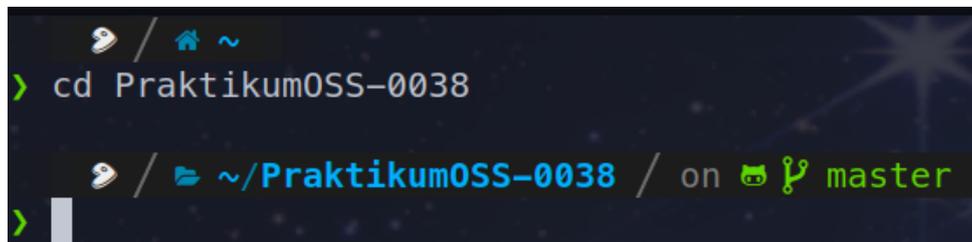
**git clone [url repository] PraktikumOSS-[4 Digit NIM]**  
**(Keterangan : Melakukan kloning dari url repository ke dalam folder)**



4. Kemudian masuk ke folder yang sudah di klon dengan perintah berikut:

**Perintah Terminal**

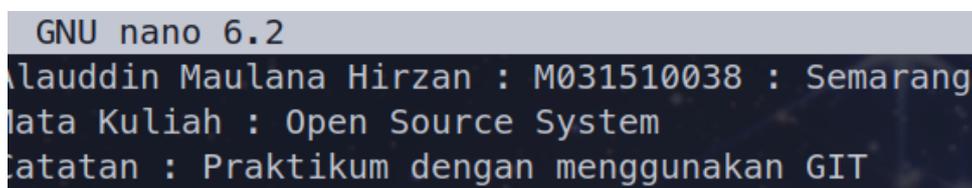
**cd PraktikumOSS-[4 Digit NIM]**



5. Buka file **Praktikum1.txt** dengan menggunakan Notepad maupun Editor Lainnya. Dan masukkan data sebagai berikut tepat di bawah data yang sudah ada

**Perintah Terminal**

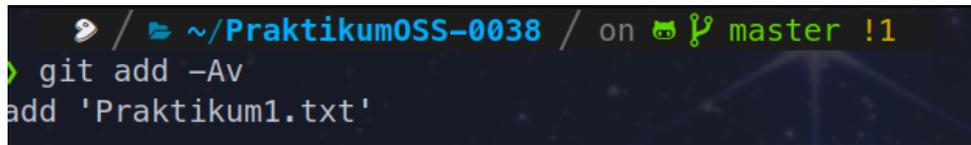
**notepad Praktikum1.txt (Hanya Windows)**  
**nano Praktikum1.txt (Hanya Linux)**  
 -----  
**Mata Kuliah : Open Source System**  
**Catatan : Praktikum dengan menggunakan GIT**



6. Data perubahan ini akan terdeteksi oleh GIT. Gunakan perintah:

Perintah Terminal

```
git add -Av
```



```
~/PraktikumOSS-0038 / on master !1  
git add -Av  
add 'Praktikum1.txt'
```

7. setelah itu lakukan *Commit* dengan perintah berikut:

Perintah Terminal

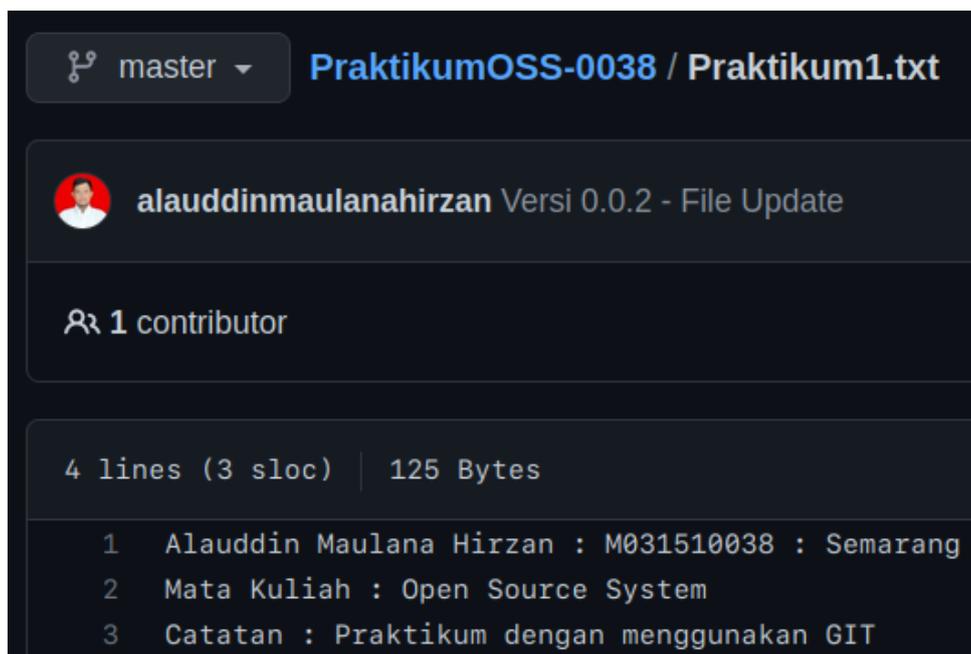
```
git commit -m "Versi 0.0.2 - File Update"
```

8. Lalu unggah dengan perintah

Perintah Terminal

```
git push
```

9. Cek file yang telah dirubah yang ada di dalam repositori



10. Praktikum 2 Selesai

## 2.3 Tugas Mandiri

Mahasiswa diwajibkan mengerjakan sesi mandiri ini setelah mengerjakan bagian **Tutorial**

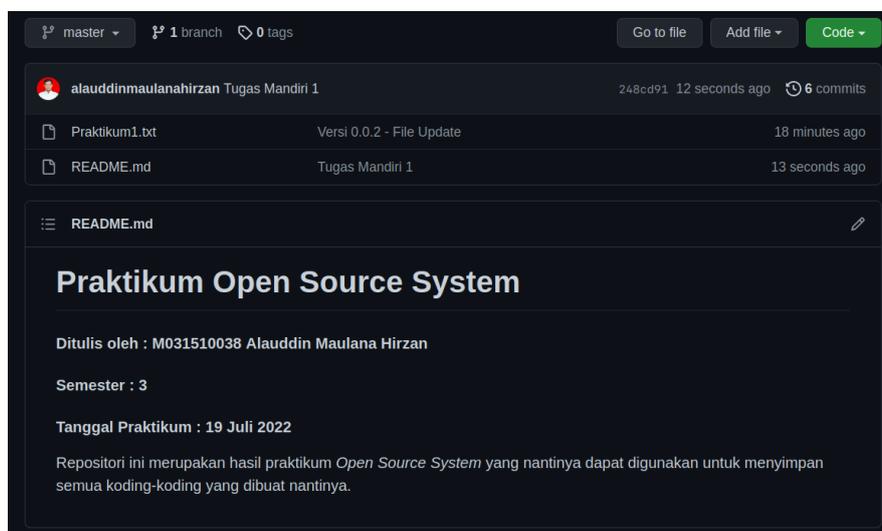
1. Buatlah satu file dengan nama **README.md** di folder **PraktikumOSS-[NIM]**
2. Dengan menggunakan **Notepad** atau **Editor Lain**, masukkan data berikut:

### Perintah Terminal

```
# Praktikum Open Source System
#### Ditulis oleh : <NIM> <Nama>
#### Semester : <Semester>
#### Tanggal Praktikum : <Tanggal>
```

Repositori ini merupakan hasil praktikum \*Open Source System\* yang nantinya dapat digunakan untuk menyimpan semua koding-koding yang dibuat.

3. Tambahkan, **Commit** dengan pesan "Tugas Mandiri 1", dan **Push** ke Repositori
4. Hasil akhir harus seperti berikut:



5. Kumpulkan hasil yang sudah dibuat ke e-Learning yang sudah disediakan

# Bab 3

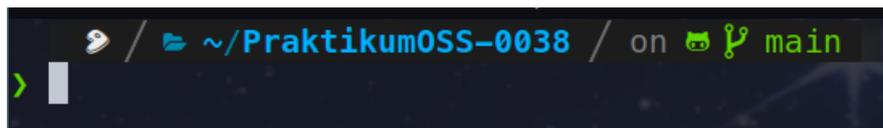
## Praktikum 3

### 3.1 Percabangan Repositori

Di bagian ini mahasiswa diajarkan bagaimana membuat percabangan (branch) dari repositori utama yang sudah dibuat sebelumnya, kemudian melakukan perubahan yang ada di dalam file, dan mengunggahnya kembali ke repositori sebagai cabang yang berbeda. Mahasiswa diwajibkan untuk menyelesaikan **Praktikum 2** sebelum masuk ke tahapan ini.

### 3.2 Tutorial

1. Buka folder sebelumnya (**PraktikumOSS-[NIM]**) dengan menggunakan **Git Bash** atau **Terminal**
2. Pastikan posisi **Git Bash** atau **Terminal** sudah berada di folder tersebut



3. Untuk memulai cabang baru dari repositori lokal saat ini, gunakan perintah berikut:
  - Perintah Alternatif 1

```
Perintah Terminal  
git branch Cabang1  
(Keterangan : Buat cabang baru dengan nama Cabang1)  
git checkout Cabang1  
(Keterangan : Pindah cabang aktif ke Cabang1)
```

```
~/PraktikumOSS-0038 / on main
> git branch Cabang1

~/PraktikumOSS-0038 / on main
> git checkout Cabang1
Switched to branch 'Cabang1'
```

- Perintah Alternatif 2

```
Perintah Terminal
git checkout -b Cabang1
(Keterangan : Buat dan pindah cabang aktif ke Cabang1)
```

```
~/PraktikumOSS-0038 / on main
> git checkout -b Cabang1
Switched to a new branch 'Cabang1'
```

4. Sesudah itu, cabang akan dialihkan ke **Cabang1**, bukan **main** lagi. Kita dapat memodifikasi sesuka kita di cabang ini tanpa harus mengganggu cabang utama.
5. Dengan menggunakan **Notepad** atau **Editor Lainnya**, buatlah sebuah file dengan nama **File Cabang Baru.txt**.

```
Perintah Terminal
notepad "File Cabang Baru.txt" (Hanya Windows)
nano "File Cabang Baru.txt" (Hanya Linux)
```

6. Kemudian isi file tersebut dengan isi sebagai berikut:

```
Perintah Terminal
File ini merupakan file yang hanya bisa dilihat di Cabang1. Sehingga tidak akan muncul di Cabang Utama (Main)
```

```
GNU nano 6.2 File Cabang Baru.txt
File ini merupakan file yang hanya bisa dilihat di Cabang1. Sehingga tidak akan muncul di Cabang Utama (Main)
```

7. Masukkan file ke dalam **stage** dengan perintah

```
Perintah Terminal
git add -Av
```

8. Beri **Commit** baru ke cabang yang akan diunggah

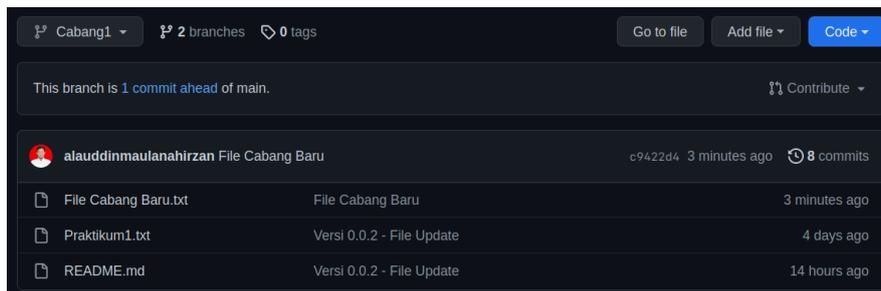
```
Perintah Terminal
git commit -m "File Cabang Baru"
```

9. Lalu unggah dengan perintah berikut

```
Perintah Terminal
git push --set-upstream origin Cabang1 -f
(Keterangan : Unggah Repositori ke Cabang1)
```

```
~/PraktikumOSS-0038 / on git Cabang1
> git push --set-upstream origin Cabang1 -f
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.06 KiB | 1.06 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'Cabang1' on GitHub by visiting:
remote:   https://github.com/alauddinmaulanahirzan/PraktikumOSS-0038/pull/new/Cabang1
remote:
To https://github.com/alauddinmaulanahirzan/PraktikumOSS-0038.git
 * [new branch]      Cabang1 -> Cabang1
branch 'Cabang1' set up to track 'origin/Cabang1'.
```

10. Cek apakah sudah terunggah sempurna di Repositori Remote



11. Jika berhasil maka **Cabang1** akan muncul di daftar Branch termasuk **File Baru** yang kita buat.

12. Praktikum 3 Selesai

### 3.3 Tugas Mandiri

1. Buat cabang baru dengan nama **Cabang2**
2. Buatlah file baru dengan nama **FileCabang2.txt**
3. Isi dengan teks berikut:

**Perintah Terminal**

**File Tugas Mandiri Cabang 2**

4. Unggah ke Repositori Remote
5. Hasil screenshot mirip **Langkah 10**, kirimkan ke e-Learning

# Bab 4

## Praktikum 4

### 4.1 Penggabungan dan Penghapusan Cabang Repositori

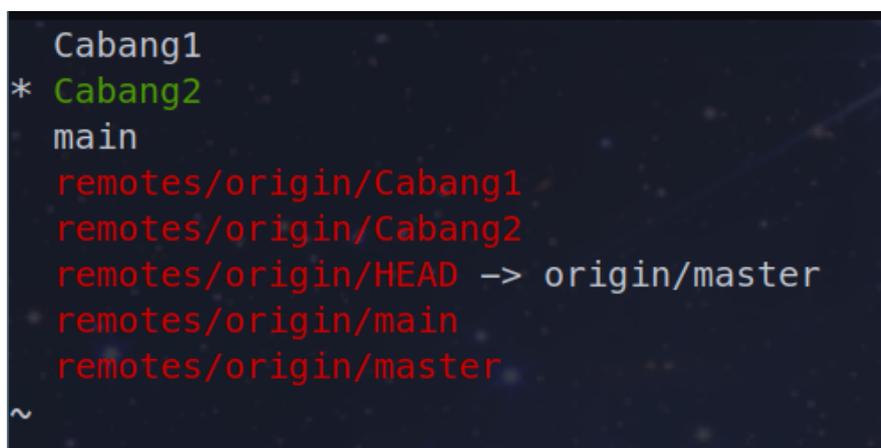
Di bagian ini mahasiswa diajarkan bagaimana menggabungkan percabangan (branch) sudah dibuat sebelumnya. Selain itu juga mengajarkan bagaimana melakukan penghapusan cabang yang ada di luring maupun daring. Mahasiswa diwajibkan untuk menyelesaikan **Praktikum 3** sebelum masuk ke tahapan ini.

### 4.2 Tutorial

1. Buka **Git Bash** atau **Terminal** dan buka folder Praktikumnya
2. Sebelum melakukan penggabungan, cek cabang yang sudah dibuat dari **Praktikum 3** dengan menggunakan perintah sebagai berikut:

**Perintah Terminal**

```
git branch -a  
(Keterangan : Tampilkan semua cabang baik luring maupun daring)
```



```
Cabang1  
* Cabang2  
main  
remotes/origin/Cabang1  
remotes/origin/Cabang2  
remotes/origin/HEAD -> origin/master  
remotes/origin/main  
remotes/origin/master  
~
```

3. Jika tulisan berwarna **Putih** maka cabang itu bersifat **LOKAL**. Jika tulisan berwarna **Hijau** dan memiliki tanda **ASTERISK \***, maka cabang itu merupakan cabang **Ak-**

tif. Jika tulisan berwarna **Merah** dengan awalan **Remote**, maka cabang tersebut adalah **ONLINE / REMOTE**

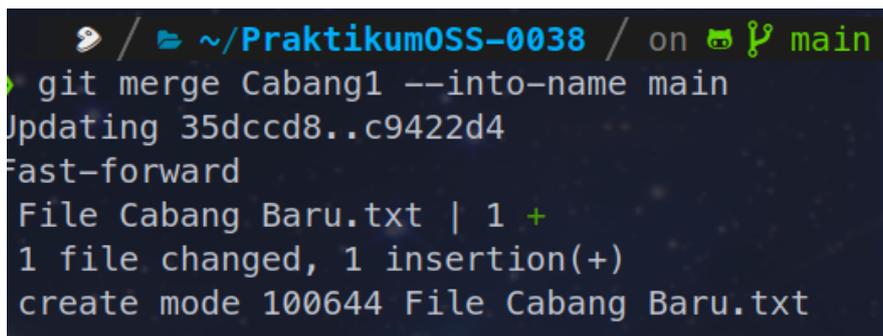
4. Tutup aplikasi dengan menggunakan kombinasi:

```
Perintah Terminal  
:q!  
(Jika menggunakan Editor vim)  
CTRL+X  
(Jika menggunakan Editor nano)  
ALT+F4  
(Jika menggunakan Notepad)
```

5. Di praktikum ini kita akan menggabungkan cabang **Cabang1** ke **Main**. Namun pastikan bahwa kita dalam posisi bukan di **Cabang1** maupun **Cabang2**. Pindah ke cabang **main** dengan perintah **checkout**

6. Untuk melakukan **merge**, gunakan perintah berikut:

```
Perintah Terminal  
git merge Cabang1 --into-name main  
(Keterangan : Gabungkan Cabang1 ke main)
```

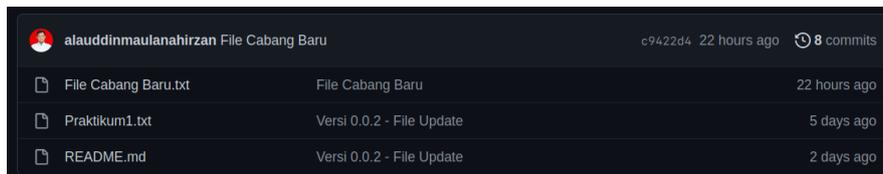


7. Cabang **Cabang1** tidak akan terhapus, tetapi semua file nya akan menggabung ke **main**.

8. Lalu lakukan **PUSH** dengan perintah berikut:

```
Perintah Terminal  
git push
```

9. File dari **Cabang1** akan menggabung ke **main**



10. Hapus sisa-sisa cabang yang tidak dipakai dengan menggunakan perintah

```
Perintah Terminal  
git branch -D Cabang1  
git branch -D Cabang2  
(Keterangan : Hapus repositori lokal Cabang1 dan Cabang2)
```

```
~/PraktikumOSS-0038 / on main
> git branch -D Cabang1
Deleted branch Cabang1 (was c9422d4).

~/PraktikumOSS-0038 / on main
> git branch -D Cabang2
Deleted branch Cabang2 (was 8c53f40).

~/PraktikumOSS-0038 / on main
>
```

11. Lalu hapus juga repositori remote nya dengan menggunakan perintah berikut

**Perintah Terminal**

```
git push -d origin Cabang1
git push -d origin Cabang2
(Keterangan : Hapus repositori remote Cabang1 dan Cabang2)
```

```
~/PraktikumOSS-0038 / on main
> git push -d origin Cabang1
To https://github.com/alauddinmaulanahirzan/PraktikumOSS-0038.git
- [deleted]          Cabang1

~/PraktikumOSS-0038 / on main
> git push -d origin Cabang2
To https://github.com/alauddinmaulanahirzan/PraktikumOSS-0038.git
- [deleted]          Cabang2
```

12. Praktikum 4 Selesai

# Bab 5

## Praktikum 5

### 5.1 *Undo* dan *Stash*

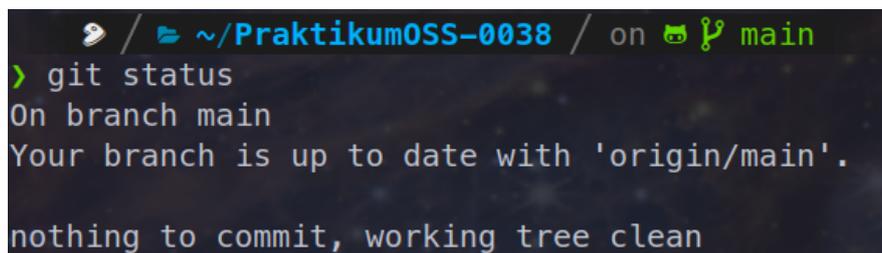
Di bagian ini mahasiswa diajarkan bagaimana membatalkan aksi yang telah dilakukan, beserta menyimpan aksi yang telah dilakukan sebagai *stash* atau *draft* sehingga perubahan yang dilakukan tidak mengubah cabang yang digunakan. Mahasiswa diwajibkan untuk menyelesaikan **Praktikum 4** sebelum masuk ke tahapan ini.

### 5.2 Tutorial

1. Buka **Git Bash** atau **Terminal**. Lalu arahkan ke folder **PraktikumOSS** yang telah dibuat.
2. Pastikan folder tersebut tidak dalam proses perubahan. Untuk mengecek status folder tersebut gunakan perintah:

\_\_\_\_\_ Perintah Terminal \_\_\_\_\_

```
git status  
(Keterangan : Mengecek status folder saat ini)
```



```
~/PraktikumOSS-0038 / on main  
> git status  
On branch main  
Your branch is up to date with 'origin/main'.  
  
nothing to commit, working tree clean
```

3. Jika GIT melaporkan *clean*, maka lanjutkan ke langkah berikutnya. Jika tidak, ulangi langkah sebelumnya di praktikum sebelumnya.
4. Perintah **Reset** hanya berlaku jika pengguna telah melakukan perubahan. Oleh karena itu buatlah satu file text dengan nama "**File.txt**" menggunakan **Notepad** atau perintah **touch**.

## Perintah Terminal

notepad File.txt (Hanya Windows)

touch File.txt (Hanya Linux)

5. Cek dengan menggunakan perintah **dir** (jika Windows) atau **ls** (jika Linux) untuk mengecek keberadaan file.

```
~/PraktikumOSS-0038 / on main ?1
> ls
'File Cabang Baru.txt'  File.txt  Praktikum1.txt  README.md
```

6. Cek juga menggunakan **git status** untuk melihat perubahan di repo lokal. Jika muncul tulisan **untracked file**, maka sudah terjadi perubahan di repositori namun tidak ditambahkan ke **stage**

```
~/PraktikumOSS-0038 / on main ?1
> git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>.." to include in what will be committed)
  File.txt

nothing added to commit but untracked files present (use "git add" to track)
```

7. Tambahkan file baru tersebut ke stage dengan perintah **Add**. Lalu hapus file tersebut (ceritanya tidak disengaja)
8. Kemudian cek kembali status dari repositori tersebut. Jika dilakukan dengan benar, maka git status akan melaporkan dua status file tersebut.

```
~/PraktikumOSS-0038 / on main +1
> rm File.txt

~/PraktikumOSS-0038 / on main +1 !1
> git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>.." to unstage)
  new file:   File.txt

Changes not staged for commit:
  (use "git add/rm <file>.." to update what will be committed)
  (use "git restore <file>.." to discard changes in working directory)
  deleted:   File.txt
```

9. Untuk mengembalikan file yang dihapus tadi, kita bisa menggunakan perintah:

## Perintah Terminal

**git checkout <Nama File> / git restore <Nama File>**

(Keterangan : Mengembalikan file yang terhapus)

```
~/PraktikumOSS-0038 / on main +1 !1
> git checkout File.txt
Updated 1 path from the index

~/PraktikumOSS-0038 / on main +1
> git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   File.txt
```

- Status GIT kembali seperti semua, dan siap ke langkah selanjutnya yaitu **STASH**.
- Sama halnya seperti **UNDO**, **STASH** hanya bisa dilakukan jika perubahan sudah terjadi. Untuk menyimpan **Kondisi Saat Ini** repositori lokal, gunakan perintah berikut:

**Perintah Terminal**

```
git stash
(Keterangan : Simpan kondisi saat ini sebagai draft)
```

```
~/PraktikumOSS-0038 / on main +1
> git stash
Saved working directory and index state WIP on main: c9422d4 File Cabang Baru
```

- Semua perubahan yang dilakukan akan tersimpan dan repositori kembali bersih seperti sedia kala

```
~/PraktikumOSS-0038 / on main *1
> git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
```

- Kita bisa mengatur **stash** yang bisa dibuat dengan menggunakan perintah:

**Perintah Terminal**

```
git stash list (Keterangan : Melihat daftar stash)
git stash show (Keterangan : Melihat perubahan dalam stash)
git stash apply (Keterangan : Menerapkan perubahan dari stash)
git stash drop (Keterangan : Membatalkan stash)
git stash clear (Keterangan : Menghapus semua stash yang ada)
```

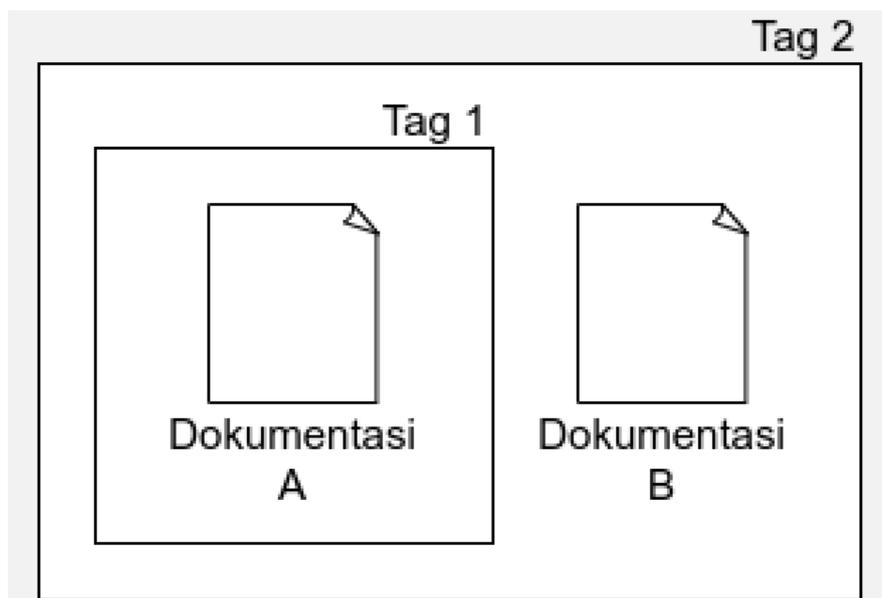
- Untuk mengakhiri praktikum ini hapus semua stash yang ada dengan salah satu perintah di atas
- Praktikum 5 Selesai

# Bab 6

## Praktikum 6

### 6.1 Tagging

Di bagian ini mahasiswa diajarkan bagaimana menambahkan atau menghapus *tanda* atau disebut dengan *tag* dari suatu titik repositori. Teknik ini mirip dengan branch, namun biasanya digunakan menandai versi file yang ada di dalam repositori. Lihat Gambar di bawah.



File **DokumentasiA** yang diberi **Tag 1** tidak akan menyertakan file **DokumentasiB**. Karena file kedua tidak disertakan saat pemberian **tag**. Sedangkan **Tag 2** menyertakan file **DokumentasiA** dan **DokumentasiB** karena ketika diberikan **tag**, kedua file ada. Mahasiswa diwajibkan untuk menyelesaikan **Praktikum 5** sebelum masuk ke tahapan ini.

### 6.2 Tutorial

1. Buka **Terminal** atau **Git Bash**, dan masuk ke folder praktikum lalu.

2. Hapus semua file yang ada di folder tersebut (boleh menggunakan file explorer) atau menggunakan perintah berikut jika menggunakan Linux:

Perintah Terminal

```
rm *  
(Keterangan : Hapus semua file)
```

3. Perbarui repositori remote dengan perintah **Add**, **Commit**, dan **Push**. Praktikum ini dimulai dengan repositori kosong baik di lokal maupun remote. (Pesan Commit bebas)
4. Buatlah file baru dengan menggunakan Editor masing-masing. Beri nama **DokumentasiA.txt**. Unggah juga file tersebut dengan perintah **Add**, **Commit**, dan **Push**. (Pesan Commit bebas)

Perintah Terminal

```
notepad DokumentasiA.txt (Hanya Windows)  
touch DokumentasiA.txt (Hanya Linux)
```



5. Berikan Tag 1 ke file tersebut dengan perintah berikut (Cara 2 direkomendasikan):

- (a) Cara 1:

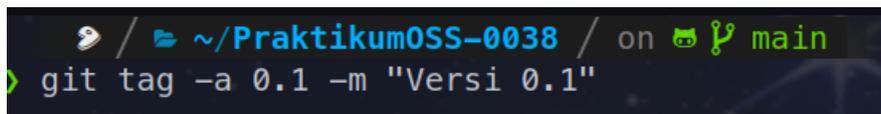
Perintah Terminal

```
git tag -a <Versi>  
(Keterangan : Memberi Tag Sambil Membuka Editor)
```

- (b) Cara 2:

Perintah Terminal

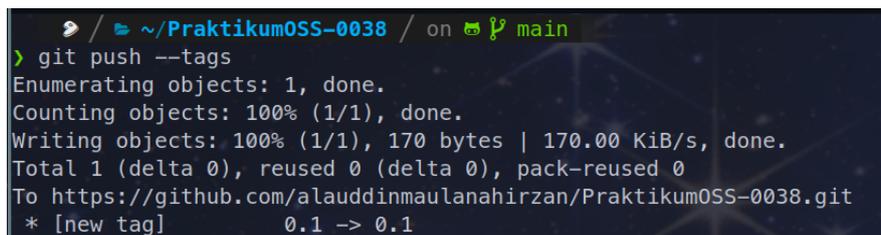
```
git tag -a <Versi> -m "<Pesan>"  
(Keterangan : Memberi Tag Beserta Pesan)
```



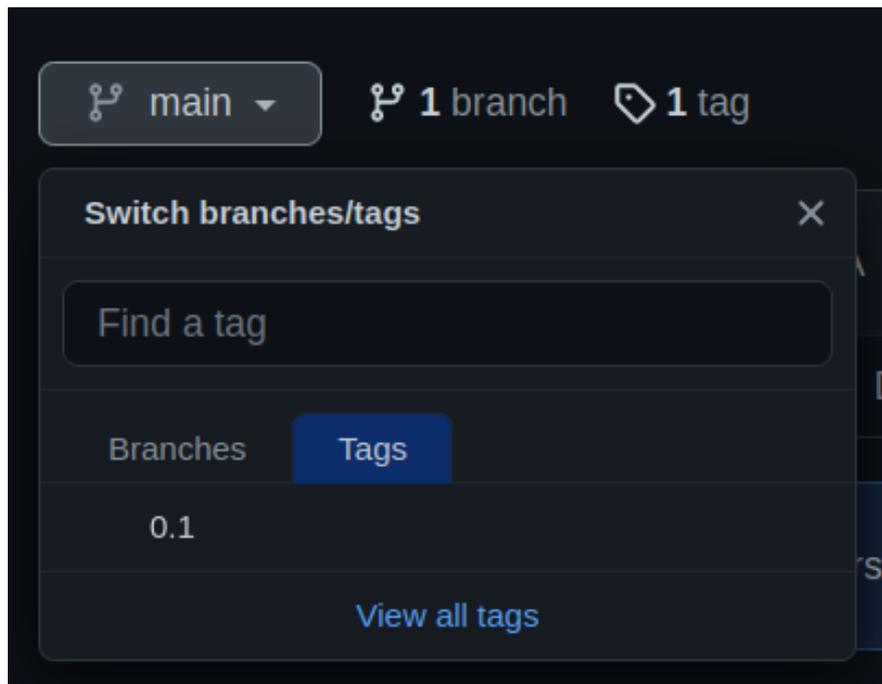
6. **Tag** tidak akan terunggah otomatis, untuk mengunggah gunakan perintah:

Perintah Terminal

```
git push --tags  
(Keterangan : Unggah Tags)
```



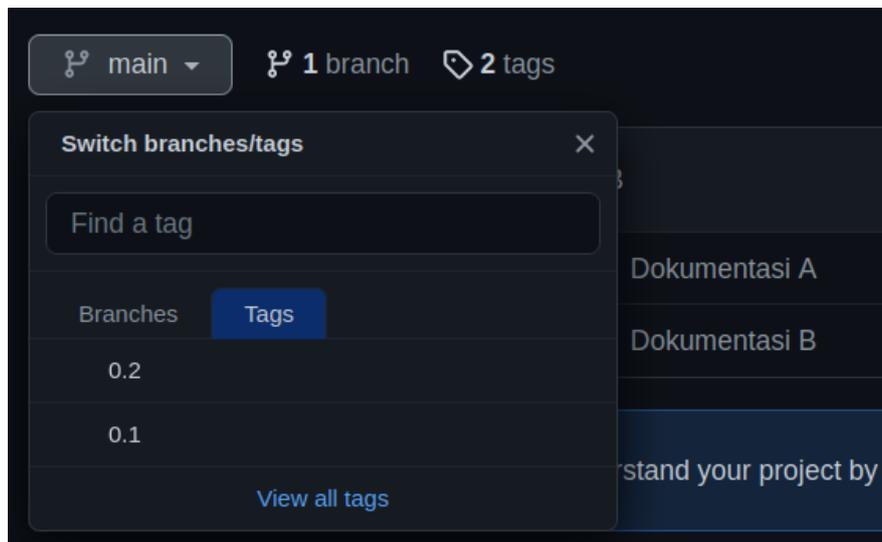
7. Buka repositori remote untuk mengecek **tags** secara daring



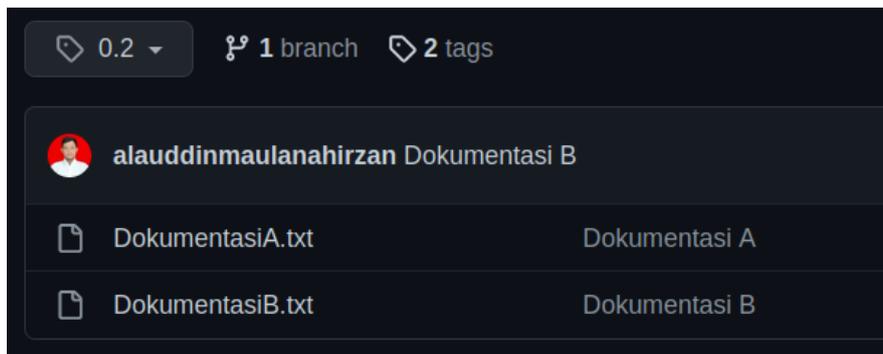
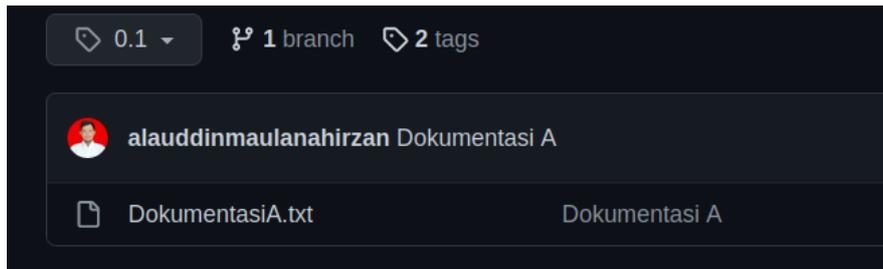
8. Buat file baru dengan nama **DokumentasiB**, lalu unggah dengan **Add**, **Commit**, dan **Push**. (Pesan Commit bebas)

9. beri tag lagi namun dengan versi yang berbeda. (Contoh 0.2)

10. Unggah **Tags**. dan cek Repositori Hasil akhir seperti berikut:



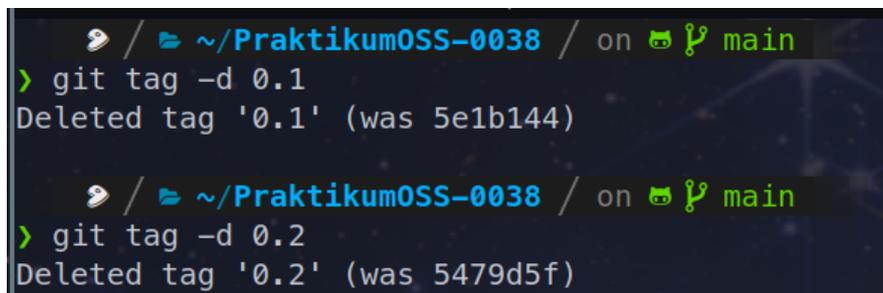
11. Bandingkan **Tag 0.1** dengan **Tag 0.2**



12. Untuk menghapus **tags** yang sudah dibuat, dapat dilakukan dengan perintah berikut:

**Perintah Terminal**

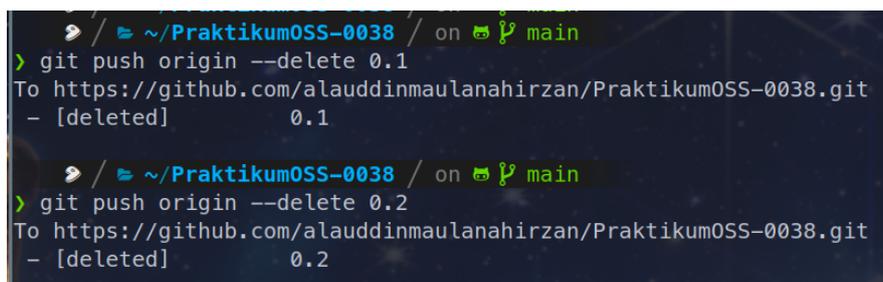
```
git tag -d <Versi>  
(Keterangan : Hapus <Versi> dari lokal)
```



13. Lalu hapus **tags** yang berada di remote dengan perintah berikut:

**Perintah Terminal**

```
git push origin --delete <Versi>  
(Keterangan : Hapus Tags dari Origin)
```



14. Praktikum 6 Selesai

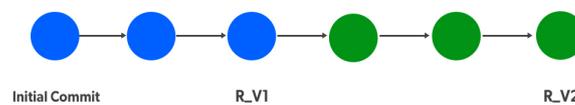
# Bab 7

## Praktikum 7

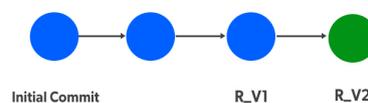
### 7.1 Commit Squashing

Di bagian ini mahasiswa diajarkan bagaimana melakukan menggabungkan dua commit yang sudah dilakukan menjadi satu commit utuh. Berikut ini adalah ilustrasi Sebelum dan Sesudah *Squash*.

**Sebelum**



**Sesudah**

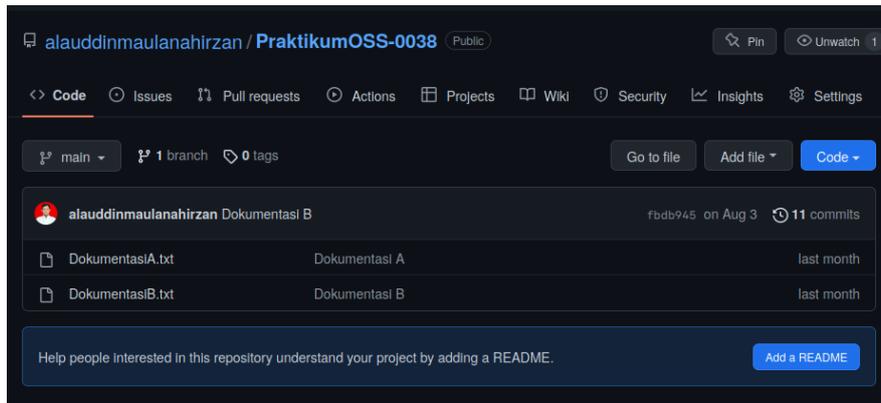


Mahasiswa diwajibkan untuk menyelesaikan **Praktikum 6** sebelum masuk ke tahapan ini.

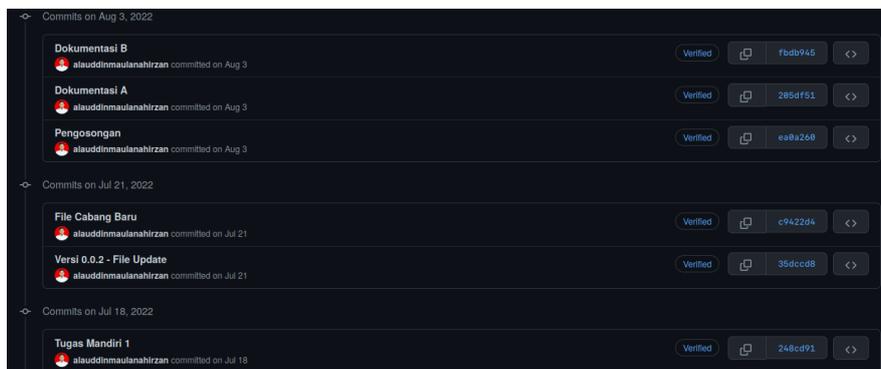
### 7.2 Tutorial

1. Buka **Terminal** atau **Git Bash**, dan masuk ke folder praktikum lalu.

2. Cek **Commit** yang sudah terjadi di repositori remote dengan membuka **github.com** dan repositori target.

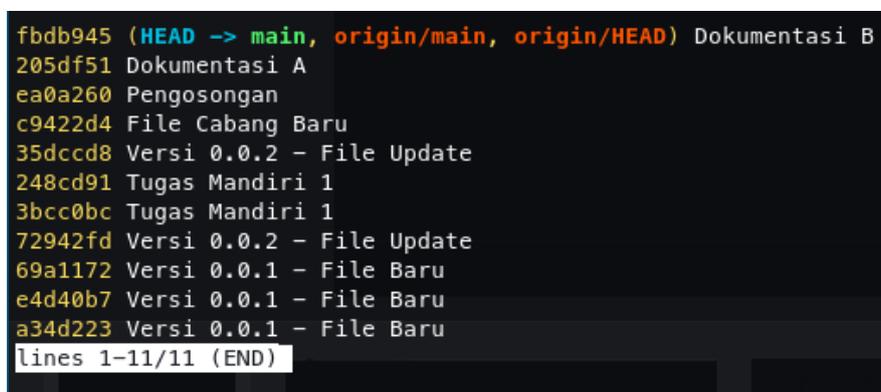


3. Jika dilihat, repositori ini memiliki **11 Commit**. Jumlah **commit** bisa berbeda dengan mahasiswa. Klik angka tersebut untuk menampilkan **history commit**



4. Di sini kita bisa melihat sejarah daripada **commit** yang sudah dibuat.
5. Atau dapat menggunakan perintah berikut untuk melihat melalui **GitBash**

```
Perintah Terminal  
git log --oneline  
(Keterangan : Tampilkan Log Commit)
```



6. Dengan menggunakan **squash**, git dapat menyatukan beberapa **commit** ke **commit** titik tertentu.

7. Untuk melakukan **squashing** 2 commit ke belakang, gunakan perintah

**Perintah Terminal**

```
git rebase -i HEAD~5  
(Keterangan : Plih 5 Commit Target Squash)
```

8. **GIT** akan menampilkan Teks Editor untuk me-review perubahan

```
GNU nano 6.3 /home/maulana/Downloads/PraktikumOSS
pick 35dccd8 Versi 0.0.2 - File Update
pick c9422d4 File Cabang Baru
pick ea0a260 Pengosongan
pick 205df51 Dokumentasi A
pick fbdb945 Dokumentasi B

# Rebase 248cd91..fbdb945 onto 248cd91 (5 commands)
#
# Commands:
# p, pick <commit> = use commit
# r, reword <commit> = use commit, but edit the commit message
# e, edit <commit> = use commit, but stop for amending
# s, squash <commit> = use commit, but meld into previous commit
# f, fixup [-C | -c] <commit> = like "squash" but keep only the previous
#   commit's log message, unless -C is used, in which case
#   keep only this commit's message; -c is same as -C but
#   opens the editor
# x, exec <command> = run command (the rest of the line) using shell
# b, break = stop here (continue rebase later with 'git rebase --continue')
# d, drop <commit> = remove commit
# l, label <label> = label current HEAD with a name
# t, reset <label> = reset HEAD to a label
# m, merge [-C <commit> | -c <commit>] <label> [# <oneline>]
# .   create a merge commit using the original merge commit's
# .   message (or the oneline, if no original merge commit was
# .   specified); use -c <commit> to reword the commit message
#
# These lines can be re-ordered; they are executed from top to bottom.
```

9. Untuk menggabungkan **commit**, ubah **pick** menjadi **squash**. Contoh gambar ini menggabungkan **3 Commit**:

```
GNU nano 6.3 /home/maulana/Downloads/PraktikumOSS
pick 35dccd8 Versi 0.0.2 - File Update
pick c9422d4 File Cabang Baru
squash ea0a260 Pengosongan
squash 205df51 Dokumentasi A
squash fbdb945 Dokumentasi B
```

10. Lalu exit, maka **squashing** sudah dilakukan secara otomatis. Dan **GIT** akan menampilkan hasil **squash**

```

GNU nano 6.3 /home/maulana/Downloads/PraktikumOSS-
# This is a combination of 4 commits.
# This is the 1st commit message:

File Cabang Baru

# This is the commit message #2:

Pengosongan

# This is the commit message #3:

Dokumentasi A

# This is the commit message #4:

Dokumentasi B

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# Date:      Thu Jul 21 19:47:09 2022 +0700
#
# interactive rebase in progress; onto 248cd91
# Last commands done (5 commands done):
#   squash 205df51 Dokumentasi A
#   squash fdb945 Dokumentasi B
# No commands remaining.
# You are currently rebasing branch 'main' on '248cd91'.
#

```

11. Cukup exit dengan menekan tombol **CTRL+X** jika menggunakan editor sama persis dengan gambar. Atau **Close** jika menggunakan **Notepad**
12. Hasil **Squash** Sukses

```

(base)
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:main o [16:50:55]
$ git rebase -i HEAD~5
[detached HEAD af0c4c3] File Cabang Baru
Date: Thu Jul 21 19:47:09 2022 +0700
4 files changed, 10 deletions(-)
create mode 100644 DokumentasiA.txt
create mode 100644 DokumentasiB.txt
delete mode 100644 Praktikum1.txt
delete mode 100644 README.md
Successfully rebased and updated refs/heads/main.
(base)
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:main o [16:56:03]
$

```

13. Bandingkan **history commit** dengan perintah sebelumnya. Hasilnya akan lebih sedikit karena proses **squash**

```

af0c4c3 (HEAD -> main) File Cabang Baru
35dcd8 Versi 0.0.2 - File Update
248cd91 Tugas Mandiri 1
3bcc0bc Tugas Mandiri 1
72942fd Versi 0.0.2 - File Update
69a1172 Versi 0.0.1 - File Baru
e4d40b7 Versi 0.0.1 - File Baru
a34d223 Versi 0.0.1 - File Baru
lines 1-8/8 (END)

```

14. Praktikum 7 Selesai

# Bab 8

## Praktikum 8

### 8.1 Reset Commit

Di bagian ini mahasiswa diajarkan bagaimana melakukan menghilangkan semua *history commit* yang sudah pernah dilakukan oleh mahasiswa. Sehingga mahasiswa dapat menghapus jejak-jejak file yang sekiranya berbahaya untuk dipublikasikan ke publik. Mahasiswa diwajibkan untuk menyelesaikan **Praktikum 7** sebelum masuk ke tahapan ini.

### 8.2 Tutorial

1. Buka **Terminal** atau **Git Bash**, dan masuk ke folder praktikum lalu.
2. Untuk melakukan **refresh** atau **reset commit** yang sudah pernah dilakukan, cukup dilakukan dengan memindahkan **branch** saat ini ke **branch orphan**
3. Masukkan perintah berikut untuk memulai **branch orphan** baru

#### Perintah Terminal

```
git checkout --orphan temp main  
(Keterangan : Buat Branch Orphan Dengan Nama temp)
```

```
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:main o [17:02:49]  
$ git checkout --orphan temp main  
Switched to a new branch 'temp'  
(base)  
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:temp x [17:03:00]  
$
```

4. Hapus **branch** saat ini dengan salah satu perintah berikut

#### Perintah Terminal

```
git branch -D main  
git branch -D master  
(Keterangan : Hapus Branch main atau Branch master)
```

```
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:temp x [17:03:00]  
$ git branch -D main  
Deleted branch main (was af0c4c3).
```

5. Ubah nama **Branch** saat ini (temp) menjadi **Branch** utama (main) dengan perintah berikut

**Perintah Terminal**

```
git branch --move temp main  
(Keterangan : Ubah Nama Branch ke main)
```

```
(base)  
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:temp x [17:10:42]  
$ git branch --move temp main  
(base)  
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:main x [17:10:46]  
$
```

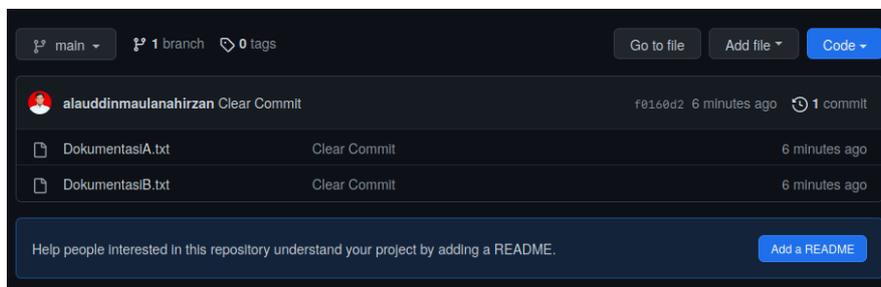
6. Lalu lanjutkan proses seperti biasa yaitu:

- git add -A
- git commit -m "Clear Commit"
- git push --set-upstream origin main --force

```
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:main x [17:13:15]  
$ git add -A  
(base)  
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:main x [17:13:17]  
$ git commit -m "Clear Commit"  
[main (root-commit) f0160d2] Clear Commit  
2 files changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 DokumentasiA.txt  
create mode 100644 DokumentasiB.txt
```

```
# maulana @ GentooHP in ~/Downloads/PraktikumOSS-0038 on git:main o [17:13:49] C:1  
$ git push --set-upstream origin main --force  
Enumerating objects: 3, done.  
Counting objects: 100% (3/3), done.  
Delta compression using up to 8 threads  
Compressing objects: 100% (2/2), done.  
Writing objects: 100% (3/3), 891 bytes | 891.00 KiB/s, done.  
Total 3 (delta 0), reused 2 (delta 0), pack-reused 0  
To https://github.com/alauddinmaulanahirzan/PraktikumOSS-0038.git  
+ fbdb945...f0160d2 main -> main (forced update)  
branch 'main' set up to track 'origin/main'.
```

7. Cek repositori yang ada di github.com, dan pastikan hanya **1 Commit** yang tersisa



8. Praktikum 8 Selesai

# Bab 9

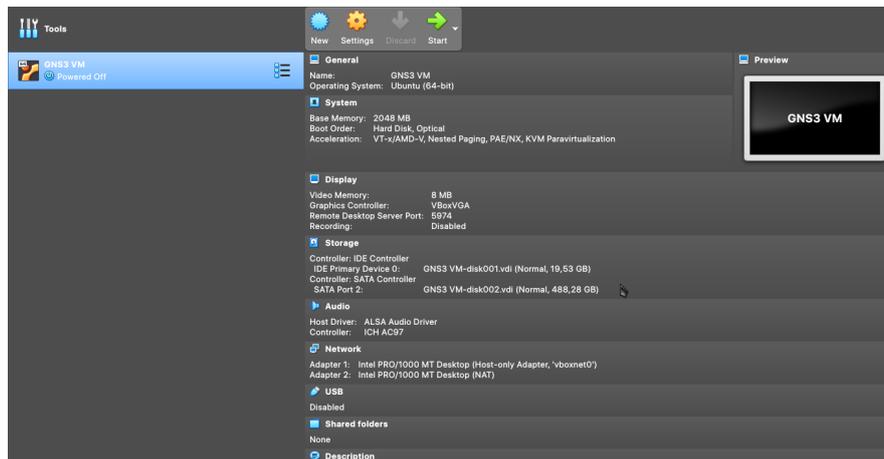
## Praktikum 9

### 9.1 Instalasi Linux Ubuntu

Di bagian ini mahasiswa diajarkan bagaimana melakukan instalasi sistem operasi sumber bebas menggunakan Ubuntu Linux di VirtualBox. Mahasiswa diperbolehkan menggunakan sistem operasi lain namun materi ini tidak akan membahas sistem operasi di luar Ubuntu. Mahasiswa direkomendasikan untuk mengunduh *image* instalasi Ubuntu dengan versi terbaru (**Ubuntu 22.04 LTS Jammy** disaat materi ini dibuat) yang dapat didapatkan melalui website resmi Ubuntu.

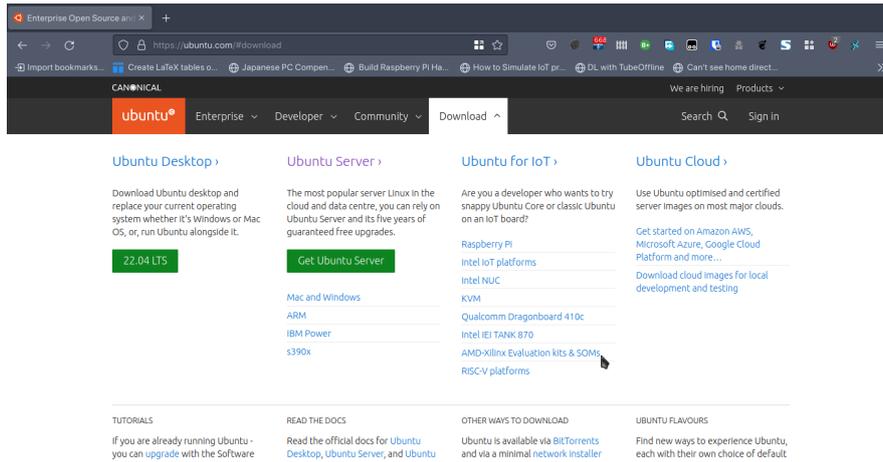
### 9.2 Tutorial

1. Buka **VirtualBox** yang sudah terpasang di komputer masing-masing

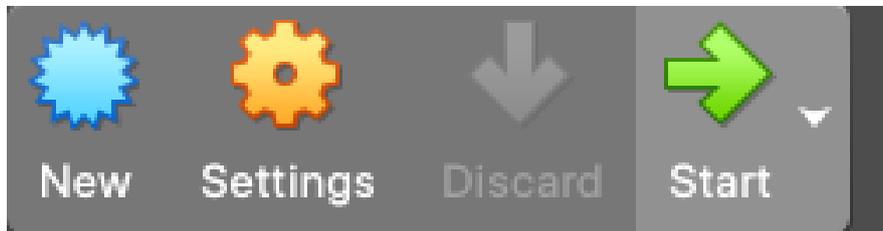


2. Unduh **ISO** Ubuntu melalui link berikut ini :

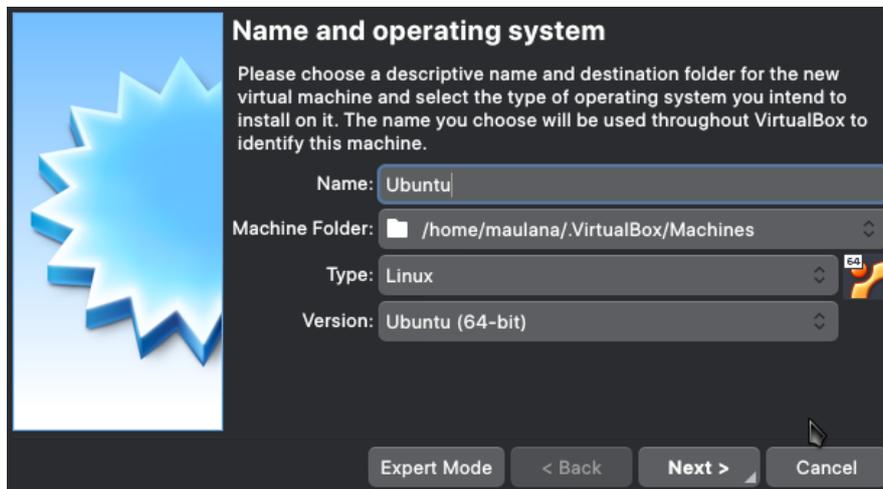
- <https://ubuntu.com/#download>
- <http://repo.ugm.ac.id/iso/ubuntu/releases/22.04.1/ubuntu-22.04.1-desktop-amd64.iso>
- <https://mirror.webmaster.my.id/ubuntu-cd/jammy/ubuntu-22.04.1-desktop-amd64.iso>



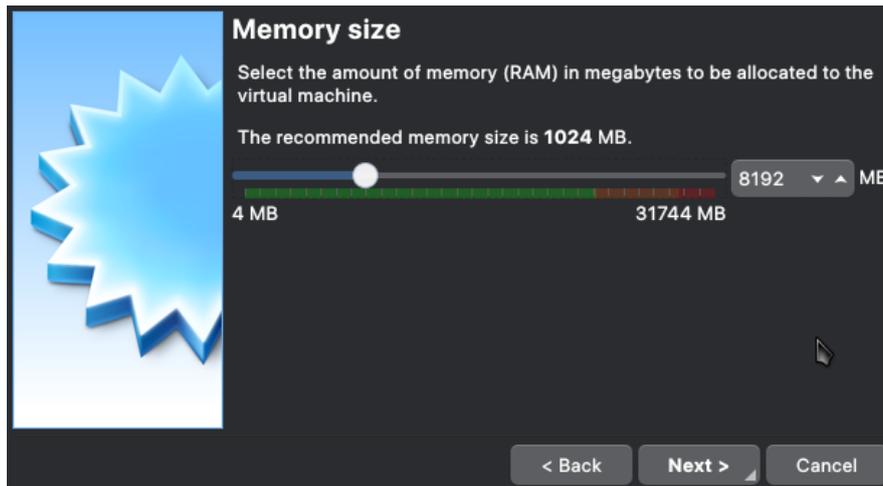
3. Setelah selesai mengunduh ISO, kembali ke VirtualBox dan klik **New**



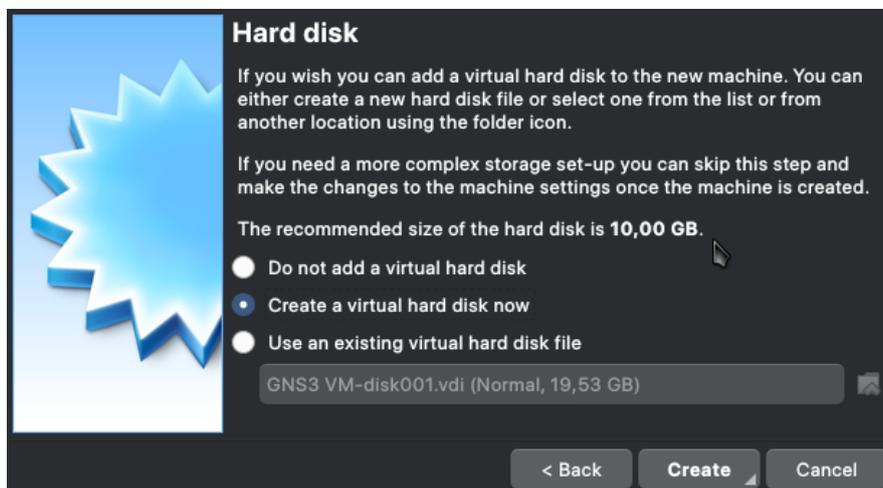
4. VirtualBox akan menampilkan Window baru untuk membuat Virtualisasi Ubuntu. Isi Nama dan Tipe Linux sesuai contoh di bawah. Lalu Klik **Next**. Cek Window berikut ini:



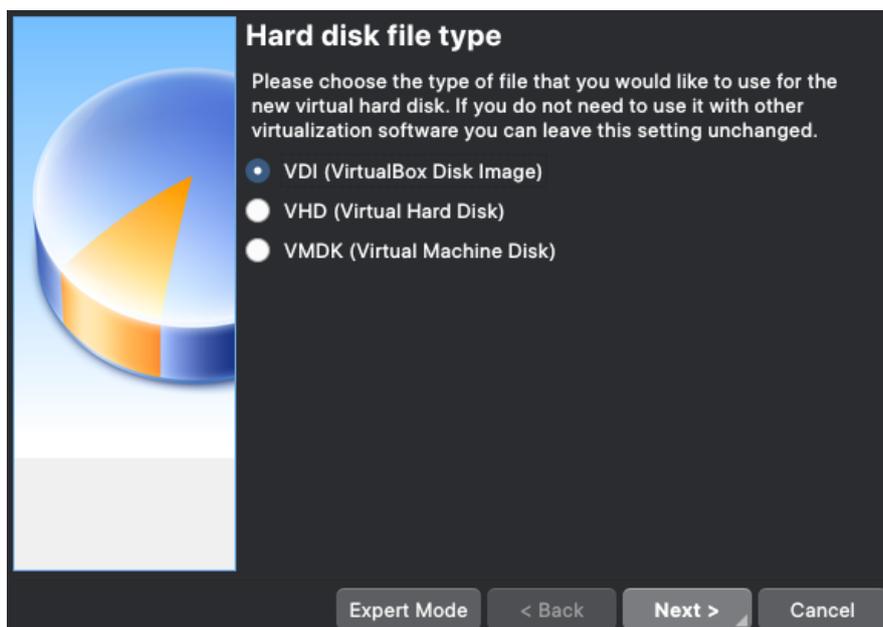
5. Berikutnya adalah mengatur memori RAM yang akan digunakan. Sangat direkomendasikan mengambil kurang dari **50%** dari total memori RAM. Jangan mengikuti nominal angka yang ada di materi ini. Lalu klik **Next**



6. Buat Hard disk untuk menyimpan sistem operasi. Klik **Create**



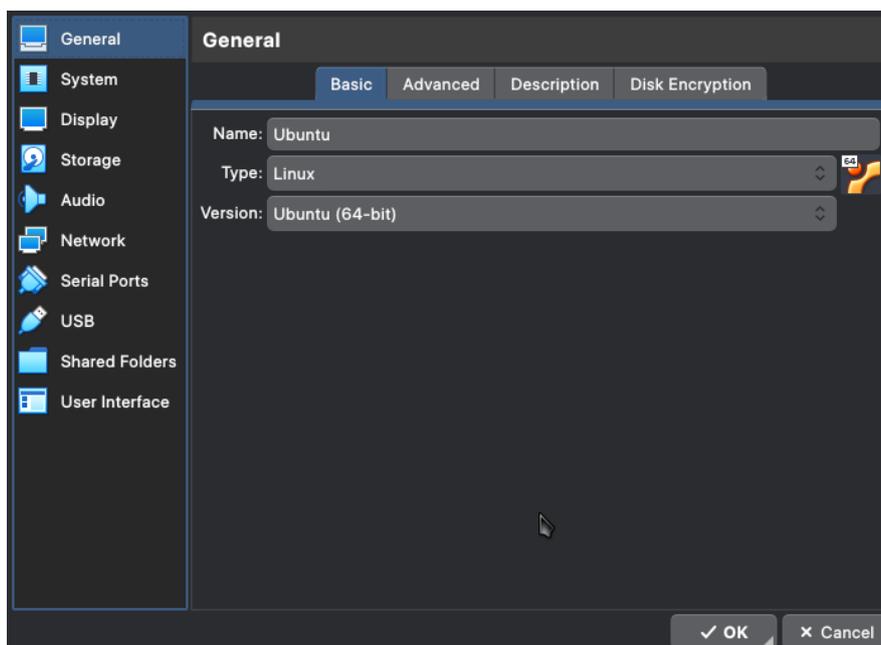
7. Pilih **VDI**, lalu klik **Next**



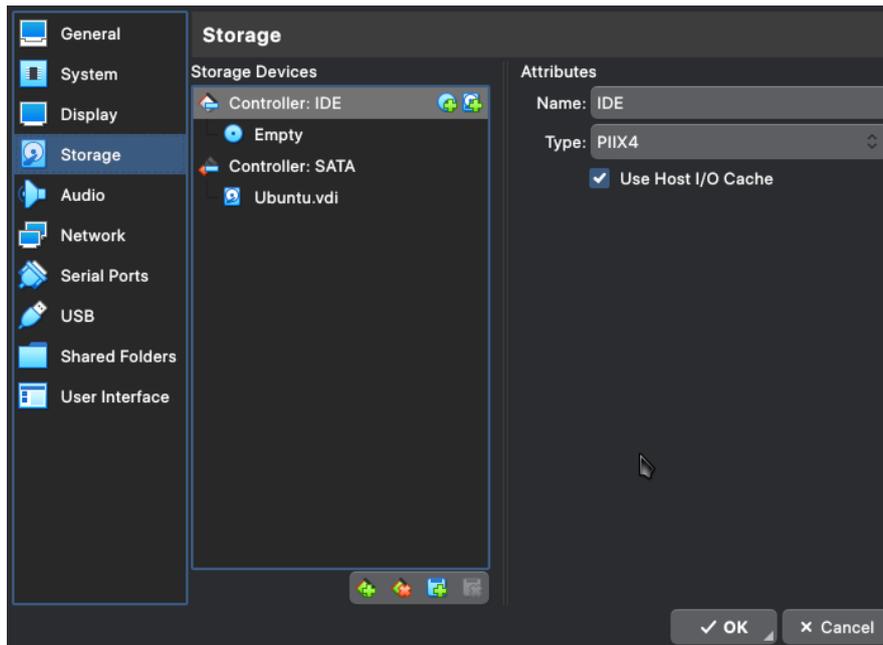
- Pilih **Dynamically allocated** agar hemat ruang hard disk di fisik komputer. Klik **Next**



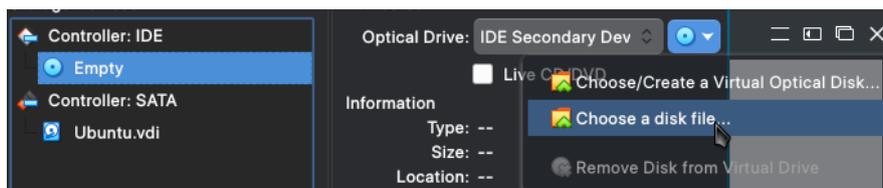
- Masukkan **30GB** di window ukuran harddisk. Lalu klik **Create**
- VirtualBox akan mengakhiri setup, dan berikutnya adalah mengkonfigurasi Virtualisasi. **JANGAN DOUBLE KLIK Virtualisasi yang dibuat**
- Untuk mengubah konfigurasi, klik kanan VirtualBox Ubuntu dan pilih **Settings**
- VirtualBox akan menampilkan Window **Settings** Ubuntu



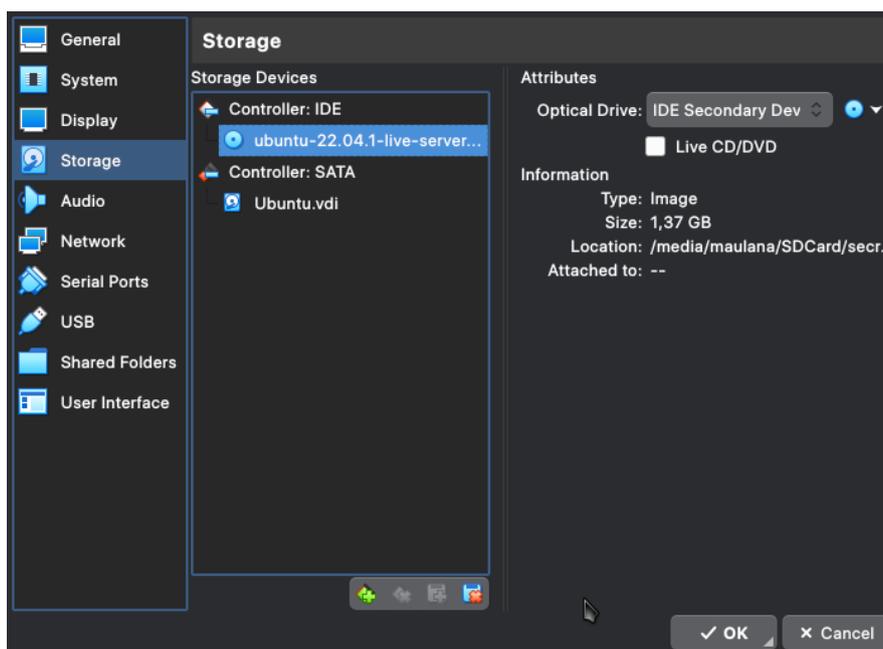
- Pilih **Storage** dan klik **Empty** icon **CD** yang berada di bawa **Controller: IDE**



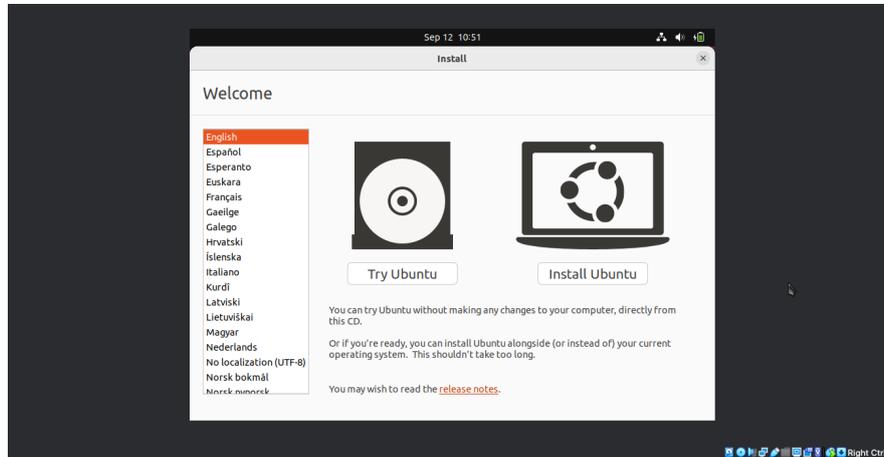
14. Klik simbol **CD** yang ada di sebelah **IDE Secondary Device**, dan pilih **Choose a disk file**



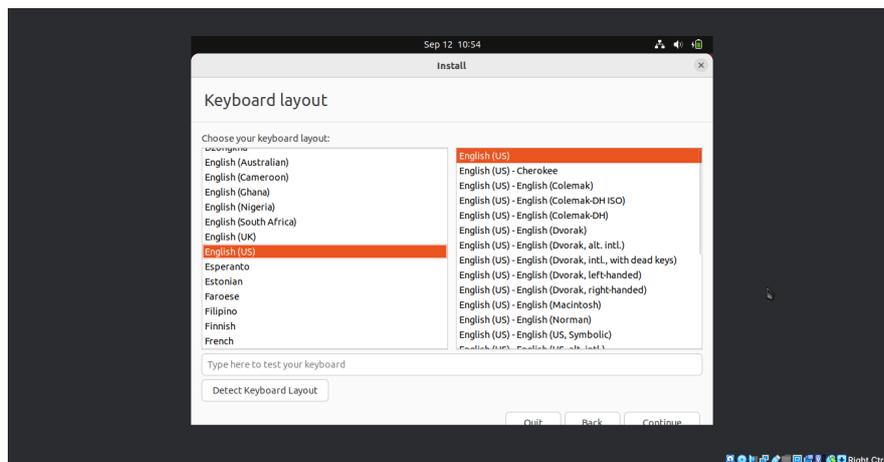
15. Pilih **ISO** yang sudah diunduh dan klik **OK**. Window akan menampilkan ISO yang sudah dipilih



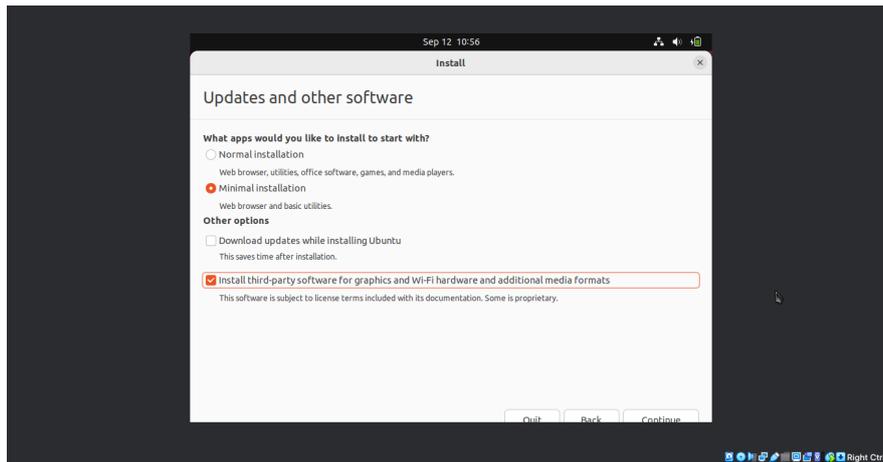
16. Klik **OK** untuk menyimpan, dan Virtualisasi siap diaktifkan
17. Klik **Start** atau **Double Klik** Virtualisasi yang ingin dinyalakan
18. Sistem akan menyalakan sistem dan tunggu hingga mendapatkan tampilan seperti berikut:



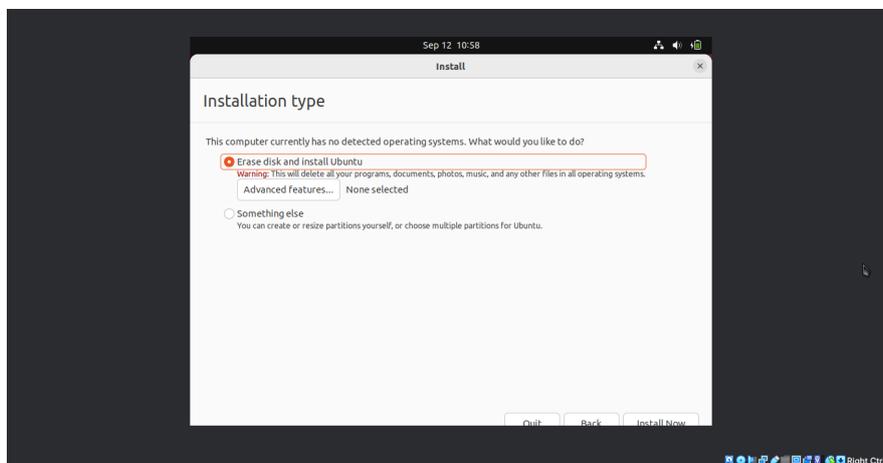
19. Klik **Install Ubuntu** untuk memulai instalasi sistem operasi ini
20. Klik **Continue** untuk memilih bahasa secara default



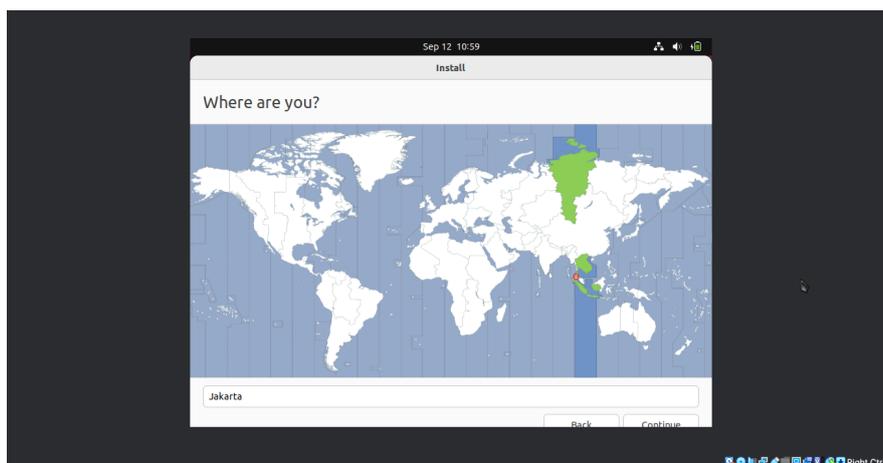
21. Pilih **Minimal Installation**, hilangkan **Download updates**, dan centang **Third-party** seperti gambar berikut lalu klik **Continue**



22. Pilih **Erase Disk and Install Ubuntu**, lalu klik **Install Now** dan klik **Continue**



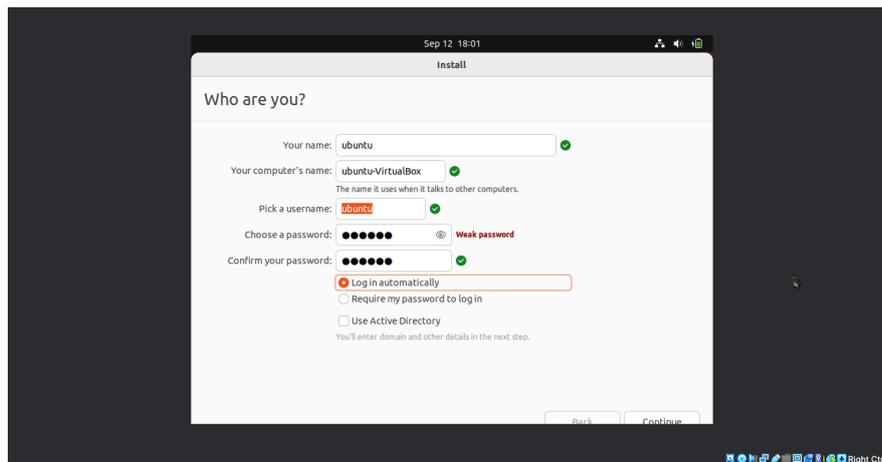
23. Pilih Zona Waktu, dan Pastikan mengarah ke **Jakarta**



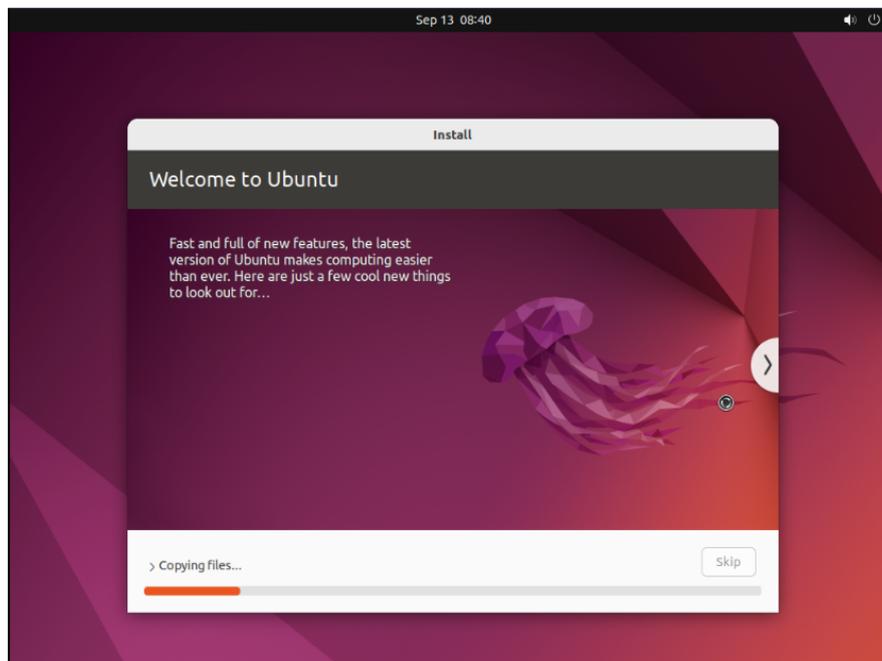
24. Lalu untuk konfigurasi user gunakan berikut, lalu klik **Continue**

- username : ubuntu
- password : ubuntu

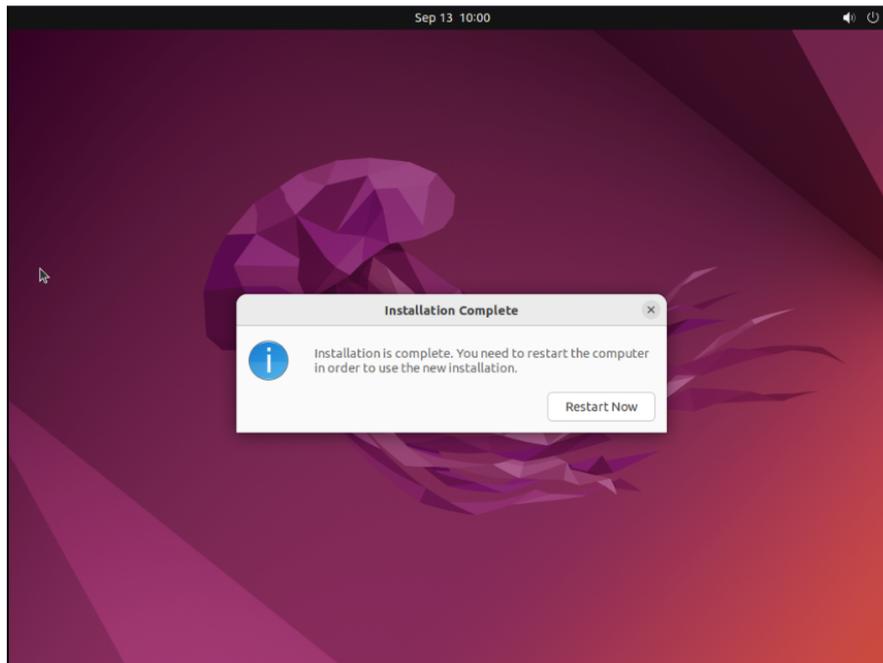
- Login : Automatically



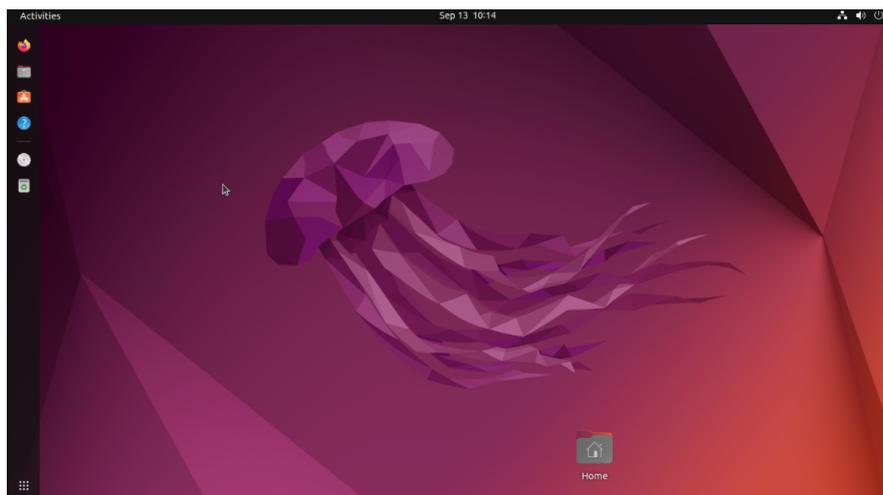
25. Tunggu proses instalasi hingga selesai. Cek gambar berikut



26. Klik **Restart** untuk menyalakan Ubuntu Linux yang sudah diinstall



27. Jika instalasi sukses, Ubuntu akan menampilkan tampilan desktopnya.



28. Praktikum 7 Selesai

# Bab 10

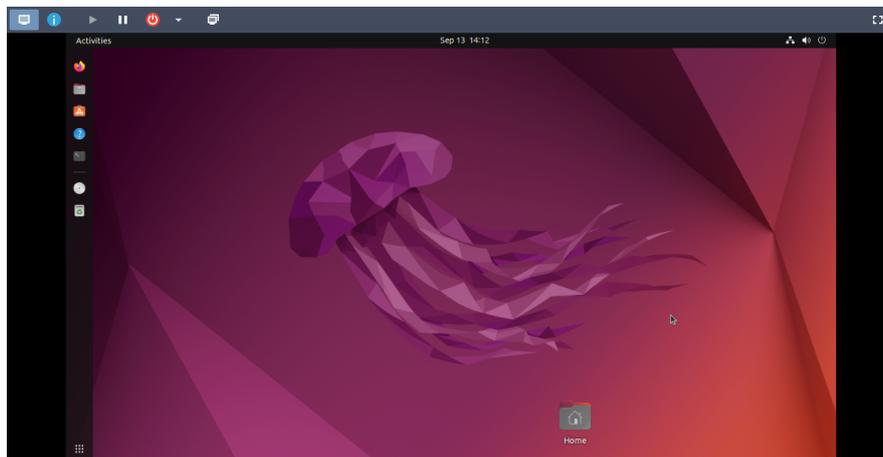
## Praktikum 10

### 10.1 Operasi Perintah Linux Ubuntu

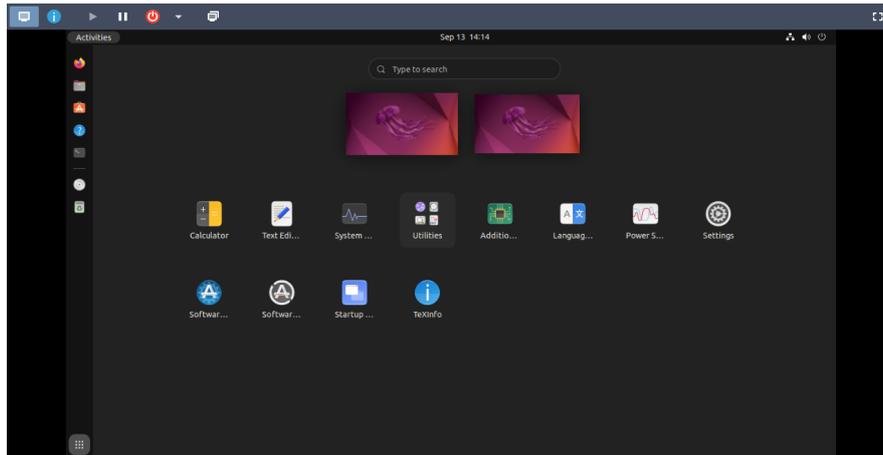
Di bagian ini mahasiswa diajarkan bagaimana melakukan operasi-operasi dasar dengan menggunakan perintah berbasis teks atau yang sering dikenal sebagai *Command Line Interface*. Mahasiswa diharapkan dapat mengoperasikan Linux secara umum dengan menggunakan perintah-perintah terminal saja.

### 10.2 Tutorial

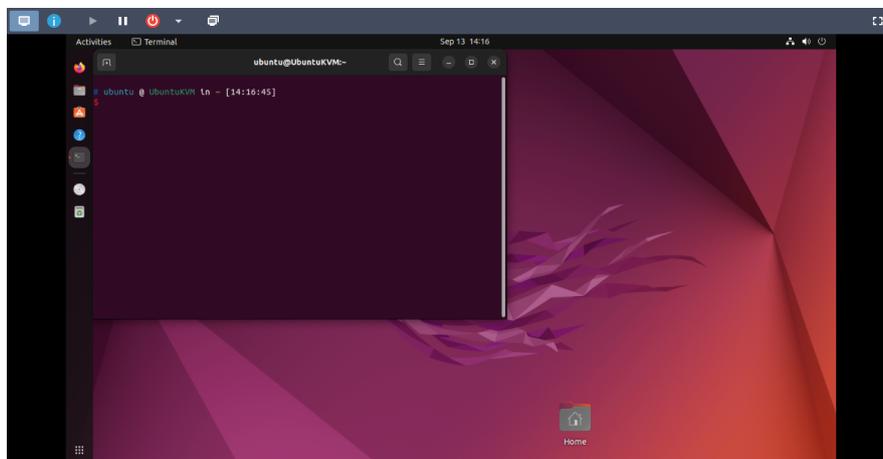
1. Nyalakan VirtualBox termasuk Virtualisasi Ubuntu
2. Tunggu hingga perangkat sukses booting dan menampilkan desktopnya



3. Klik **Menu** yang bisa diakses di bagian pojok kiri bawah berbentuk titik sembilan



4. Cari **Terminal** dengan mengetiknya di bagian atas pencarian. Lalu klik aplikasi **Terminal** tersebut. Maka akan muncul aplikasi seperti berikut:



5. Praktikum bisa dimulai dengan aplikasi ini.
6. Perintah sederhana untuk **menampilkan folder dan file** yang ada di folder saat ini adalah
  - `ls` → *LIST* (Daftar File dan Folder)
  - `dir` → *DIRECTORY* (Daftar File dan Folder)

```
# ubuntu @ UbuntuKVM in ~ [14:28:48]
$ ls
Desktop    Downloads  Pictures    snap        Videos
Documents  Music      Public     Templates

# ubuntu @ UbuntuKVM in ~ [14:28:50]
$ dir
Desktop    Downloads  Pictures    snap        Videos
Documents  Music      Public     Templates
```

7. Perintah yang digunakan untuk **masuk ke satu folder** dapat menggunakan perintah

- `cd` → *CHANGE DIRECTORY* (Ganti Folder)
- `cd [Nama Folder]` (Ganti ke Folder Tujuan)
- `cd ..` (Keluar dari Folder Saat Ini)

```
# ubuntu @ UbuntuKVM in ~ [14:33:50]
$ cd Documents

# ubuntu @ UbuntuKVM in ~/Documents [14:33:52]
$
```

8. Untuk **mengecek lokasi aktif** saat ini dengan menggunakan perintah

- `pwd` → *PRINT WORKING DIRECTORY* (Tampilkan Lokasi Saat Ini)

```
# ubuntu @ UbuntuKVM in ~/Documents [14:40:14]
$ pwd
/home/ubuntu/Documents

# ubuntu @ UbuntuKVM in ~/Documents [14:40:16]
$
```

9. Untuk **membuat folder baru** di dalam folder tersebut, gunakan perintah

- `mkdir` → *MAKE DIRECTORY* (Buat Folder Baru)
- `mkdir [Nama Folder]`

```
# ubuntu @ UbuntuKVM in ~/Documents [14:42:28]
$ mkdir Praktikum

# ubuntu @ UbuntuKVM in ~/Documents [14:42:33]
$
```

10. Gunakan perintah `ls` untuk melihat folder baru tersebut.

11. Untuk **membuat file baru**, gunakan perintah berikut untuk membuat file teks kosong

- `touch` (Buat File atau Perbarui Status File)
- `touch Praktikum.txt` (Jika tidak ada file tersebut, maka perintah akan membuat file `Praktikum.txt`)

```
# ubuntu @ UbuntuKVM in ~/Documents [14:45:37]
$ touch Praktikum.txt

# ubuntu @ UbuntuKVM in ~/Documents [14:45:41]
$ ls
Praktikum  Praktikum.txt
```

12. Untuk **mengecek status file**, gunakan perintah berikut:

- stat → STATUS (Cek Status File)
- stat [Nama File/Folder]

```
# ubuntu @ UbuntuKVM in ~/Documents [14:51:57]
$ stat Praktikum
File: Praktikum
Size: 4096          Blocks: 8          IO Block: 4096   directory
Device: fc03h/64515d Inode: 1319616    Links: 2
Access: (0775/drwxrwxr-x)  Uid: ( 1000/  ubuntu)   Gid: ( 1000/  ubuntu)
Access: 2022-09-13 14:42:33.129644800 +0700
Modify: 2022-09-13 14:42:33.105643203 +0700
Change: 2022-09-13 14:42:33.105643203 +0700
Birth: 2022-09-13 14:42:33.105643203 +0700
```

13. Untuk **memindahkan file atau folder ke satu tempat**, gunakan perintah

- mv → MOVE (Pindahkan)
- mv [Nama File/Folder] [Folder Tujuan]

```
# ubuntu @ UbuntuKVM in ~/Documents [14:57:02]
$ mv Praktikum.txt Praktikum

# ubuntu @ UbuntuKVM in ~/Documents [14:57:10]
$ ls
Praktikum
```

14. Untuk **menghapus file atau folder**, gunakan perintah

- rm → REMOVE (Hapus)
- rm [Nama File]
- rm -rf [Nama Folder]

```
# ubuntu @ UbuntuKVM in ~/Documents [14:59:46]
$ rm -rf Praktikum

# ubuntu @ UbuntuKVM in ~/Documents [14:59:50]
$ ls
```

15. Untuk **mengecek waktu dan tanggal yang berjalan di sistem**, gunakan perintah

- date → DATE (Tanggal dan Jam)

```
# ubuntu @ UbuntuKVM in ~/Documents [15:01:12]
$ date
Sel 13 Sep 2022 03:01:13 WIB
```

16. Untuk **mengecek ukuran suatu file atau folder**, gunakan perintah

- du → DISK USAGE (Cek Penggunaan Disk Tertentu)
- du -h [Nama File/Folder]
- du -sh [Name Folder]

```
# ubuntu @ UbuntuKVM in ~/Documents [15:07:17] C:1
$ du -h /home/ubuntu/Documents
4,0K    /home/ubuntu/Documents
```

17. Untuk **melihat penggunaan disk saat ini** dapat menggunakan perintah berikut

- df → DISK FREE (Cek Penggunaan Disk)
- df -h

```
# ubuntu @ UbuntuKVM in ~/Documents [15:11:27]
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
tmpfs           795M  1,7M  794M   1% /run
/dev/vda3       29G   15G   13G  55% /
tmpfs           3,9G   0    3,9G   0% /dev/shm
tmpfs           5,0M   4,0K  5,0M   1% /run/lock
/dev/vda2       512M   5,3M  507M   2% /boot/efi
tmpfs           795M   96K  795M   1% /run/user/1000
/dev/sr0        3,6G   3,6G   0 100% /media/ubuntu/Ubuntu 22.04.1 LTS amd64
```

18. Perintah Linux mendukung teknologi *pipelining* yang mampu melanjutkan **output** suatu perintah dan menjadi **input** di perintah berikutnya dengan tanda | (garis vertikal).

19. Contoh Perintah:

- stat Documents | grep Device
- → stat Documents (Menampilkan Status Folder Documents)
- → grep Device (Filter Teks Device)

```
# ubuntu @ UbuntuKVM in ~ [15:20:24]
$ stat Documents
  File: Documents
  Size: 4096          Blocks: 8          IO Block: 4096   directory
Device: fc03h/64515d Inode: 393910     Links: 2
Access: (0755/drwxr-xr-x)  Uid: ( 1000/  ubuntu)   Gid: ( 1000/  ubuntu)
Access: 2022-09-13 14:59:51.353469907 +0700
Modify: 2022-09-13 14:59:50.181625266 +0700
Change: 2022-09-13 14:59:50.181625266 +0700
 Birth: 2022-09-08 12:28:58.316000000 +0700

# ubuntu @ UbuntuKVM in ~ [15:20:25]
$ stat Documents | grep Device
Device: fc03h/64515d Inode: 393910     Links: 2
```

20. Praktikum 8 Selesai