LAPORAN PRAKTIKUM

MATA KULIAH REKAYASA PERANGKAT LUNAK

GIT LANJUT BAGIAN IV

DISUSUN OLEH

L0122061 – FARRELLY THEO ARIELA

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI DAN SAINS DATA

UNIVERSITAS SEBELAS MARET

2024

PERCOBAAN

1. Perbedaan Git Checkout, Git Reset, dan Git Revert

- **Git Checkout**

Perintah **git checkout** bisa dikatakan seperti mesin waktu, dengan perintah ini dapat dilakukan pengembalian kondisi file proyek seperti waktu yang dituju. Seperti waktu yang dituju disini dapat diketahui edngan menggunakan perintah **git log** untuk mengetahui nomer *commit*.

```
C:\Users\Danang Apri\Documents\tugas\PRPL11>git log
commit b91620a7fdece0b62e49f4d74ad25ef54939f3f9 (HEAD -> master)
Merge: d89d1f1 2ad5a1b
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 17:02:44 2024 +0700

    benerin konflik

commit d89d1f17e8d94da653093250d3cb51e6a14cfc64
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 17:00:18 2024 +0700

    konflik

commit 2ad5a1b1d91f9eb6becf5b483aab28ee97deb17a
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:58:33 2024 +0700

    conflict solving

commit 29b0bc960739286f1bdfa77ed638cc61f96aad9b
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:55:22 2024 +0700

    adding login page

commit 5d761ab619580337924919112699051307666 4cf (origin/master, origin/HEAD, editing_style)
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:48:40 2024 +0700

:...skipping...
commit b91620a7fdece0b62e49f4d74ad25ef54939f3f9 (HEAD -> master)
Merge: d89d1f1 2ad5a1b
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 17:02:44 2024 +0700

    benerin konflik

commit d89d1f17e8d94da653093250d3cb51e6a14cfc64
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 17:00:18 2024 +0700

    konflik

commit 2ad5a1b1d91f9eb6becf5b483aab28ee97deb17a
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:58:33 2024 +0700

    conflict solving
```

Misalnya praktikan ingin mengembalikan semua file seperti keadaan pada nomer *commit* **5d761ab619580337924919112699051307666 4cf**, untuk melakukannya dapat dilakukan dengan menjalankan perintah **git checkout 5d761ab619580337924919112699051307666 4cf**.

```
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git checkout 5d761ab6195803379249191126990513076664cf
Note: switching to '5d761ab6195803379249191126990513076664cf'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 5d761ab editing title again
PS C:\Users\Danang Apri\Documents\tugas\PRPL11>
```

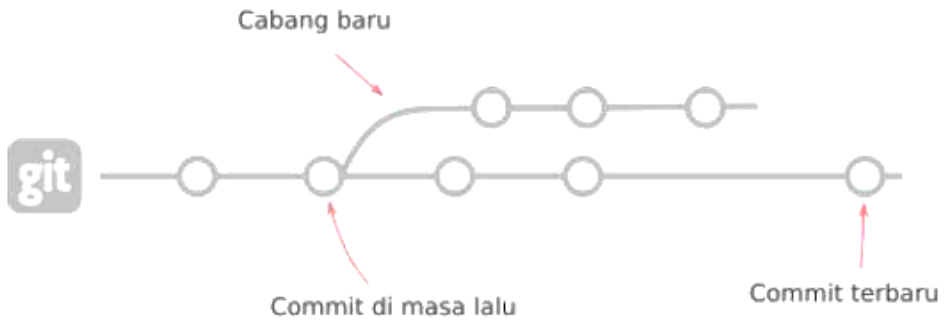
Dengan menjalankan perintah tersebut, dapat dilihat bahwa semua file telah dikembalikan seperti keadaan pada nomer *commit* tersebut.

Akan tetapi, hal ini bersifat temporer (sementara). Pengembalian ini tidak disimpan dalam database Git. Sehingga dapat dikatakan bahwa perintah **git checkout** merupakan perintah untuk mengecek kondisi file di setiap *commit*.

Selain untuk mengembalikan kondisi file proyek seperti waktu yang dituju, **git checkout** juga dapat digunakan untuk pembuatan cabang baru berdasarkan kondisi kode di masa lalu. Untuk melakukan hal tersebut dapat dilakukana dengan menggunakan perintah **git checkout -b nama_cabang <nomer_commit>**. Contohnya

```
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git checkout -b checking 5d761ab6195803379249191126990513076664cf
Switched to a new branch 'checking'
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git branch
* checking
  editing_style
  master
PS C:\Users\Danang Apri\Documents\tugas\PRPL11>
```

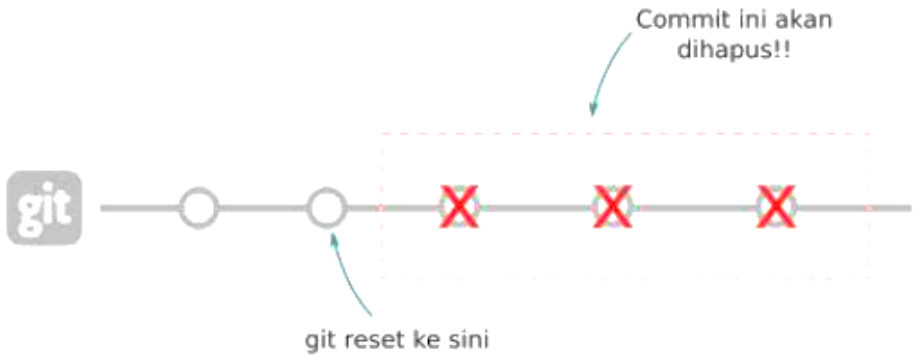
Dapat dilihat pada gambar diatas bahwa terbuat cabang baru dimana isi file nya sama seperti nomer *commit* yang di maksud.



Cara ini bisa di ibaratkan seperti menulis cerita baru dengan plot yang berbeda.

- **Git Reset**

  Perintah **git reset** sering disebut sebagai perintah berbahaya yang dapat menghancurkan catatan sejarah perubahan.



Perintah ini memiliki tiga argumen atau opsi utama, yaitu –soft, --mixed, dan –hard.

--soft akan mengembalikan kondisi file dalam keadaan *staged*

--mixed akan mengembalikan kondisi file dalam keadaan *modified*

--hard akan mengembalikan kondisi file dalam keadaan *commited*

Berikut contoh penggunaan **git reset**

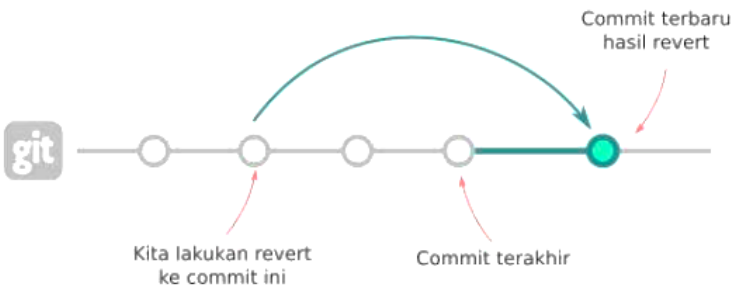
```
C:\Users\Danang Apri\Documents\tugas\PRPL11>git log
commit 1e47f7cca3f872d23a654a66dac26dd3761acfcb (HEAD -> checking)
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 18:19:58 2024 +0700

    test

commit 5d761ab619580337924919112699051307666 4cf (origin/master, origin/HEAD, editing_style)
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:48:40 2024 +0700

    editing title again

commit 80661ba0f820011b41d315bde903c98083ddabd1
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:41:01 2024 +0700

    editing title

commit 4ead534b29e230f46d5b459b37ae6fd0e69064e8
Author: Pr1nzx <danangapri30@gmail.com>
```

```
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git reset --soft 80661ba0f820011b41d315bde903c98083ddabd1
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git log
commit 80661ba0f820011b41d315bde903c98083ddabd1 (HEAD -> checking)
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:41:01 2024 +0700

    editing title

commit 4ead534b29e230f46d5b459b37ae6fd0e69064e8
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sat Jun 15 02:24:24 2024 +0700

    editing index

commit 67281f7b42935674f5bc8e2b51710f12a7080208
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Fri Jun 14 23:53:11 2024 +0700
```

Dapat dilihat pada gambar diatas bahwa file dalam keadaan *staged*.

- **Git Revert**

Revert artinya mengembalikan. Perintah ini lebih aman daripada **git reset**, karena tidak akan menghapus catatan sejarah revisi. *Revert* akan akan mengambil kondisi file yang ada di masa lalu, kemudian menggabungkannya dengan commit terakhir.



Sebelum menjalankan perintah **git revert**, perlu untuk mengetahui nomor *commit* yang akan dituju dengan menggunakan perintah **git log**.

```
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git reset --soft 5d761ab6195803379249191126990513076664cf
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git log
commit 5d761ab6195803379249191126990513076664cf (HEAD -> checking, origin/master, origin/HEAD, editing_style)
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:48:40 2024 +0700

    editing title again

commit 80661ba0f820011b41d315bde903c98083ddabd1
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:41:01 2024 +0700

    editing title

commit 4ead534b29e230f46d5b459b37ae6fd0e69064e8
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sat Jun 15 02:24:24 2024 +0700

    editing index

commit 67281f7b42935674f5bc8e2b51710f12a7080208
Author: Pr1nzx <danangapri30@gmail.com>
```

Kemudian, dijalankan perintah **git revert**. Pada percobaan ini, dilakukan Namun, terdapat *error* karena revert akan menimpa local change , maka dari itu harus di commit terlebih dahulu.

```
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git revert ce4f78a049f16b13ab42d0a8b1b48fdaab45bceb
error: your local changes would be overwritten by revert.
hint: commit your changes or stash them to proceed.
fatal: revert failed
PS C:\Users\Danang Apri\Documents\tugas\PRPL11>
```

```
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git add login.html
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git commit -m "test revert"
[checking 21bd58c] test revert
 1 file changed, 5 insertions(+)
 create mode 100644 login.html
PS C:\Users\Danang Apri\Documents\tugas\PRPL11>
```

Setelah *error* teratasi maka dapat dilakukan *commit*.

```
C:\Users\Danang Apri\Documents\tugas\PRPL11>git log
commit 21bd58c78bdc871eb6ffb56506011e7a62a6b695 (HEAD -> checking)
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 18:26:43 2024 +0700

    test revert

commit 5d761ab6195803379249191126990513076664cf (origin/master, origin/HEAD, editing_style)
Author: Pr1nzx <danangapri30@gmail.com>
Date:   Sun Jun 16 16:48:40 2024 +0700

    editing title again

commit 80661ba0f820011b41d315bde903c98083ddabd1
```

Apabila dilakukan pengecekan dengan **git log** maka dapat terlihat bahwa *commit* sebelumnya tidak terhapus.

2. Studi Kasus : Kapan Waktu yang Tepat Menggunakan Git Pull dan Git Fetch

Perintah **git pull** dan **git fetch** adalah dua perintah untuk mengambil *commit* terbaru dari *remote repository*.

Perbedaanya perintah **git pull** dan **git fetch** :

- Perintah **git pull** akan mengambil *commit* terbaru lalu otomatis menggabungkan (*merge*) dengan *branch* yang aktif.
- Sedangkan **git fetch** akan mengambil *commit* saja. Perintah **git fetch** tidak akan langsung melakukan *merge*.

Perintah **git pull** dan **git fetch** dapat digunakan seperti berikut ini :

git pull origin master

git fetch origin master

Dimana artinya akan diambil *commit* dari *branch master* pada *repository remote*.

Origin adalah nama *remote*-nya.

Untuk penjelasan yang lebih detail, dapat dilakukan dengan membaca dokumentasi dengan mengetik perintah **git pull --help** atau **git fetch --help**.

- **git pull**

  Perintah *git pull* akan mengambil *commit* terbaru lalu otomatis menggabungkan (*merge*) dengan branch yang aktif.

  Misalnya dilakukan perubahan terhadap file index.html pada cabang master seperti berikut :

```
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git diff
diff --git a/index.html b/index.html
index 3daeadd..2a2d004 100644
--- a/index.html
+++ b/index.html
@@ -3,7 +3,7 @@
 <head>
     <meta charset="UTF-8">
     <meta name="viewport" content="width=device-width, initial-scale=1.0">
-    <title>Praktikum 12 git lanjutan
+    <title>Praktikum 13 git lanjutan
     </title>
 </head>
 <body>
PS C:\Users\Danang Apri\Documents\tugas\PRPL11>
```

Kemudian, dilakukan *commit* untuk perubahan tersebut.

```
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git checkout checking
Switched to branch 'checking'
PS C:\Users\Danang Apri\Documents\tugas\PRPL11>
```

Selanjutnya, pindah ke cabang **checking** yang telah dibuat sebelumnya kemudian dapat menjalankan perintah **git pull origin master**.

```
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git pull origin master
From https://github.com/Pr1nzx/PRPL11
 * branch            master     -> FETCH_HEAD
Already up to date.
PS C:\Users\Danang Apri\Documents\tugas\PRPL11>
```

Terlihat pada gambar di atas bahwa ketika menjalankan perintah **git pull origin master** akan diambil *commit* terbaru di cabang **master** dan otomatis menggabungkannya dengan cabang yang aktif sekarang. Namun, perintah tersebut cocok digunakan apabila tidak ada commit yang pernah dilakukan di lokal.

- **git fetch**

Perintah **git fetch** akan mengambil *commit* saja tanpa melakukan *merge*.

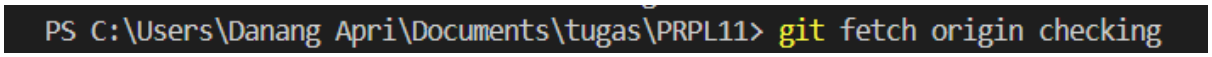
Misalnya dilakukan perubahan terhadap file index.html pada cabang checking seperti berikut :

```
Already up to date.
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git diff
diff --git a/index.html b/index.html
index 3daeadd..3cf19dc 100644
--- a/index.html
+++ b/index.html
@@ -3,7 +3,7 @@
 <head>
     <meta charset="UTF-8">
     <meta name="viewport" content="width=device-width, initial-scale=1.0">
-    <title>Praktikum 12 git lanjutan
+    <title>Praktikum 13
     </title>
 </head>
 <body>
PS C:\Users\Danang Apri\Documents\tugas\PRPL11>
```

Kemudian, dilakukan *commit* untuk perubahan tersebut.

```
</body>
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git add .
PS C:\Users\Danang Apri\Documents\tugas\PRPL11> git commit -m "nyoba fetch"
[checking e410948] nyoba fetch
 1 file changed, 1 insertion(+), 1 deletion(-)
```

Selanjutnya, pindah ke cabang **master** yang telah dibuat sebelumnya kemudian jalankan perintah **git fetch origin Checking**.



Terlihat pada gambar di atas bahwa ketika menjalankan perintah **git fetch origin Checking** akan diambil *commit* terbaru di cabang **Checking** namun tidak dilakukan penggabungan yang aktif sekarang.

KESIMPULAN

Praktikum ini mengkaji perbedaan antara perintah Git Checkout, Reset, dan Revert, serta penggunaan yang tepat dari Git Pull dan Fetch. Git Checkout memungkinkan navigasi ke kondisi proyek pada commit tertentu, mirip mesin waktu, namun perubahan ini tidak permanen dalam database Git. Perintah ini juga bisa menciptakan cabang baru dari commit lama. Git Reset, meski sering dianggap berisiko karena dapat menghilangkan riwayat perubahan, mengembalikan file ke keadaan tertentu dengan tiga pilihan: staged, modified, atau committed. Git Revert menawarkan pendekatan yang lebih aman dengan memulihkan file ke commit sebelumnya tanpa menghapus riwayat, dengan mengadopsi kondisi file dari commit terdahulu dan mengintegrasikannya dengan commit terkini.

Terkait Git Pull dan Fetch, keduanya berfungsi untuk mengambil commit terbaru dari remote repository. Git Pull secara otomatis mengambil dan menggabungkan commit terbaru dengan branch aktif, ideal ketika tidak ada commit lokal yang perlu dipertahankan. Sebaliknya, Git Fetch hanya mengambil commit tanpa penggabungan langsung, memberi kesempatan untuk memeriksa perubahan sebelum melakukan merge. Pemahaman mendalam tentang kelima perintah Git ini krusial dalam manajemen perubahan repository.