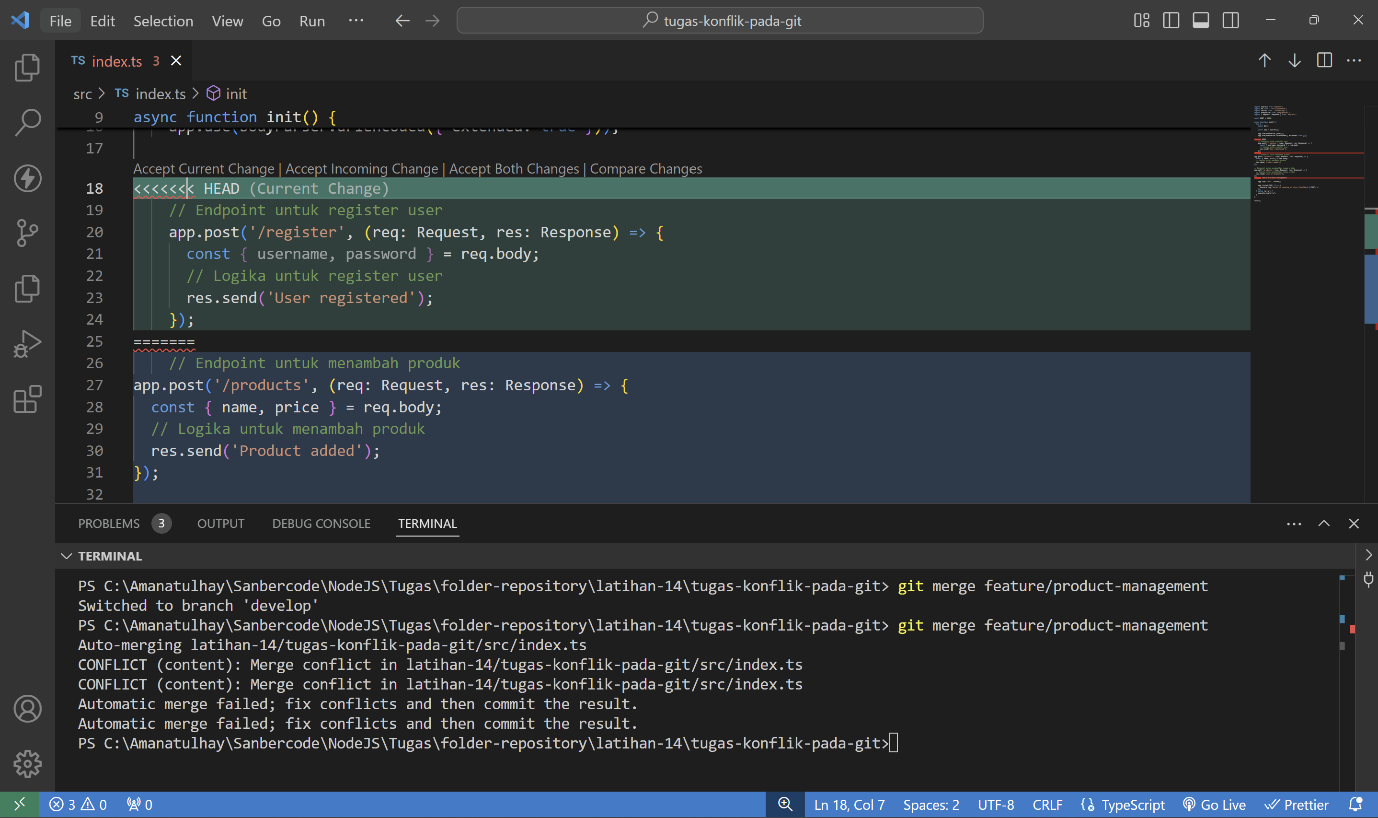
**Tugas 14: Konflik Pada Git**

NodeJS Backend Development with Xpress (Batch 62)

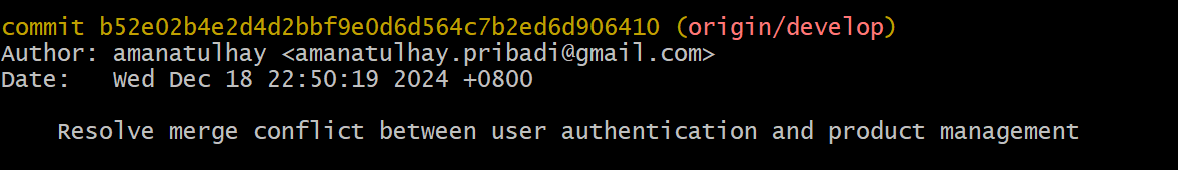
Nama: Amanatulhay Pribadi

**Latihan**

Saat mengikuti langkah-langkah yang diberikan, ditemui konflik saat melakukan merge:



Setelah memilih salah satu penyelesaian (yaitu dengan menggabungkan keduanya), add, commit dan push. Check kembali git log untuk memeriksa perubahan:



**Tugas 14: Konflik Pada Git**

NodeJS Backend Development with Xpress (Batch 62)

Nama: Amanatulhay Pribadi

**Soal Essay**

1. Bagaimana Anda akan mengatur branching untuk fitur-fitur yang berbeda seperti autentikasi pengguna, manajemen produk, dan pemrosesan pesanan? Jelaskan strategi branching yang akan Anda gunakan dan bagaimana Anda akan memastikan bahwa setiap fitur dikembangkan dan diintegrasikan dengan baik.

*Jawaban:*

Strategi branching untuk fitur-fitur yang berbeda

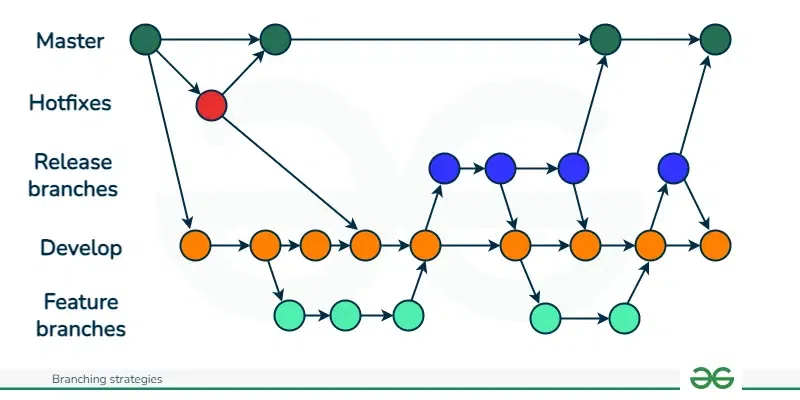
1. **Main/Master Branch** branch utama yang biasanya berisi kode yang siap diproduksi.
2. **Feature Branch** branch yang dibuat untuk mengembangkan fitur baru.
   1. feature/user-authentication
   2. feature/product-management
   3. feature/order-processing
3. **Bugfix Branch** branch yang dibuat untuk memperbaiki bug.
4. **Release Branch** branch yang dibuat untuk persiapan rilis produk.

Untuk memastikan setiap fitur dikembangkan dan diintegrasikan dengan baik, secara rutin lakukan pull/fetch sehingga branch up-to-date dengan perubahan terbaru dan tidak tertinggal jauh.

Selain metode di atas, terdapat berbagai macam strategi branching yang bisa diterapkan bergantung pada kompleksitas dan tenggat waktu project tersebut. Berdasarkan artikel yang ada pada GeeksforGeeks (Referensi: [*https://www.geeksforgeeks.org/branching-strategies-in-git/*](https://www.geeksforgeeks.org/branching-strategies-in-git/)), setidaknya ada empat strategi branching:

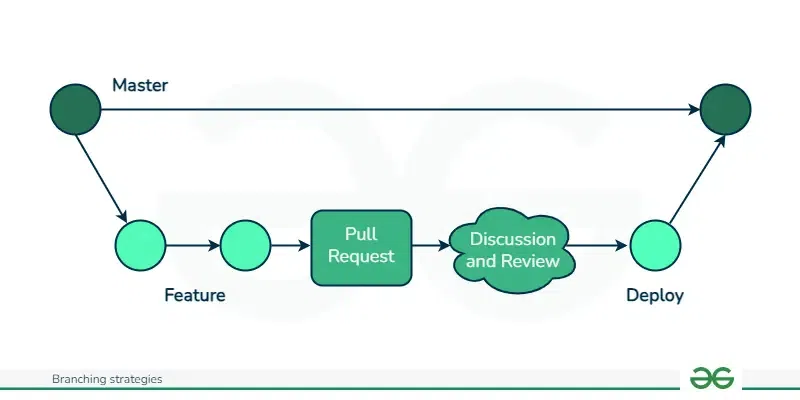
1. **Gitflow Workflow**

Gitflow memungkinkan development berjalan secara parallel di mana para developer dapat bekerja secara terpisah di feature branch. Branch master dan develop merupakan branch yang utama dan bertahan sepanjang project, namun branch yang lainnya hanya sementara dan pada dasarnya hanya untuk supporting project.



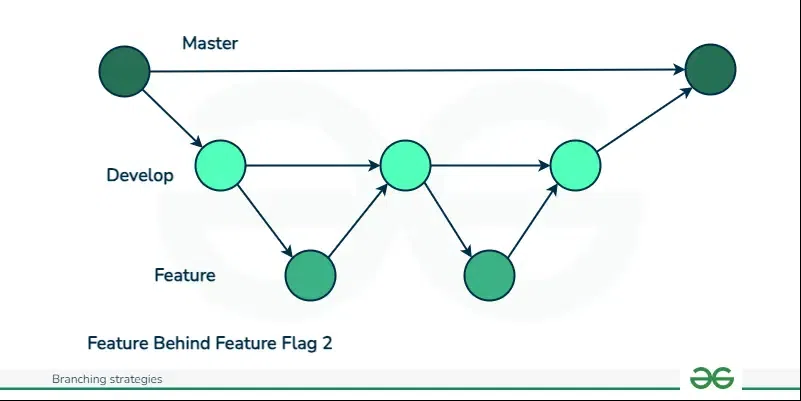
1. **Github Flow**

Gitflow merupakan alternatif yang lebih sederhana dibandingkan GitFlow, dan ideal untuk team berukuran kecil. Flow ini hanya memiliki branch feature yang bercabang langsung dari master dan merge kembali ke master begitu perubahannya selesai. Branch ini tidak memiliki release branch.



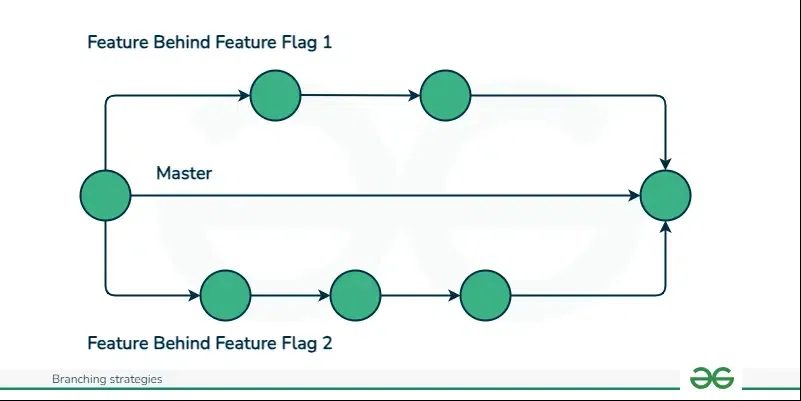
1. **Gitlab**

Gitlab flow juga merupakan alternatif dari GitFlow, namun lebih robust dan dapat diterapkan untuk skala project yang lebih besar dibandingkan GitHub Flow.



1. **Trunk Based Development**

Strategi ini cocok untuk project kecil dan workflow yang efisien. Developer bekerja di satu “trunk” branch, dan kebanyakan di master branch, dan menggunakan feature flag untuk mengisolir feature sampai mereka siap ke branch release. Flow ini membuat perubahan yang lebih kecil lebih sering untuk menghindari merge conflict dan membatasi branch yang berumur lama.



1. Pemilihan strategi:

Untuk membantu pertimbangan dalam pemilihan strategi, berikut table perbandingan keempat strategi yang telah disebutkan.

|  |  |  |
| --- | --- | --- |
| Tipe Product | Ukuran Team | Strategi yang Dapat Diterapkan |
| Deployment dan release yang kontinyu | Kecil | Github Flow dan TBD |
| Version release yang terjadwal dan periodik | Sedang | GitFlow dan Gitlab Flow |
| Deploymen yang kontinyu untuk produk yang berfokus pada kualitas | Sedang | Gitlab Flow |
| Produk dengan siklus pemeliharaan yang penjang | Besar | GitFlow |

1. Berikan contoh skenario dimana konflik terjadi saat menggabungkan dua branch fitur yang berbeda. Jelaskan langkah-langkah yang akan Anda ambil untuk menyelesaikan konflik tersebut, termasuk penggunaan alat atau metode yang dapat memudahkan proses resolusi konflik. Bagaimana Anda memastikan bahwa perubahan dari kedua branch berfungsi dengan baik setelah konflik diselesaikan?

*Jawaban:*

Contoh skenario konflik adalah seperti pada Latihan yaitu saat dua branch fitur (feature/product-management dan feature/product-management) menuliskan endpoint masing-masing pada file src/index.ts di line yang sama sehingga saat merge ke branch develop akan terjadi conflict.

Langkah yang diambil untuk menyelesaikannya adalah:

* Membuka salah satu tool yang memiliki fitur merge-conflict (contoh Visual Studio Code)
* Perhatikan adanya marker konflik. Pilih salah satu dari empat solusi:
  + Accept current change
  + Accept incoming change
  + Accept both changes
  + Compare changes
* Setelah solve, add dan commit file index.ts tersebut.
* Lakukan pengujian lalu merge ke develop

Untuk memastikan perubahan dari kedua branch berfungsi dengan baik setelah konflik, lakukan git log pada branch develop atau main untuk memeriksa apakah update dari kedua branch feature sudah ada.