**STAGE & SNAPSHOT**

Anlık görüntülerle ve Git aşama alanıyla çalışma:

* git status: Çalışma dizininde değiştirilen ve bir sonraki commit için aşamaya alınan dosyaları gösterir.
* git add [dosya]: Dosyayı, şu anki haliyle bir sonraki commit'e ekler (aşamaya alır).
* git reset [dosya]: Dosyayı aşamadan çıkarır, ancak çalışma dizinindeki değişiklikleri korur.
* git diff: Değiştirilen ancak aşamaya alınmayan kısımların farklarını gösterir.
* git diff --staged: Aşamaya alınmış ama henüz commit edilmemiş değişikliklerin farklarını gösterir.
* git commit -m “[açıklayıcı mesaj]”: Aşamaya alınan içerikleri yeni bir commit olarak kaydeder.

**SETUP**

Tüm yerel depolar için kullanılan kullanıcı bilgilerini ayarlama:

* git config --global user.name “[ad soyad]”: Commit geçmişinde görülecek şekilde tanımlanabilir bir ad ayarlayın.
* git config --global user.email “[geçerli e-posta]”: Commit geçmişinde kullanılacak bir e-posta adresi ayarlayın.
* git config --global color.ui auto: Git komut satırı için otomatik renkleme sağlar, böylece gözden geçirmek daha kolay olur.

**SETUP & INIT**

Kullanıcı bilgilerini ayarlama, depo başlatma ve klonlama:

* git init: Mevcut bir dizini Git deposu olarak başlatır.
* git clone [url]: Belirtilen URL'deki tüm depoyu yerel bir kopya olarak alır.

**BRANCH & MERGE**

Dallarda izole çalışma, bağlam değiştirme ve değişiklikleri birleştirme:

* git branch: Mevcut dalları listeler. Aktif dalın yanında bir (\*) bulunur.
* git branch [dal-adı]: Geçerli commit'te yeni bir dal oluşturur.
* git checkout [dal-adı]: Başka bir dala geçiş yapar ve o dalı çalışma dizinine getirir.
* git merge [dal]: Belirtilen dalın geçmişini mevcut dala birleştirir.
* git log: Geçerli dalın tüm commit geçmişini gösterir.

**SHARE & UPDATE**

Başka bir depodan güncellemeleri almak ve yerel depoları güncellemek:

* git remote add [takma-ad] [url]: Bir git URL'sini takma ad olarak ekler.
* git fetch [takma-ad]: Uzak depodaki tüm dalları indirir.
* git merge [takma-ad]/[dal]: Uzak bir dalı mevcut dalınıza birleştirir.
* git push [takma-ad] [dal]: Yerel dal commit'lerini uzak depoya gönderir.
* git pull: Takip edilen uzak daldan commit'leri indirir ve birleştirir.

**TRACKING PATH CHANGES**

Dosya kaldırma ve yol değişikliklerini sürümleme:

* git rm [dosya]: Projeden bir dosyayı siler ve bu silme işlemini commit için aşamaya alır.
* git mv [mevcut-yol] [yeni-yol]: Mevcut bir dosya yolunu değiştirir ve bu değişikliği aşamaya alır.
* git log --stat -M: Taşınan yollarla ilgili göstergeleri içeren tüm commit günlüklerini görüntüler.

**TEMPORARY COMMITS**

Geçici olarak değiştirilen, izlenen dosyaları saklama:

* git stash: Değiştirilen ve aşamaya alınan değişiklikleri kaydeder.
* git stash list: Kaydedilen değişikliklerin yığın sırasını listeler.
* git stash pop: Yığının en üstündeki değişiklikleri çalışma alanına yazar.
* git stash drop: Yığının en üstündeki değişiklikleri siler.

**REWRITE HISTORY**

Dalları yeniden yazma, commit'leri güncelleme ve geçmişi temizleme:

* git rebase [dal]: Mevcut dalın commit'lerini belirtilen dalın üzerine uygular.
* git reset --hard [commit]: Aşama alanını temizler, belirtilen commit'ten çalışma dizinini yeniden yazar.

**INSPECT & COMPARE**

Günlükleri, farkları ve nesne bilgilerini inceleme:

* git log: Geçerli aktif dalın commit geçmişini gösterir.
* git log branchB..branchA: branchA dalında olup branchB dalında olmayan commit'leri gösterir.
* git log --follow [dosya]: Bir dosyanın yeniden adlandırılması da dahil olmak üzere değişikliklerini gösterir.
* git diff branchB...branchA: branchA dalında olup branchB dalında olmayan farkları gösterir.
* git show [SHA]: Git'teki herhangi bir nesneyi insan tarafından okunabilir formatta gösterir.

**IGNORING PATTERNS**

Dosyaların istemeden aşamaya alınmasını veya commit edilmesini önleme:

* git config --global core.excludesfile [dosya]: Tüm yerel depolar için sistem genelinde yok sayma deseni ayarlar.

.gitignore dosyasında istenen desenleri şu şekilde kaydedin:

markdown

KopyalaDüzenle

logs/

\*.notes

pattern\*/