git log ==> Projeniz için oluşturduğunuz commit'lerin tarihçesini incelemek için

git pwd ==> Şuan çalışılan dosya görüntüleme (present working directory)

git commit -a -m "commit mesajı" ==> Add yaparak commit etme

git checkout ==> commitler arası geçiş yapma

git checkout master ==> master branchine geçiş yapar

git stash ==> Tamamlanmamış değişikliklerinizi kesinlikle commit etmemeye özen gösterin. Eğer zaman zaman değişikliklerinizi kayıt

altına almak isytiyorsanız commit işlemi yerine Git'in Stash özelliğini/komutunu kullanabilirsiniz.

git stash list ==> stash listesini gösterir

git stash pop ==> listenin en üstünde yer alan değişiklik geri yüklenecek ve bu değişiklik listeden silinecek.

git branch ==> branchleri gösterir

git branch frontend ==> frontend branch i oluşturur

git checkout frontend ==> frontend branchine geçer

git init ==> git reposu olusturur

git commit -m "message" ==> commit atmamızı sağlar

git add . ==> Yaptığımız değişiklikleri git'e eklememizi sağlar

git branch -M main ==> Ana branchin ismini main olarak değiştirdik

git remote add origin ".git uzantısı" ==> localdeki dosyaları uzak sunucuya ekler

git push -u origin main ==> Eklediğimiz dosyları githuba gönderme işlemi

git commit --amend -m "message" ==> son commitin mesajını değiştirir

git clone .git ==> Uzak sunucudaki bir repoyu locale getirmemizi sağlar

git stash save "stash_adi" ==> stashin adını değiştirmemizi sağlar

git stash apply stash@{0} ==>hashi verilen stashi geri yükler

git stash drop stash@{0} ==> hashi verilen stashi siler

git log --all --decorate --graph --oneline ==> yazmıs oldugumuz commit yapılarını bir satırda sıra ile gösterir

git config --global alias.kisa "git log --all --decorate --graph --oneline"

git kisa ==> Kisa bir şekilde yukarıdaki yapıyı calıstırmamızı saglar

git status ==> Şuanki durumumuzu gösterir

git branch --all ==> git branch -a ==> Kısaltılması

git branch -D branchadi ==> Adini belirtmiş olduğumuz branchi siler

git branch frontend ==> frontend branch i oluşturur

git checkout frontend ==> frontend branchine geçer

$mkdir frontend ==> Make Directory Yeni klasör oluşturmak için kullanıyoruz. Mac ve Windows'ta kullanımı aynıdır.

$cd frontend ==> Change Directory Bu komut ile dosyalar arası geçiş için kullanırız. Mesela frontend klasörüne geçmek için $cd frontend

$cat >> frontend.html ==> frontend.html dosyayı oluşturur.

git history ==> girilen git komutlarını listeler

git merge frontend ==> git merge komutu ile kaynak branch'deki commit edilmiş değişiklikler main branche entegre edilir.

git clone ".git uzantısı" ==>Githubda yer alan repoyu kendi bilgisayarımıza kopyalar

git config --global alias.push1 "git push -u origin master"

git push1 ==> Tırnak icerisinde belirttigim komutu artık push1 kısa yolu ile yapmamı saglar

git checkout -b backend ==>Backend branchini olustur ve o branche gecis yap

git merge --no-ff database => No fast-forward

NO FAST FORWARD

---

git checkout -b backend

cat >> deneme.txt

git add .

git commit -m "no fast forward"

git checkout main

git merge --no-ff backend
