GIT VERSİYON KONTROL SİSTEMİ

GIT KURSUMUZA HOŞGELDİNİZ!

GIT : BAŞLANGIÇ KURSU

GIT VERSİYON KONTROL SİSTEMİ

1 **BÖLÜM 1 :** GIT NEDİR?

2 **BÖLÜM 2 :** GIT KURULUMU

3 **BÖLÜM 3 :** TEMEL GIT İŞLEMLERİ

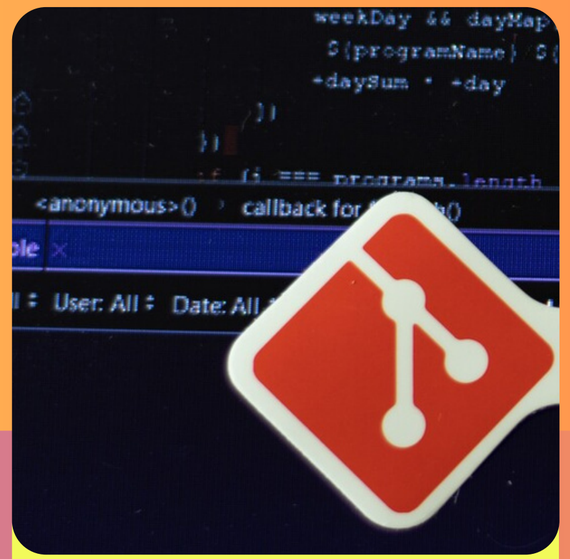
BÖLÜM 1 : GIT NEDİR?

Git, yazılım geliştirme süreçlerinde kullanılan bir versiyon kontrol sistemidir.

Git Versiyon Kontrol Sistemi, bir proje üzerinde birden çok kişinin çalışmasına ve her birinin kendi versiyonunu oluşturmasına daha sonra değişiklik yapılması istenildiğinde istenilen versiyona dönülüp oradan değişiklik yapılmasına olanak veren bir kontrol sistemidir..

GIT'IN TEMEL İLKELERİ

1. DAĞITIK YAPI
2. HIZLI VE VERİMLİ
3. PARALEL GELİŞTİRME
4. KOLAY GERİ ALMA VE KARŞILAŞTIRMA
5. DAL VE BİRLEŞTİRME (BRANCH VE MERGE)
6. İŞBİRLİĞİ VE PAYLAŞIM
7. GÜVENİLİRLİK



GIT'IN DİĞER VERSİYON KONTROL SİSTEMLERİNDEN FARKI

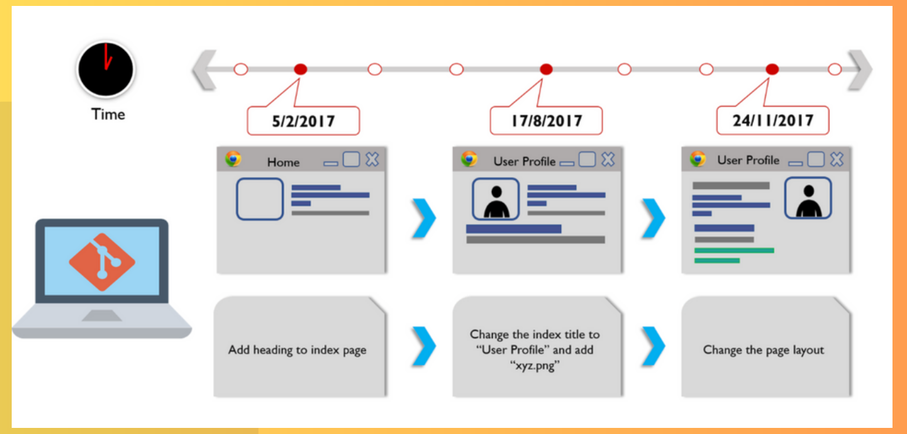
Dağıtık bir versiyon kontrol sistemidir.

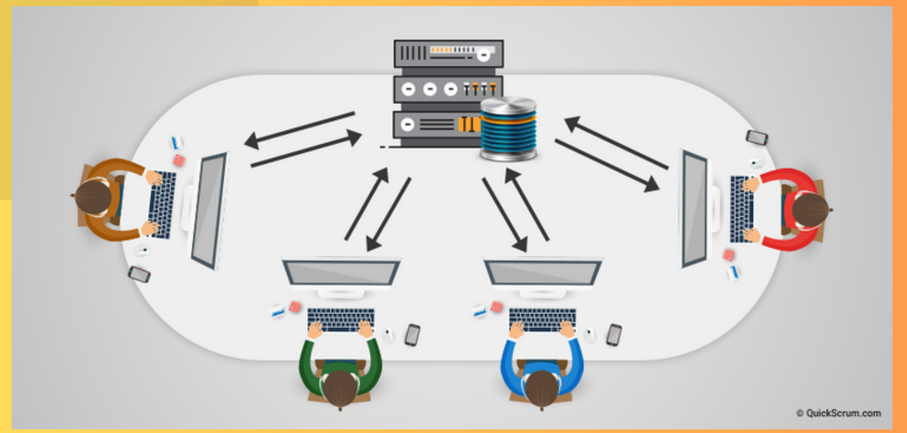
Git, hızlı ve performanslı bir sistemdir.

Dal(branch) yönetimini kolaylaştırır.

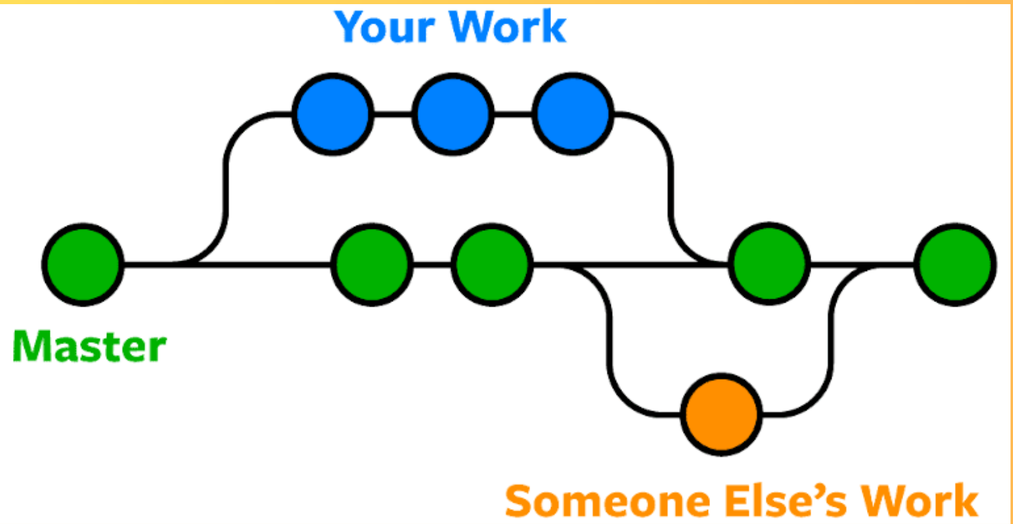
Geniş bir ekosistem ve aktif bir topluluğa sahiptir.

Projeleri uzak depolarda saklayabilir.





GIT'IN FAYDALARI



Aynı projede eş zamanlı çalışabilme imkanı sunar.

Git, çalıştığınız projelerde yer alan kodları.versiyonları, yazdığınız yorumları saklar.

Git, comment adı verilen açıklamalar girmenizi ister. Bu comment'ler sayesinde projenizin herhangi bir versiyonundaki değişikliklerin nedenlerini de kayıt altına alıp ihtiyaç halinde geri dönüp inceleyebilirsiniz.

BÖLÜM 2 : GIT KURULUMU

1. ADIM: Git'i kurun.

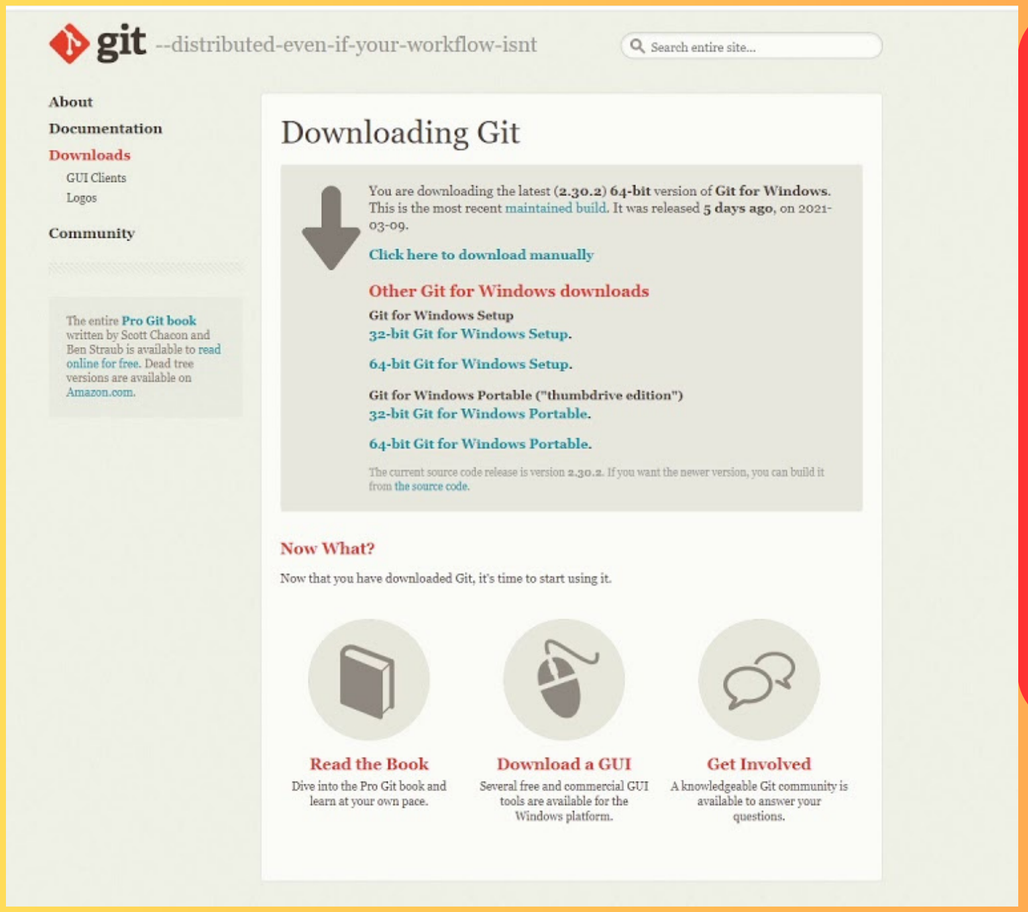
Bunun için **(https://git-scm.com/book/tr/v2/Ba%C5%9Flang%C4%B1%C3%A7-Git%E2%80%99i-Y%C3%BCklemek**) adresinden yararlanabilirsiniz.

2. ADIM: Git'i Yapılandırın.

Git yapılandırması, Git'in çalışma ortamınızda nasıl davranacağını ve yapılandırma seçeneklerini belirlemenizi sağlayan ayarları içerir. Git'i kendinize göre ayarlayın.

2. ADIM: Git'i Test Edin.

Git'i kullanmaya başlayın!



GIT VERSİYON KONTROL SİSTEMİ

GIT TERMİNOLOJİSİ

Repository (Depo)

Remote (Uzak)

Commit

Clone

Branch (Dal)

Staging Area

Merge

Pull (Çekme)

Pull Request

Push (İtme)

Git terminolojisi, Git'in kullanımında sıkça karşılaşılan ve anlamlandırılması gereken terimleri ifade eder.İşte terminolojide sıkça kullanılan bazı terimler:

GIT VERSİYON KONTROL SİSTEMİ

GIT İSTEMCİLERİ

Git istemcileri, Git'i kullanmanızı sağlayan yazılımlardır. Git, komut satırı tabanlı olarak kullanılabilen bir versiyon kontrol sistemi olduğundan, komutları doğrudan komut satırından girerek işlemleri gerçekleştirebilirsiniz. Ancak, bazı kullanıcılar için komut satırı kullanımı zor olabilir veya daha kullanıcı dostu bir arayüz tercih edebilirler.

GIT VERSİYON KONTROL SİSTEMİ

GIT İSTEMCİLERİ

1. GRAFİK ARAYÜZ TABANLI GIT İSTEMCİLERİ

   GITHUB DESKTOP

   SOURCE TREE

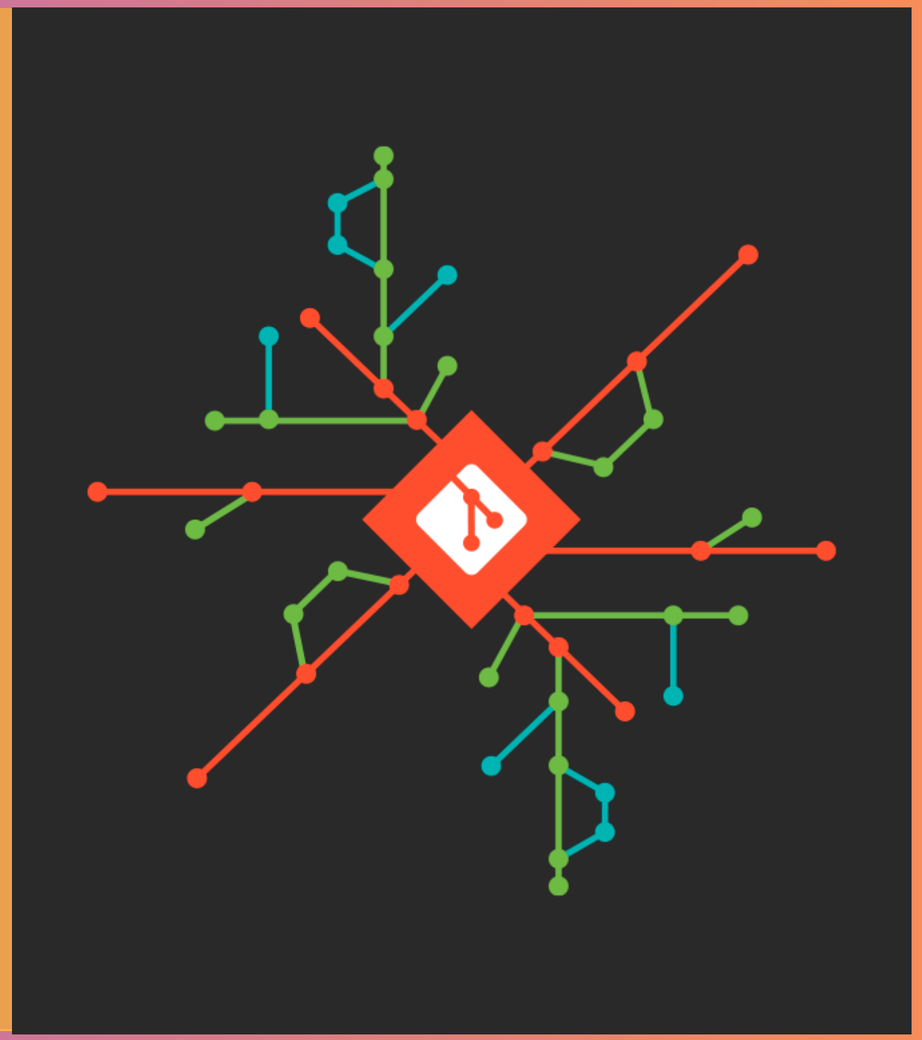
   GITKRAKEN

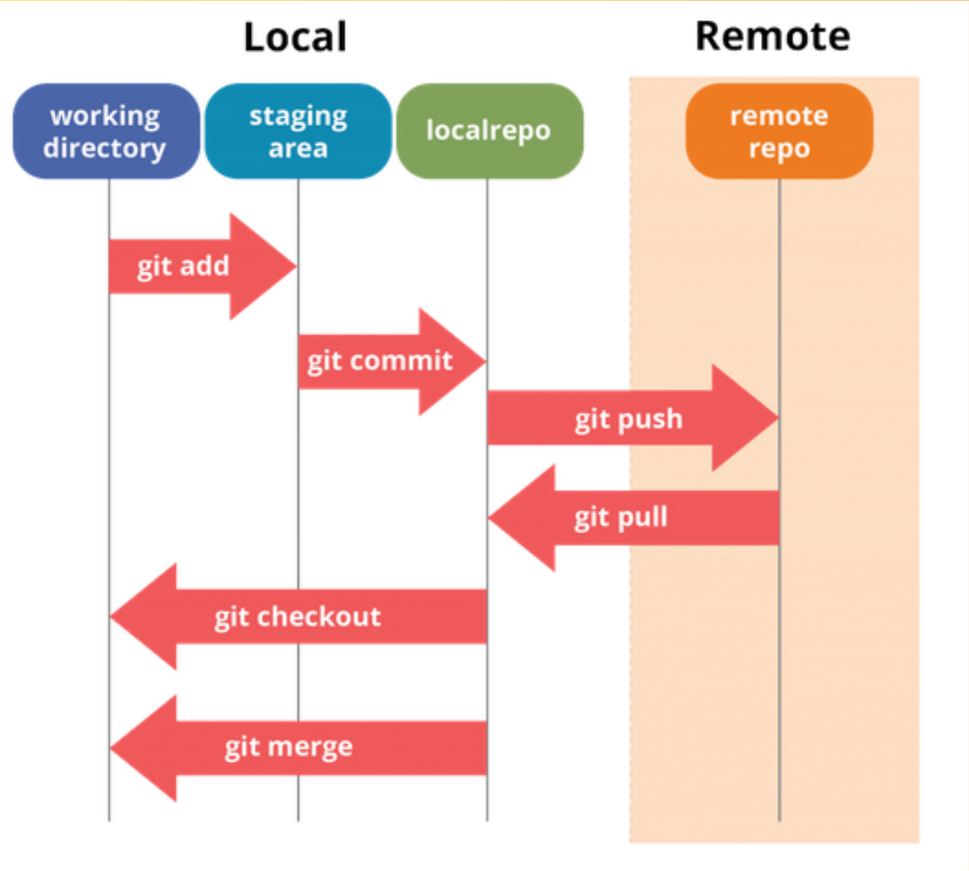
2. KOMUT SATIRI TABANLI GIT İSTEMCİLERİ

   GIT

   GIT BASH

   GIT FOR WİNDOWS (MSYS GIT)





BÖLÜM 3 : TEMEL GIT İŞLEMLERİ

3.1 Yeni Bir Depo Oluşturma

3.2 Değişiklikleri İzleme

3.3 Değişiklikleri Kaydetme

3.4 Dal Yönetimi

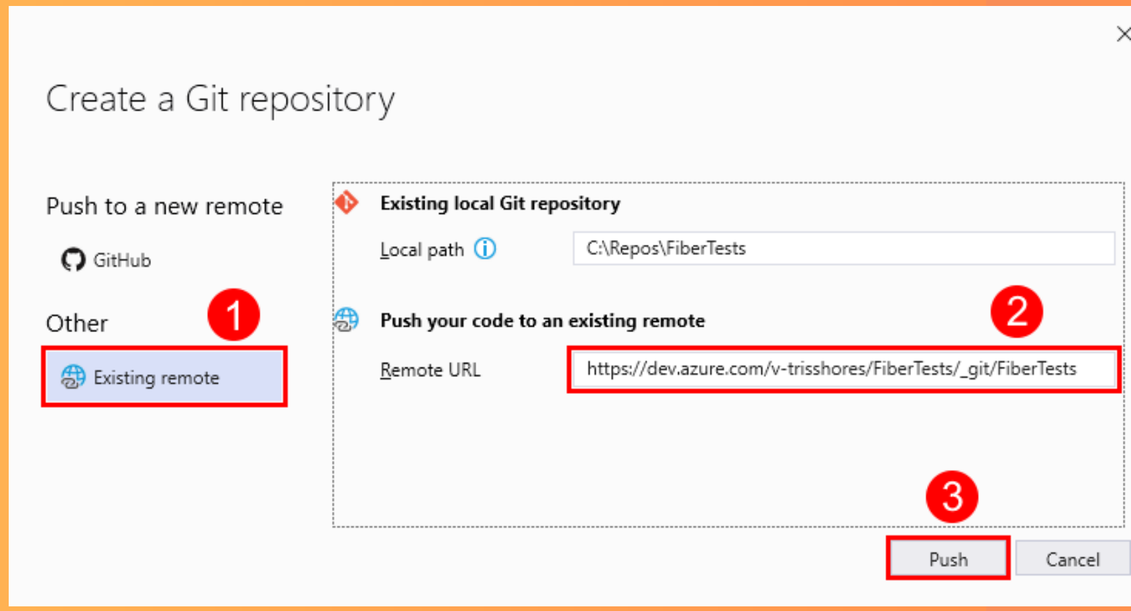
3.5 Uzak Depolarla Çalışma

3.1 Yeni Bir Depo Oluşturma

A. Boş Bir Depo Oluşturma

1-Boş bir depo oluşturmak için yerel bir klasör oluşturun. Bu klasör, projenizin çalışma dizini olarak kullanılacaktır.

2- Eğer hazır olarak oluşmuş bir klasörünüz yoksa "mkdir" komutunu kullanarak yeni bir klasör oluşturabilirsiniz.



3- İşletim sisteminize uygun bir terminal veya komut istemi açın.

4- Komut satırında, oluşturduğunuz klasöre gitmek için cd komutunu kullanın. Örneğin, cd /path/to/folder şeklinde bir komut girerek klasöre geçiş yapabilirsiniz.

5- 'git init' komutunu kullanarak Git'te depo oluşturun.

6- Artık boş bir Git deposuna sahipsiniz. Depo, .git adlı bir gizli klasör içerisinde depolanan Git verilerini içerir.

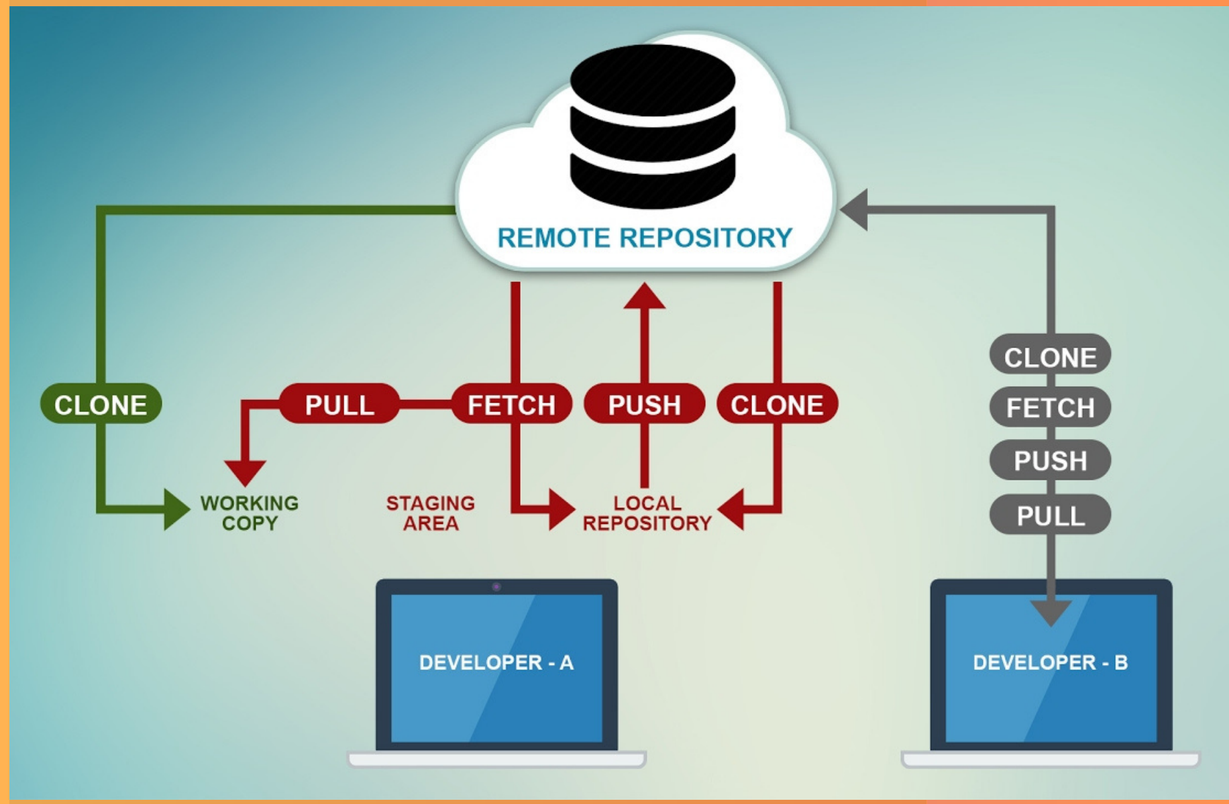
7- Depoyu başarıyla oluşturduktan sonra, dosyalarınızı bu depoya ekleyebilir, commit yapabilir ve Git'i kullanarak projenizi yönetebilirsiniz.

B. Bir Depoyu Klonlamak

1-Klonlamak istediğiniz Git deposunun URL'sini alın.

2- Klonlanacak depo için bir klasör oluşturun. Bu klasör, projenizin çalışma dizini olarak kullanılacaktır.

3- Terminalde **git clone <repo_url>** komutunu kullanarak repo_url kısmına kopyaladığınız url'yi yapıştırın ve depo klonlama işlemini tamamlayın.



3.2 DEĞİŞİKLİKLERİ İZLEME

DOSYA DURUMUNU KONTROL ETME (GIT STATUS)

Git deposundaki dosya durumunu kontrol etmek için kullanılan bir komuttur. Mevcut dosya dizinindeki dosyaların ve değişikliklerin durumunu gösterir.

DOSYA EKLEMEK (GIT ADD)

git add komutu, Git deposuna dosyaları eklemek için kullanılır. Bu komut çalışma dizinindeki değişiklikleri Git'e bildirir ve takip edilen dosyanın yeni sürümünü oluşturmak için hazırlar.

DOSYA ÇIKARMAK (GIT RM)

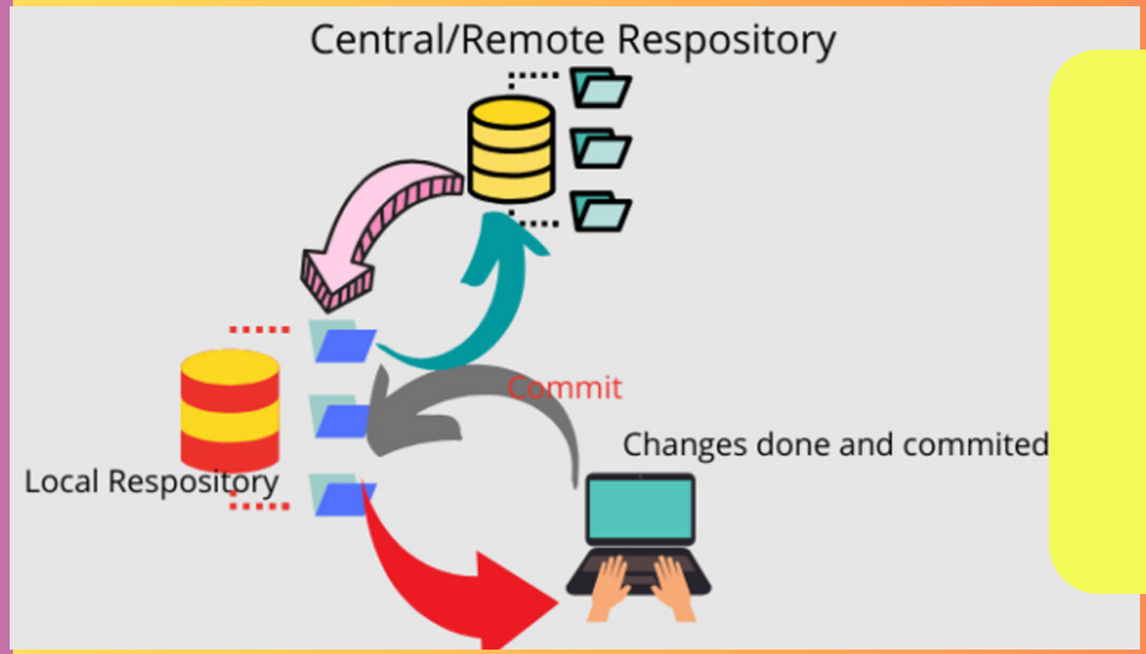
git rm' komutu, Git deposundan dosyaları veya dizinleri silmek için kullanılır. Bu komut aynı zamanda dosyanın izini Git deposundan kaldırır.

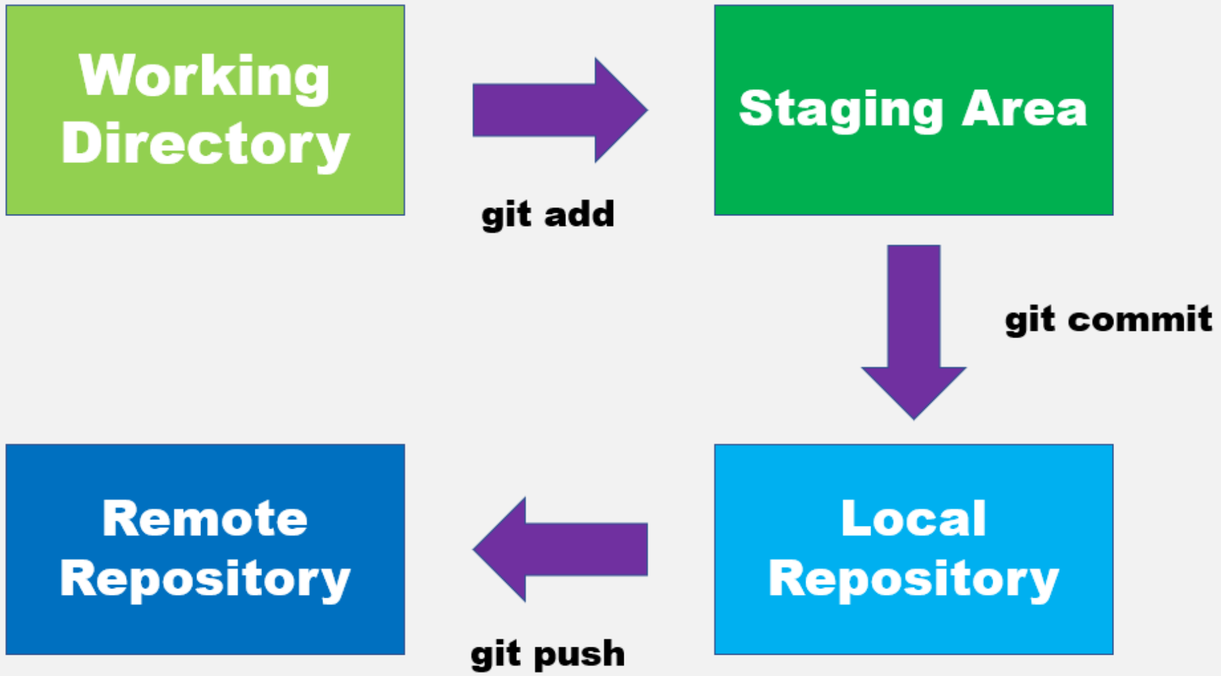
DEĞİŞİKLİKLERİ GERİ ALMAK (GIT CHECKHOUT)

'git chechkout' komutu Git'te değişiklikleri geri almak için kullanılır.Bu komut çalışma dizinindeki dosyaları veya bir dalı geri almak için kullanılır.

3.3 DEĞİŞİKLİKLERİ KAYDETME (COMMİT)

‘git commit’ komutu, Git deposunda yapılan değişiklikleri kaydetmek için kullanılır.Bu komut, dosyayı sürüm geçmişinde kalıcı olarak kaydeder veya anlık görüntüler.Her commit birbirinden farklı bir kimlik ile tanımlanır.Commit'ler, Git deposunda yapılan değişikliklerin geçmişini oluşturur ve takip eder. Bu commit'leri başkalarıyla paylaşmak veya depoda saklamak için **‘git push’** komutunu kullanabilirsiniz.





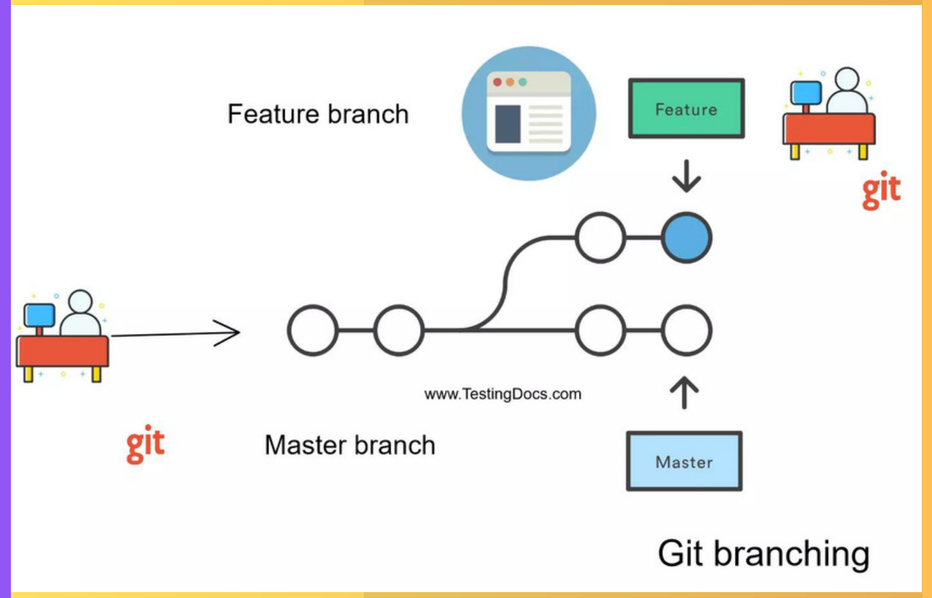
COMMIT GEÇMİŞİNİ GÖRÜNTÜLEME (GIT LOG)

git log komutu, Git deposundaki commit geçmişini görüntülemek için kullanılan bir komuttur.

git log komutunu kullanarak aşağıdaki bilgileri elde edebilirsiniz:

1.Commit Kimliği (Commit Hash)
2.Commit Tarihi ve Saati
3.Commit Yapan Kişi
4.Commit Mesajı

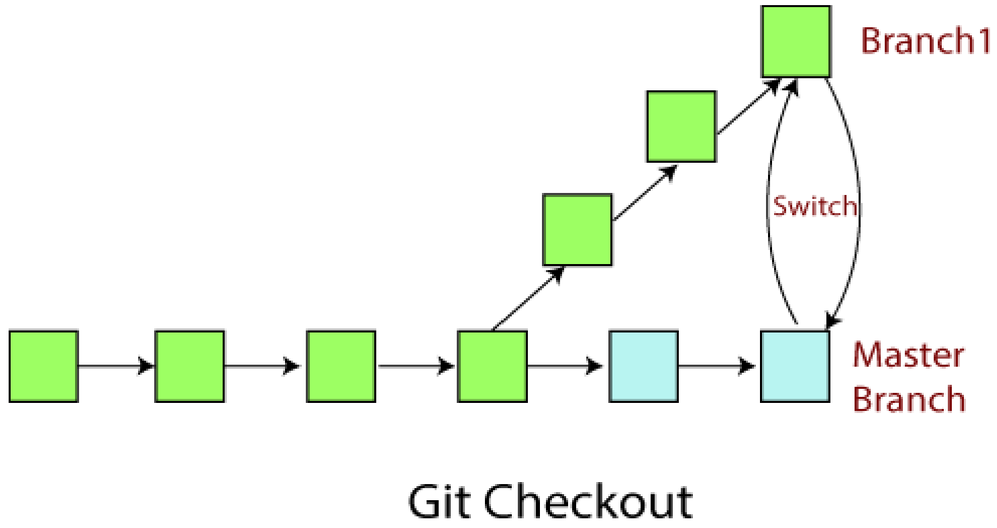
Ayrıca, **git log** komutuna çeşitli seçenekler ekleyerek çıktıyı filtreleyebilir ve daha spesifik bir şekilde commit geçmişini görüntüleyebilirsiniz.



3.4 DAL YÖNETİMİ

A.YENİ BİR DAL OLUŞTURMA (GIT BRANCH)

‘git branch’ komutu, Git deposundaki mevcut dalları (branches) listeleyen, yeni dallar oluşturan veya mevcut bir dalı silen bir komuttur.

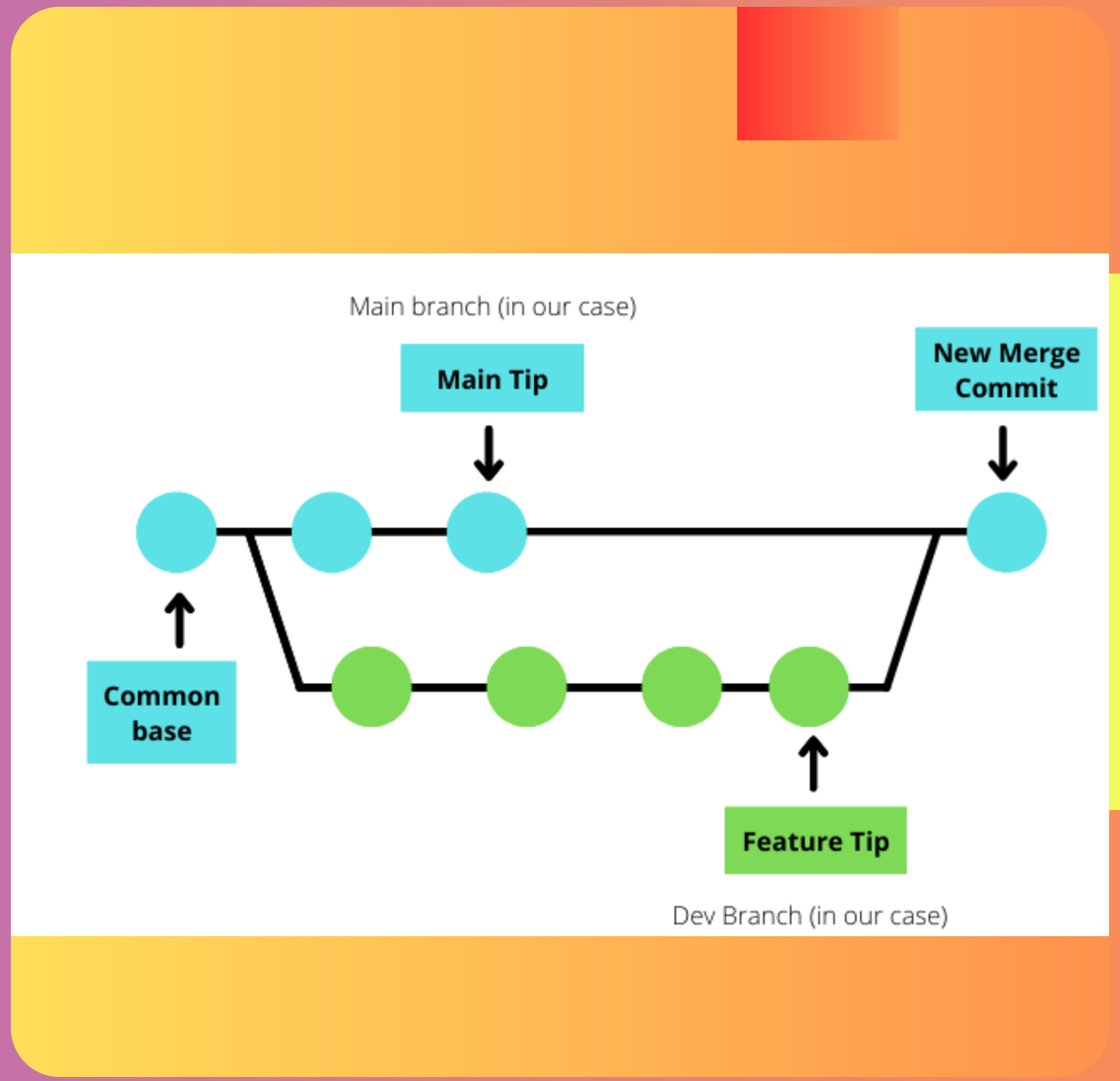


B. BİR DALA GEÇİŞ YAPMA (GIT CHECKOUT)

‘git checkout’ komutu, Git deposunda farklı bir dal, commit veya dosya sürümüne geçiş yapmak için kullanılır.

C. BİR DALI BİRLEŞTİRME (GIT MERGE)

‘git merge’ komutu, Git deposundaki farklı dalları birleştirmek için kullanılır. İki farklı dalın değişikliklerini tek bir dalda birleştirmek amacıyla kullanılır.



3.5 UZAK DEPOLARLA ÇALIŞMAK

GIT REMOTE
UZAK DEPO EKLEMEK

GIT PUSH
DEĞİŞİKLİKLERİ UZAK DEPOYA GÖNDERMEK

GIT CLONE
UZAKTAKİ BİR DEPOYU KLONLAMAK

GIT PULL
UZAKTAKİ DEĞİŞİKLİKLERİ ALMAK

1

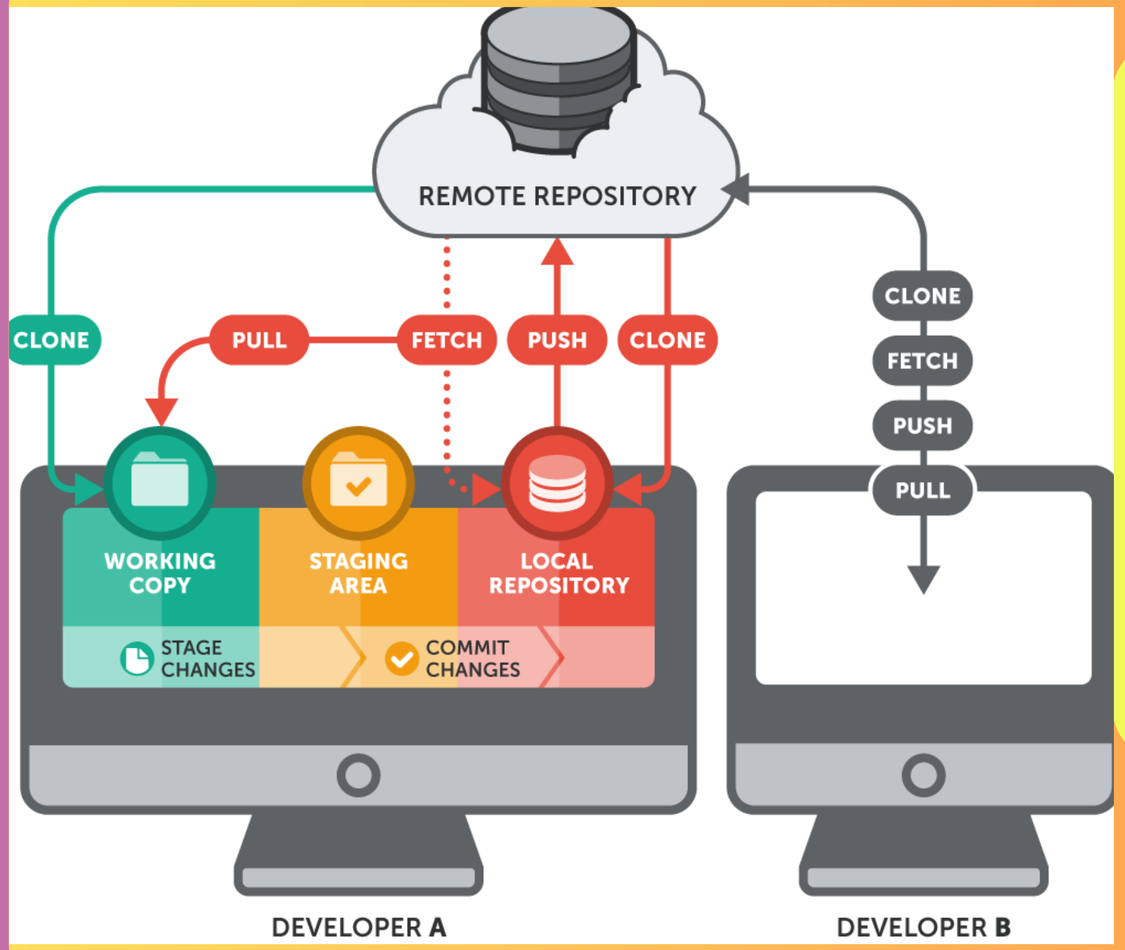
GIT REMOTE

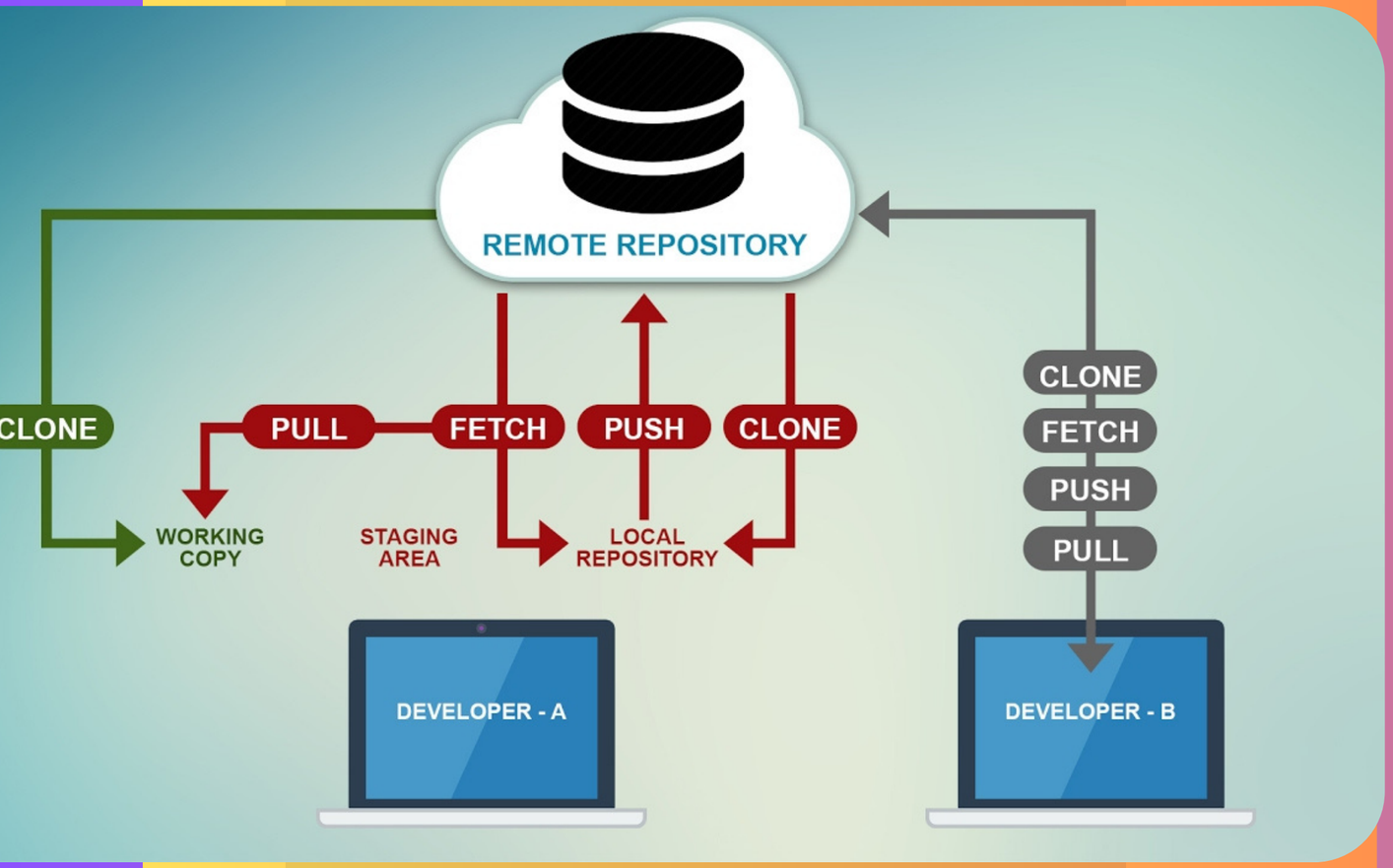
git remote' komutu, Git deposuyla uzak sunucu veya başka bir kopya arasındaki bağlantıları yönetmek için kullanılır. Uzak sunucu, depoya erişim sağlayan ve depodaki değişiklikleri paylaşan bir sunucudur.

2

GIT CLONE

`**git clone**` komutu, bir uzak Git deposunu yerel bir makineye kopyalamak için kullanılır. Bu komut, uzak depodaki tüm geçmişi, dalları ve dosyaları indirir ve yerel bir kopyasını oluşturur.





3

GIT PUSH

`git push` komutu, yerel Git deposundaki değişiklikleri bir uzak sunucuya (örneğin GitHub, GitLab veya Bitbucket gibi) göndermek için kullanılır. Bu komut, yerelde yapılan commit'leri uzak sunucudaki depoya yükler ve depoyu günceller.

4

GIT PULL

`git pull` komutu, uzak Git deposundan en güncel değişiklikleri almak ve yerel depoyu güncellemek için kullanılır. Bu komut, `git fetch` komutunu çalıştırarak uzak depodaki değişiklikleri yerel depoya getirir ve ardından bu değişiklikleri mevcut çalışma dalına birleştirir (merge).

KAYNAKÇA

https://www.btkakademi.gov.tr/portal/course/versiyon-kontrolleri-git-ve-github-19439

https://git-scm.com/

https://bitbucket.org/

https://odayibasi.medium.com/git-komutlar%C4%B1-ve-kullan%C4%B1m%C4%B1-51ec0c07a434

https://www.datree.io/resources/git-commands
https://www.atlassian.com/git/tutorials/comparing-workflows

GIT VERSİYON KONTROL SİSTEMİ

TEŞEKKÜRLER!

ÖDEV SAHİPLERİ

HELİN ÖZALKAN
MERVE KUL
BERKAY AYDIN