Git İş Akışının Temelleri

- **Cloning**: Takımın GitHub'da barındırdığı bir depoyu (repository) klonlamak, projenin kodunun ve sürüm geçmişinin yerel bilgisayarınıza kopyasını oluşturur.
- **Branching**: Ana dal (main branch) üzerinden yeni bir dal (branch) oluşturmak, ana kod tabanını etkilemeden değişiklik yapmanıza olanak tanır.

Değişikliklerin Yönetimi

- **Staging Area**: Değişiklikleri kaydetmeden önce dosyaları geçici olarak topladığınız alan.
- **Committing**: Dosyaları dalınıza kaydetmek, değişikliklerinizi kaydeder ve yeni özelliğinizi dalın bir parçası haline getirir.

Kodun Paylaşımı ve İnceleme

- **Pull Request**: Dalınızdaki değişiklikleri ana dal ile birleştirmek için yöneticiden (maintainer) inceleme ve onay talep etme süreci.
- **Merging**: Yöneticinin pull request'i onaylamasıyla dalınızdaki değişikliklerin ana dalda görünür hale gelmesi.

Temel Git Komutları

- **git init**: Yeni bir Git deposu (repository) oluşturur ve gerekli dosya yapısını ayarlar.
- **git add**: Çalışma dizinindeki (working directory) değişiklikleri geçici alana (staging area) taşır.
    - **git add <filename.txt>** (belirli bir dosyayı eklemek için)
    - **git add .** (geçerli dizindeki yeni veya değiştirilmiş tüm dosyaları eklemek için)
    - **git add -A** (dizinin yapısında nerede olursanız olun, depo kökünden itibaren tüm değişiklikleri eklemek için)

Değişikliklerin Kaydedilmesi

- **git commit**: Geçici alandaki değişiklikleri kaydeder ve bir açıklama mesajı ile birlikte projeye ekler.
- **git log**: Projeye yapılan önceki değişiklikleri gözden geçirmenizi sağlar.

Dal Yönetimi

- **git branch**: Yeni bir dal (branch) oluşturur veya mevcut dalları listeler.
    - **git branch** (dalları listelemek için)
    - **git branch <new-branch>** (yeni bir dal oluşturmak için)
    - **git branch -d <branch-name>** (bir dalı silmek için)
- **git checkout**: Mevcut dallar arasında geçiş yapmanızı sağlar.

Birleştirme İşlemleri

- **git merge**: Özellik dalını (feature branch) ana dal ile birleştirir.
- **git status**: Çalışma dizinindeki dosyaların durumunu gösterir.
- **git clone**

    - Bir depoyu uzaktan kaynaklardan yerel makinenize kopyalar. Bu, geçerli çalışma dizininizde bir depo kopyası oluşturur.
    - *Sözdizimi*: **git clone <repository URL>**

Git branching workflow (dal akışı), yazılım geliştirme projelerinde sürüm kontrolü ve işbirliği için kritik öneme sahiptir.

Branching Workflow

- Branches (dallar), özellikler, deneyler veya düzeltmeler üzerinde bağımsız çalışmayı sağlar.
- Git clone komutu ile bir repository (depo) kopyalanarak yerel makinede bir kopya oluşturulur.

Creating and Managing Branches

- Yeni bir branch oluşturmak için git branch [branch_name] komutu kullanılır.
- Branch'e geçmek için git checkout [branch_name] komutu ile geçiş yapılır.

Committing and Merging Changes

- Değişiklikleri kaydetmek için git add [file_name] ve git commit -m "[message]" komutları kullanılır.
- Branch'i ana branch ile birleştirmek için önce ana branch'e geçilir (git checkout main) ve ardından git merge [branch_name] komutu ile birleştirilir.

Cloning Repositories

- **Cloning**: Bir repository'yi (depo) yerel makinenize kopyalamak için kullanılır. Bu işlem, değişikliklerinizi senkronize etmenizi sağlar.
- **git clone** komutu ile repository URL'si kullanılarak yerel bir kopya oluşturulur.

Forking Projects

- **Forking**: Mevcut bir projeyi alıp yeni bir proje için temel oluşturmak amacıyla kullanılır. Bu, orijinal projeyi etkilemeden değişiklik yapmanıza olanak tanır.
- Fork oluşturmak için GitHub'da "Fork" butonuna tıklanır.

Syncing Changes

- Değişiklikleri GitHub'a geri göndermek için **git add**, **git commit**, ve **git push** komutları kullanılır.
- **git fetch** ve **git pull** komutları, uzak repository'den (remote repository) değişiklikleri yerel repository'ye almak için kullanılır. **git pull**, değişiklikleri alıp birleştirir.

Collaboration and Pull Requests

- **Pull Request**: Değişikliklerinizi orijinal projeye göndermek için kullanılır. Proje sahibi, değişikliklerinizi gözden geçirip kabul edebilir.
- **Upstream** ve **origin** terimleri, uzak repository'ler arasında ilişkiyi tanımlar; **origin** genellikle fork'unuzu, **upstream** ise orijinal projeyi ifade eder.

Cloning Repositories

- **Cloning**: Bir repository'yi (depo) yerel makinenize kopyalamak için kullanılır. Bu, projeyi yerel ortamda çalıştırmanıza olanak tanır.
- **git clone** komutu ile repository URL'si kullanılarak yerel bir kopya oluşturulur.

Forking Projects

- **Forking**: Mevcut bir projeyi alıp yeni bir proje oluşturmak için kullanılır. Bu, orijinal projeyi etkilemeden yeni özellikler eklemenizi sağlar.
- Fork oluşturmak için GitHub'da "Fork" butonuna tıklanır; bu işlem, orijinal projeden bağımsız bir kopya oluşturur.

Syncing Changes

- Değişiklikleri GitHub'a göndermek için **git add**, **git commit**, ve **git push** komutları kullanılır.
- **git fetch** ve **git pull** komutları, uzak repository'den (remote repository) değişiklikleri almak için kullanılır. **git pull**, değişiklikleri alıp mevcut branch ile birleştirir.

Collaboration and Pull Requests

- **Pull Request**: Değişikliklerinizi orijinal projeye göndermek için kullanılır. Proje yöneticisi, değişikliklerinizi gözden geçirip kabul edebilir.
- **Upstream** ve **origin** terimleri, uzak repository'ler arasındaki ilişkiyi tanımlar; **origin** genellikle fork'unuzu, **upstream** ise orijinal projeyi ifade eder.

Proje Yönetiminde Rol Dağılımı

- **Developer (Geliştirici)**: Grup projesinde yer alan bir katılımcıdır ve diğerleriyle iletişim kurmak için belirli Git komutlarını kullanmalıdır.
- **Integrator (Entegratör)**: Diğerlerinin yaptığı değişiklikleri gözden geçirir, entegre eder ve sonuçları yayınlar.

Git Komutları

- **Git clone**: Uzak bir paylaşılan depodan (repository) kopya oluşturur.
- **Git pull** ve **git fetch**: Depoyu güncel tutmak için kullanılır.
- **Git push**: Yerel depodaki değişiklikleri paylaşılan depoya gönderir.

Repository Administrator (Depo Yöneticisi)

- Depo yapısını düzenler ve kullanıcıların etkileşimlerini yönetir.
- **GitHub Actions** kullanarak yazılım iş akışlarını otomatikleştirir.

GitHub Temel Bilgileri

- GitHub, 100 milyondan fazla repository (depo) barındırır.

- Bir repository'yi (depo) klonlayabilir (clone) ve değişiklikleri orijinaline senkronize edebilirsiniz.

Git İş Akışı

- Uzak repository'yi klonlayın (clone) veya yeni bir Git repository'si başlatın (initialize).
- Dosyaları staging area (hazırlık alanı) taşıyın.
- İlk commit (taahhüt) işlemini gerçekleştirin.
- Bir branch (dal) oluşturun ve üzerinde çalışın.
- Dosyaları staging area'ya ekleyin ve commit yapın.
- Yerel commit'leri uzak repository'ye push (gönderin).
- İnceleme ve birleştirme için pull request (çekme isteği) oluşturun.
- Yerel repository'yi güncellemek için pull (çekme) işlemini kullanın.

Roller ve Komutlar

- Developer (geliştirici): git clone, git pull, git push gibi komutları kullanır.
- Integrator (entegratör): Diğerlerinin yaptığı değişiklikleri gözden geçirir ve entegre eder.
- Repository Administrator (depo yöneticisi): Repository'nin organizasyonunu yapılandırır ve kullanıcı etkileşimlerini yönetir.

Cloning	Projenin kodunun ve tam sürüm geçmişinin uzaktan depodan yerel makinede bir kopyasını oluşturma süreci.
Commit	Projenin belirli bir andaki mevcut durumunun bir anlık görüntüsü ve yapılan değişikliklerin açıklamasıyla birlikte.
Developer	Kod yazmaktan sorumlu bilgisayar programcısı.
Distributed version control system (DVCS)	Kod değişikliklerini, nerede depolandığına bakılmaksızın takip eden bir sistem. Birden fazla kullanıcı, gerektiğinde bilgisayarlarında kod tabanını yansıtarak aynı kod tabanı veya depo üzerinde çalışır; dağıtılmış sürüm kontrol yazılımı, çeşitli kod tabanı yansımaları arasındaki senkronizasyonu yönetmeye yardımcı olur.
Fork	GitHub hesabınıza bir deponun kopyası.

Forking	Forking, orijinal depoyu etkilemeden üzerinde çalışabileceğiniz bir deponun kopyasını oluşturur.
GitHub	Git deposu için web tabanlı bir hizmet.
Git	GNU Genel Kamu Lisansı altında dağıtılan ücretsiz ve açık kaynaklı bir yazılım. Kullanıcılara, dünyanın herhangi bir yerinde bilgisayarlarında kendi projelerinin bir kopyasına sahip olma imkanı veren dağıtılmış bir sürüm kontrol sistemidir.
Integrator	Geliştiricilerin yaptığı değişiklikleri yönetmekten sorumlu bir rol.
Main branch	Kodunuzun dağıtılabilir sürümünü depolayan bir dal. Ana dal varsayılan olarak oluşturulur ve kesin bir dal olarak kabul edilir.
Merge	Genellikle bir özellik dalını ana dala birleştirmek için bir dalın değişikliklerini diğerine birleştirme süreci.
Origin	Kopyanın klonlandığı depoyu ifade eden bir terim.
Pull request	Değişikliklerinizin nihai hale gelmeden önce birinin gözden geçirmesini ve onaylamasını talep etmek için kullanılan bir süreç.
Remote repositories	Başka yerlerde depolanan depolar. İnternette, ağınızda veya yerel bilgisayarınızda var olabilirler.
Repository administrator	Depoya erişimi yapılandırma ve sürdürme sorumluluğuna sahip bir rol.
Repository	Uygulama kaynak kodu dahil belgeleri depolamak için kullanılan bir veri yapısı. Sürüm kontrolü için ayarlanmış proje klasörlerini içerir.
Staging area	Commitlerin biçimlendirilebileceği ve tamamlanmadan önce gözden geçirilebileceği bir alan.
Upstream	Geliştiricilerin yerel kopyanın klonlandığı orijinal kaynağı ifade etmek için kullandığı bir terim.
Version control	Belgelerinizdeki değişiklikleri takip etmenizi sağlayan bir sistem. Bu süreç, herhangi bir hata yapıldığında belgelerin eski sürümlerini geri almanıza olanak tanır.