Pull Requests

Pull Request nedir, gerçekçi bir feature branch workflow'u üzerinden nasıl oluşturulur /incelenir/onaylanır, merge vs squash-and-merge farkı, merge sonrası branch'leri silmek, ve branch protection rules.

Orta 30 dk

Son güncelleme: 27 Ağustos 2026
Çevrimiçi oku: https://www.learnforgex.com/tr/topics/pull-requests

---

Bir branch'i GitHub'a push etmek, kodunu paylaşılan sunucuya götürür — onu `main`'e sokmaz. Bu boşluk kasıtlıdır: profesyonel takımlar neredeyse hiçbir zaman başka biri değişikliği incelemeden doğrudan `main`'e merge etmez. Pull Request, GitHub'ın bu inceleme-sonra-merge süreci için mekanizmasıdır, ve gerçek Java/Spring Boot takımlarının günlük olarak işbirliği yapmasının en yaygın tek yoludur.

Pull Request Nedir?

Bir **Pull Request** (PR), bir branch'in başka birine (tipik olarak feature branch'inin `main`'e) merge edilmesini öneren ve o değişiklik kümesinin gerçekten merge edilmeden önce tartışılması, incelenmesi ve nihayetinde onaylanması için özel bir sayfa açan bir GitHub özelliğidir. İsmine rağmen, kendi başına hiçbir şeyi "pull" etmez — birinin branch'ini incelemesi ve merge etmesi için bir istektir.

Feature Branch Workflow

Pull Request'ler, "Branches & Merging" dersindeki feature-branch workflow'unun doğal devamıdır — değişen tek şey branch'in `main`'e *nasıl* merge edildiğidir:

```
PR olmadan:  branch → git merge (yerel olarak, inceleme yok)
PR ile:      branch → GitHub'a push → PR aç → incele → merge et (GitHub'da)
```

Branch, commit ve push adımları, zaten bildiğinle tamamen aynıdır — bir PR, yalnızca yerel `git merge` adımını, incelenmiş, GitHub'da barındırılan bir merge ile değiştirir.

Pull Request Oluşturmak

Feature branch'in GitHub'a push edildikten sonra:

```
git push -u origin add-password-reset
```

GitHub genellikle az önce push ettiğin branch için bir PR açmayı öneren bir banner gösterir. Bir tane oluşturmak şunları gerektirir: kaynak branch (seninki), hedef branch (genellikle `main`), bir başlık, ve değişikliğin *ne* yaptığını ve *neden* yaptığını açıklayan bir açıklama — PR açıklaması, reviewer'ların ilk izlenimlerini oluşturduğu yerdir, bu yüzden ona iyi bir commit mesajıyla aynı özeni göster.

Bir Pull Request'i İncelemek

İncelemek, önerilen diff'i okumak (GitHub bunu dosya dosya gösterir, tam olarak `git diff` gibi) ve geri bildirim bırakmak demektir. Bir reviewer, belirli satırlara satır-içi yorumlar, PR'ın tamamına genel yorumlar bırakabilir, ve nihayetinde üç inceleme sonucundan birini seçebilir: yalnızca yorum, değişiklik iste, ya da onayla.

Değişiklik İstemek

Reviewer gerçek bir sorun bulursa — bir bug, eksik bir test, katılmadığı bir yaklaşım — incelemesini **"Request changes"** olarak gönderir. Bu, birçok projede, çözülene kadar PR'ın merge edilmesini gerçekten engelleyen resmi bir sinyaldir (rastgele bir yorumdan farklı olarak). Sonra yazar aynı branch'e yeni commit'ler push eder; GitHub PR'ı bunlarla otomatik olarak günceller, yeni bir PR gerekmez.

Bir Pull Request'i Onaylamak

Reviewer memnun kaldığında, incelemesini **"Approve"** olarak gönderir. Çoğu profesyonel takımda, merge butonu kullanılabilir hale gelmeden önce en az bir onay gerekir (aşağıda ele alınan branch protection rule'ları tarafından zorunlu kılınır) — bu, ikinci bir çift gözün değişikliğin `main`'e ulaşmasını gerçekten onayladığı andır.

Bir Pull Request'i Merge Etmek

Onaylandıktan sonra, GitHub'ın merge butonu merge'i sunucuda gerçekleştirir, tam olarak yerel `git merge`'ün yapacağı gibi — fark, *ne zaman* gerçekleştiği (incelemeden sonra, önce değil) ve *nerede* gerçekleştiğidir (GitHub'da, tüm takım tarafından görülebilir, kimin inceleyip onayladığına dair kalıcı bir kayıtla).

Merge vs Squash and Merge

GitHub aslında o buton üzerinde bir merge stratejisi seçimi sunar:

- **Create a merge commit** — tam olarak zaten bildiğin `git merge` gibi davranır: branch'ten her bireysel commit'i, artı onları bir araya bağlayan bir merge commit'i korur.

- **Squash and merge** — branch'teki *her* commit'i `main`'de tek, yeni bir commit'te birleştirir. Bu, özellikle dağınık geçmişi olan feature branch'leri için faydalıdır ("fix typo", "fix typo again", "actually fix it") — branch'in `main`'e bakan geçmişi tek, temiz bir commit haline gelir, bireysel work-in-progress commit'leri yol boyunca var olmuş olsa bile. Squash etmeyi (bunun elle yapılan `git rebase -i` versiyonu dahil) bu kursta daha sonra kendi özel dersinde ele alıyoruz.

Merge Sonrası Branch'leri Silmek

Bir PR merge edildikten sonra, branch'i amacına hizmet etmiştir — GitHub, merge edilmiş PR sayfasında tam olarak tek tıkla "Delete branch" butonu sunar. Bu, yerel branch'lerden zaten bildiğin `git branch -d` temizliğinin remote karşılığıdır, ve o kadar güvenlidir: commit'ler `main`'in geçmişinde kalıcı olarak yaşamaya devam eder.

Branch Protection Rules

Branch protection rule'ları, bir repository admin'inin bu pratikleri, herkesin uymayı hatırlamasına güvenmek yerine otomatik olarak zorunlu kılmak için GitHub'da yapılandırdığı ayarlardır — örneğin: merge etmeden önce en az bir onay gerektirmek, önce branch'in `main` ile güncel olmasını gerektirmek, ya da `main`'e doğrudan push'ları tamamen engellemek (*herkesi*, adminler dahil, bir PR'dan geçmeye zorlamak). Profesyonel takımlarda, `main` neredeyse her zaman bu şekilde korunur.

Yaygın Hatalar

- **Belirsiz bir başlıkla ve açıklama olmadan bir PR açmak**, reviewer'ı değişikliğin *neden* var olduğunu yalnızca diff'ten tersine mühendislikle çıkarmaya bırakmak.
- **Bunu bekleyen bir takımda, incelemeyi beklemeden kendi PR'ını merge etmek** — branch protection rule'ları tam olarak bunun yanlışlıkla mümkün olmasını önlemek için vardır.
- **İnceleme ortasında reviewer'ı uyarmadan bir branch'in üzerine force-push yapmak** — bu, tam olarak reviewer'ın baktığı commit'leri yeniden yazar (bunun neden riskli olduğunu Force Push dersinde ayrıntılı olarak ele alıyoruz).

En İyi Pratikler

- PR'ları küçük ve tek bir şeye odaklı tut — daha küçük PR'lar daha hızlı ve daha kapsamlı incelenir.
- *Neden*i açıklayan bir açıklama yaz, iyi bir commit mesajıyla aynı disiplin.
- İnceleme yorumlarına, reviewer'ın zaten gördüğü şeyi sessizce düzenleyip force-push yapmak yerine yeni commit'ler push ederek yanıt ver.

Özet, Cheat Sheet ve Terimler Sözlüğü

Özet

- Bir Pull Request, bir branch'in başka birine merge edilmesini önerir ve merge gerçekleşmeden önce inceleme için özel bir alan sağlar.
- İncelemeler yorum / değişiklik iste / onayla ile sonuçlanır — merge etmeden önce genellikle onay gerekir.
- "Create a merge commit" her commit'i korur; "Squash and merge" onları `main`'de tek, temiz bir commit'te birleştirir.
- Branch protection rule'ları, yalnızca disipline güvenmek yerine inceleme gereksinimlerini otomatik olarak zorunlu kılar.

Cheat Sheet

```
git push -u origin <branch>      # branch''ini push et, sonra GitHub''da bir PR aç
# ... inceleme GitHub'da gerçekleşir: yorum / değişiklik iste / onayla ...
# ... GitHub'da merge et: "Create a merge commit" ya da "Squash and merge" ...
git branch -d <branch>           # merge''den sonra yerel kopyanı temizle
```

Terimler Sözlüğü

- **Pull Request (PR)** — bir branch'in başka birine merge edilmesini öneren, özel bir inceleme alanına sahip GitHub önerisi.
- **Request changes** — (korunan branch'lerde) çözülene kadar merge etmeyi engelleyen resmi bir inceleme sonucu.
- **Branch protection rule** — merge'e izin vermeden önce gerekli onaylar gibi gereksinimleri zorunlu kılan bir repository ayarı.