Race Condition – Aynı Rozetin Paralel İstekler ile Bir Kullanıcıya Birden Fazla Kez Atanabilmesi

---

Özet

Topluyo platformunda rozet atama mekanizmasında eşzamanlılık kontrolü bulunmamaktadır. Aynı rozet için gönderilen paralel istekler sonucunda sistem, kullanıcının ilgili rozeti daha önce alıp almadığını güvenilir şekilde doğrulayamamakta ve aynı rozet bir kullanıcıya birden fazla kez atanabilmektedir.

Bu durum iş mantığı bütünlüğünü bozan bir race condition zafiyetidir.

Zafiyet Türü

- Race Condition
- Business Logic Flaw
- Concurrency Control Eksikliği

Teknik Açıklama

Rozet atama süreci “kontrol et → sonra ekle” mantığıyla çalışmaktadır. Ancak bu iki adım atomik değildir.

Paralel istek senaryosunda:

- İstek A → “rozet yok” kontrolünü geçer
- İstek B → aynı anda “rozet yok” kontrolünü geçer
- Her iki istek de ayrı ayrı insert işlemi yapar

Bu durum sonucunda aynı kullanıcıya aynı rozet birden fazla kez atanır.

Tekrar Adımları (PoC)

1. Geçerli bir kullanıcı hesabı ile giriş yapılır
2. Rozet atama isteği yakalanır
3. Aynı HTTP isteği çoğaltılır
4. İstekler eşzamanlı olarak sunucuya gönderilir
5. Sistem concurrency kontrolü yapamaz
6. Aynı rozet kullanıcıya birden fazla kez atanır

Beklenen Sonuç

- Aynı rozet aynı kullanıcıya yalnızca bir kez atanmalıdır
- Duplicate kayıt oluşmamalıdır

Gerçek Sonuç

- Paralel istekler sonucunda aynı rozet aynı kullanıcıya birden fazla kez atanabilmektedir
- Veritabanında duplicate kayıt oluşmaktadır

Etki

İş Mantığı Etkisi

- Gamification sistemi manipüle edilebilir hale gelir
- Kullanıcılar aynı rozeti çoğaltabilir

Veri Bütünlüğü Etkisi

- Duplicate veri oluşur
- Raporlama ve istatistikler bozulur

Sistemsel Etki

- Backend iş mantığı güvenilirliği zayıflar
- İleride daha kritik yetkilendirme sistemlerine zemin oluşturur

---

Kök Neden

- İşlem adımlarının atomik olmaması
- Concurrency kontrolünün bulunmaması
- Veritabanı seviyesinde unique constraint eksikliği
- Idempotent API tasarımının uygulanmaması

Çözüm Önerileri

1. Veritabanı Seviyesi Koruma

- `user_id + badge_id` kombinasyonu için **UNIQUE constraint** eklenmelidir
- Duplicate kayıtlar kesin olarak engellenir

---

2. Atomik Transaction Kullanımı

- Kontrol ve insert işlemleri tek transaction içinde yapılmalıdır

---

3. Row-Level Locking

- `SELECT ... FOR UPDATE` ile ilgili kayıt kilitlenmelidir
- Paralel işlem yarış durumu engellenir

---

4. Idempotency Mekanizması

- Aynı request tekrar geldiğinde aynı sonuç dönülmelidir

---

5. İş Mantığı Güçlendirme

- “check then insert” yerine “insert + constraint rely” yaklaşımı kullanılmalıdır

---

Sonuç

Bu zafiyet sistemde veri bütünlüğünü bozan kritik bir iş mantığı problemidir. Eşzamanlılık kontrolü ve veritabanı seviyesinde constraint uygulanması ile tamamen önlenebilir.