

Stok, Cari ve Fiş Hizmeti Sağlayan Web API

Aşağıda belirtilen gereksinimlere uygun bir Web API ve UI geliştirmenizi istiyoruz. Ödevi tamamlandıktan sonra, çözümünüzü bir GitHub reposunda paylaşmanızı ve bize linki iletmek üzere e-posta yoluyla geri göndermenizi rica ediyoruz. . Ödevle ilgili sorularınız ve ödev teslimi için mülakat sorumlusu ile iletişime geçebilirsiniz.

Beklenenler:

- Admin, user yetkilendirmesi olmalıdır.
- Stok, Cari ve Fiş (Fatura) hizmetlerini sağlayan bir web API geliştirmeniz gerekmektedir.
- Tüm büyüklükler için CRUD (Create, Read, Update, Delete) işlemleri desteklenmelidir.
- Listelerde sayfalama ve filtreleme özellikleri eklenmelidir. Stok ve cariler için cache kullanılmalı ve CRUD işlemlerine göre otomatik olarak yenilenmelidir.
- Kullanıcı dostu bir ön yüz geliştirmesi bekleniyor.
- UI projesinde, HTTP isteklerinin cache' lenerek tutulması.
- Optimize bir UI/UX bekleniyor.

Büyüklükler:

- Admin, temel büyüklükleri kontrol eder.
- Şirket: Varlıkların ilişkilendirildiği ana büyüklük olacaktır. Şirket bir kullanıcı olabilir.
- Stok: Fiş detaylarında kullanılmak üzere oluşturulmalıdır.
- Cari: Fişin kim için oluşturulduğunu ifade etmelidir. Cari bir kullanıcı olabilir.
- Fiş: Fiş, başlık ve detaylardan oluşmalıdır. Başlık bilgisi olarak cari, tarih ve seri bilgilerini içermelidir.
- Detaylar için stok seçimi yapılmalıdır. İlk oluşturulduğunda taslak durumunda olmalıdır.
- Onaylama mekanizması ile cari ve stok hareketleri oluşturulmalıdır.

Örnek Senaryo: Ahmet' e kesilmiş bir fiş bulunmaktadır ve 2 kg Elma satılmıştır. StockTrans tablosuna, Elma ile ilgili 2 kg'lık bir tüketim işlemi eklenmelidir. ActTrans tablosuna, Ahmet carisi için fiş tutarı kadar borç kaydı eklenmelidir. ActTrans ve StockTrans tablolarının cari ve stok gruplamaları kullanılarak elde edilen toplamlar, Balance olarak tutulmalıdır. Balance değerleri, stok ve cari kayıtlarında saklanabilir.

Araçlar:

- NET 8.0 kullanılmalıdır.
- MSSQL
- Cache işlemleri için Redis kullanılabilir.
- Serilog
- Prime Vue, PrimeFlex

Gereksinimlere uygun bir çözüm geliştirmek için aşağıdaki adımları takip edebilirsiniz:

1. Projenizin temelini oluşturun ve .NET ile bir Web API projesi başlatın.
2. Şirket, Stok ve Cari gibi varlık sınıflarını tanımlayın ve bunları veritabanında saklamak için gerekli ORM (Object-Relational Mapping) aracını kullanın.
3. CRUD işlemlerini gerçekleştirecek uygun API ve iş mantığını yazın.
4. Sayfalama ve filtreleme özelliklerini destekleyen API oluşturun.
5. Redis'i kullanarak cache mekanizmasını projenize entegre edin ve stok ve cari kayıtlarını CRUD işlemleriyle otomatik olarak güncelleyin.
6. Fiş oluşturma, taslak durumu, onaylama mekanizması ve cari/stok hareketlerini oluşturma işlemleri yönetecek API oluşturun.
7. Gereksinimleri tam olarak karşıladığından emin olun ve çözümünüzü postman ile test edin.
8. Vue.js Projesi oluşturun.
9. Büyüklükleri backend servisten alarak geliştirme yapın.
10. **(Opsiyonel)** Yapılan UI uygulamasının, Nuxt.js e geçirilmesi
11. **(Opsiyonel)** Uygulamaların Docker teknolojisi ile konteynır haline getirilmesi.

Notlar:

- Redis, cache işlemleri için performans avantajı sağlayan bir araçtır ve projenizin verimliliğini artırabilir.
- Uygulamayı test edebilmek için postman collectionı veya swagger olması gerekmektedir.

Bu ödev, .NET Core ve backend ve frontend geliştirme becerilerinizi değerlendirmek için tasarlanmıştır. Çözümünüzü en iyi şekilde sunmanızı ve gereksinimleri mümkün olan en iyi şekilde karşılayan, düşük hata oranına sahip ve iyi tasarlanmış bir projeyi temsil etmenizi bekliyoruz.