Patika.dev & Papara Kadın Yazılımcı Bootcamp Bitirme Projesi

Bir şirket özelinde sahada çalışan personeli için masraf kalemlerinin takibi ve yönetimi için bir uygulama talep edilmektedir. Bu uygulama ile saha da çalışan personel masraflarını anında sisteme girebilecek ve işveren bunu aynı zamanda hem takip edip edebilecek hemde vakit kaybetmeden harcamayı onaylayıp personele ödemesini yapabilecektir. Çalışan hem evrak fiş vb toplamaktan kurtulmuş olacak hemde uzun süre sahada olduğu durumda gecikmeden ödemesini alabilecektir. Uygulama şirket üzerinde yönetici ve saha personeli olmak üzere 2 farklı rolde hizmet verecektir. Çalışan saha personeli sadece sisteme masraf girişi yapacak ve geri ödeme talep edecektir. Personel mevcut taleplerini görecek ve taleplerinin durumunu takip edebilecektir. Onayda bekleyen taleplerini görebilir ve bunları takip edebilir. Sistem yöneticisi konumunda olan şirket kullanıcıları ise mevcut talepleri görecek ve onları onaylayıp red edebilecektir. Onayladıkları ödemeler için anında ödeme işlemi banka entegrasyonu ile gerçekleştirilecek olup çalışan hesabına EFT ile ilgili tutar yatırılacaktır. Red olan talepler için bir açıklama alanı girişi sağlanmalı ve talep sahibi masraf talebinin neden red olduğunu görebilmeli.

1. Kullanıcı işlemleri

- Sistem üzerinde 2 farklı rolde kullanıcı olabilir. Admin ve Personel.
- Personel sadece kendisi için masraf girişi yapabilir.
- Personel sadece kendi masraf tanımlarını görebilir.
- Personel taleplerini ilgili kriterlere göre filitreleyebilir .
- Personel red olan talepleri için neden red olduklarına dair açıklama görebilir. (Şahsi veya keyfi harcama vs.)
- Onaylanan talepleri için ödeme banka hesabına anında geçecek bir hayali ödeme sistemi tasarlanabilir. (Bankadan ödeme talimatı gönderilmesi için bir simülasyon uygulaması konumlandırılabilir.)
- Şirket sahibi yani admin kullanıcılar tüm personelin taleplerini görebilir ve o talepleri değerlendirip onaylayıp otomatik ödeme talimatı girişi sağlar yada talebi bir sebep ile reddeder.

2. Raporlar

- Personelin kendi işlem hareketleri için bir rapor geliştirilmeli. Personel kendi taleplerini ve detaylarını görebilir.
- Şirket günlük haftalık ve aylık raporlar ile ödeme yoğunluğu raporlanmalı.
- Şirket personel bazlı günlük haftalık ve aylık harcama yoğunluğunu raporlayabilmeli.
- Şirket günlük haftalık aylık onaylanan ve red edilen masraf miktarlarını raporlayabilmeli.
- Rapor tarafında veri tabanı üzerinde view sp kullanımı yapılabilir.
- Rapor için Dapper kullanımı yapılmalıdır.

3. Masraf işlemleri

- Masraf talebi için kategori bilgisi olmalı. Talep edilen ödeme kategorisi ödeme aracı ödeme yapılan konum ve fiş yada fatura vb dökümanlar sisteme yüklenebilmeli.
- Talep onaylandığında sanal bir ödeme simülasyonu gerçekleştirilerek ödeme işlemi sonlandırılmalı.

4. Yönetici işlemleri

- Şirket için tanımlı kullanıcılar sistem kurulumu ile birlikte default en az 2 kullanıcı ile açılmalı. (İnitial migration dosyasında hazırlanmalı.)
- Şirket yöneticileri personel tanımlarını gerçekleştirecektir.
- Ödeme kategorisi vb sabit alanlar ayrı tablolarda tutulacak olup sadece Yönetici tarafında ekleme çıkarma güncelleme işlemi yapılabilecektir.

NOT

- Modeller tasarlanırken isimlendirme standartlarına dikkat edilmeli.
- Modellerdeki alanlar sistemin taleplerini karşılayacak şekilde eksiksiz olmalı. (Personel için IBAN bilgisi vs.)
- Talep edilen tüm işlem setlerini karşılayan api metodları geliştirilmeli ve dokümante edilmeli.
- Talep edilen tüm işlem setleri bir api metodu olarak tasarlanmalı en basit hali ile test edilebilmelidir.
- Modeller için apiler üzerinde gerekli validasyonları ekleyip gerekli kontrolleri gerçekleştiriniz. Örn. ödeme talep tutarı sıfırdan küçük olamaz. Ödeme talep kategorisi boş olmamalı ve geçerli bir kayıt olmalı. Bir kategori silinmek istiyorsa o kategoride aktif talep olmamalı vb.

İhtiyaçlar

- Yetkilendirme icin jwt token altyapısı
- Masraf apileri, oluşturma , aktif masraf talepleri, geçmiş masraf talepleri apileri
- Tüm modeller için GET GETBYID POST PUT DELETE metodları hazırlanmalıdır.
- Tüm modeller için Controller ve metodlar eksiksiz olmalıdır.
- Proje iyi dokümante edilmelidir.

Tech stack

- Veri tabanı (Postgresql,Mssql)
- JWT token (Yetkilendirme)
- EF-Repositroy - Unitofwork
- Postman veya Swagger
- Rabbitmq (Opsiyonel)
- Redis (Opsiyonel)

Teslim kriterleri

- Proje kapsamında sadece swagger üzerinde bu senaryonun uçtan uca çalışması beklenmektedir.
- Postman yada herhangi bir api dokümantasyon aracı kullanarak sistem dokümante edilmeli.
- CodeFirst yada DbFirst yaklaşımlarından birisi ile ilerleyebilirsiniz.
- Code first geliştirme yaptıysanız yeni bir db oluştuğunda migrationların çalıştığından emin olunuz.
- Db first geliştirme yaptıysanız proje tesliminde db backup ve scriptleri ekleyiniz.
- Projenizin başarılı şekilde derlendiğinden emin olunuz.

Kriterler

- Değişken isimleri anlamlı, amaç neyse ona göre verilmiş.
- Metot isimleri ile metodun amacını net ifade edilmiş.
- Class'ların içindeki metot sayısı az ve amaca yönelik belirlenmiş.
- İç içe if ler olmayacak. Complexity düşük tutulmuş.
- Ayni kod parçasının tekrarlandığı duruma yer verilmemiş.
- Kodu okumayı zorlaştıran conditional complexity yaratılmamış
- Uzun metotlar (25 satırdan uzun olmamali.

- Hardcoded değerler Const olarak isimlendirerek kullanılmış.
- Aynı kodlama standardı tüm kodlarda uygulanmış. Farklı dosyalar arasında tutarsızlık yaratılmamış.
- Class'ların içindeki metotlar tek bir sorumluluk alanında odaklı yazılmış.
- Objectler arası bağımlılıklar enjekte edilmiş.
- Dependency Injection kullanılmış.
- Classlar arası bağımlılıkların en azda tutulmasına dikkat edilmiş.
- İnterface gibi abstraction lar ihtiyaç olduğu için kullanılmış. Gereksiz abstraction eklenmemiş.
- İnterface içeriği az ve tek sorumluluğa özgü metot imzası barındıracak şekilde tasarlanmış. Gereksiz design pattern kullanımı yapılmamış.
- Gerçekten bir problem çözmek için kullanılmış.
- Open-Closed Prensibine dikkat edilmiş. Kod genişlemeye açık, değişikliğe kapalı şeklinde tasarlanmış. Web/REST standartlarına dikkat edilmiş.
- Rest Api de açık metot parametreler için defansif validation kodları yazılmış olmalıdır.