📝 TKpay Ödeme Süreci ve QR Kod Entegrasyonu

Teknik Analiz: TKpay Ödeme Süreci ve QR Kod Entegrasyonu

Genel Yapı ve Süreç

- **Ödeme Talebi Oluşturma**
- **Ödeme Onayı ve Gerçekleştirme**
- **Harcama Takibi ve Kategorilendirme**
- **Bildirimler ve Geri Bildirim**

1. Ödeme Talebi Oluşturma

Adım 1.1: Kullanıcı, TKpay uygulamasına giriş yapar ve ortak para havuzunu seçer.

- **Gereken Bilgiler:** Kullanıcı kimlik doğrulaması, seçilen ortak havuz bilgisi.

Adım 1.2: "Ödeme Yap" seçeneğine tıklar ve ödeme yapmak istediği iş yerinin kategorisini(otel, araç kiralama, uçak bileti vs.) seçer.

- **Gereken Bilgiler:** Kategori adı *(otel, araç kiralama, uçak bileti, restoran, vb.), Kategori ID'si (veritabanında tanımlanmış kategorilere göre)*, İş Yeri Bilgileri: İş yeri adı, İş yeri ID'si *(veritabanında kayıtlı iş yerleri için)*, İş yeri IBAN'ı *(sistem tarafından otomatik olarak atanır, kullanıcıya görünmez)*

Adım 1.3: Ödeme yapmak istediği iş yerini seçer.

- **Gereken Bilgiler:** İş yeri bilgileri (kayıtlı iş yeri ID'si), İş yeri IBAN'ı *(sistem tarafından otomatik olarak atanır, kullanıcıya görünmez)*.

Adım 1.4: Ödeme detaylarını girer.

- **Gereken Bilgiler:** Miktar.

Adım 1.5: Ödeme talebi oluşturulur.

2. Ödeme Onayı ve Gerçekleştirme

Adım 2.2: Ödeme işlemi başlatılır.

- **Gereken Bilgiler:** İşlem başlatma durumu.

Adım 2.3: TKpay'in ödeme ağ geçidi üzerinden ödeme gerçekleştirilir.

Alt Adım 2.3.1: Ödeme Ağ Geçidi İşlemleri

- **Gereken Bilgiler:** Ödeme bilgileri (from, to, tutar).
- **İşlem:** QR kod üzerinden gelen bilgilerin doğrulanması.

```
if (validateQRData(fromAccount, toAccount, amount)) {
initiatePayment(fromAccount, toAccount, amount);
}
else {
sendErrorNotification("Ödeme bilgileri doğrulanamadı.");
}
```

Alt Adım 2.3.2: Doğrulama Başarılıysa Ödeme İşlemi

- **Gereken Bilgiler:** Onaylanmış ödeme bilgileri.

- **İşlem:** Ödeme işleminin gerçekleştirilmesi.

```
public class PaymentHandler {
  public static void main(String[] args) {
      String fromAccount = "123456";
      String toAccount = "654321";
      double amount = 100.0;

      if (paymentSuccessful()) {
          recordTransaction(fromAccount, toAccount, amount);
          sendSuccessNotification("Ödeme başarılı.");
      } else {
          reversePayment(fromAccount, toAccount, amount);
          sendErrorNotification("Ödeme başarısız, geri ödeme yapıldı.");
      }
  }

  public static boolean paymentSuccessful() {
      // Ödemenin başarılı olup olmadığını kontrol et
      // Bu örnek kodda, ödeme başarılı olarak varsayılıyor
      return true; // Gerçek duruma göre bu değeri değiştirin
  }

  public static void recordTransaction(String fromAccount, String toAccount, double amount) {
      // İşlemi kaydet
      System.out.println("İşlem kaydedildi: " + amount + " TL, " + fromAccount + " hesabından " + toAccount + "
  }

  public static void sendSuccessNotification(String message) {
      // Başarı bildirimi gönder
      System.out.println("Başarı bildirimi gönderildi: " + message);
  }

  public static void reversePayment(String fromAccount, String toAccount, double amount) {
      // Ödemeyi geri al
      System.out.println("Geri ödeme yapıldı: " + amount + " TL, " + toAccount + " hesabından " + fromAccount +
  }

  public static void sendErrorNotification(String message) {
      // Hata bildirimi gönder
      System.out.println("Hata bildirimi gönderildi: " + message);
  }
}
```

Alt Adım 2.3.3: İşlem Başarılı Bildirimi

- **Gereken Bilgiler:** Başarılı işlem bilgileri.
- **İşlem:** Kullanıcıya bildirim gönderme.

`send_success_notification("Ödeme başarılı.")`

Alt Adım 2.3.4: İşlem Başarısız Bildirimi ve Geri Ödeme

- **Gereken Bilgiler:** Başarısız işlem bilgileri.
- **İşlem:** Kullanıcıya hata bildirimi gönderme ve geri ödeme işlemi.

```
public class PaymentHandler {
  public static void main(String[] args) {
```

```
        String fromAccount = "123456";
        String toAccount = "654321";
        double amount = 100.0;

        if (paymentFailed()) {
            reversePayment(fromAccount, toAccount, amount);
            sendErrorNotification("Ödeme başarısız, geri ödeme yapıldı.");
        }
    }

    public static boolean paymentFailed() {
        // Ödemenin başarısız olup olmadığını kontrol et
        // Bu örnek kodda, ödeme başarısız olarak varsayılıyor
        return true; // Gerçek duruma göre bu değeri değiştirin
    }

    public static void reversePayment(String fromAccount, String toAccount, double amount) {
        // Ödemeyi geri al
        System.out.println("Geri ödeme yapıldı: " + amount + " TL, " + toAccount + " hesabından " + fromAccount +
    }

    public static void sendErrorNotification(String message) {
        // Hata bildirimi gönder
        System.out.println("Hata bildirimi gönderildi: " + message);
    }
}
```

Ek Özellikler ve Gereksinimler

- **QR Kod İle Ödeme:** QR kod taraması ile ödeme işlemi başlatılabilir.
  - **İşlem:** QR kod verilerinin işlenmesi ve ödeme ağ geçidine iletilmesi.

```
public class QRPaymentProcessor {
    public static void main(String[] args) {
        String qrData = "sampleQRData";
       processQRPayment(qrData);
    }

    public static void processQRPayment(String qrData) {
        String[] parsedData = parseQRData(qrData);
        String fromAccount = parsedData[0];
        String toAccount = parsedData[1];
        double amount = Double.parseDouble(parsedData[2]);

        if (validateQRData(fromAccount, toAccount, amount)) {
            initiatePayment(fromAccount, toAccount, amount);
        } else {
            sendErrorNotification("QR kod bilgileri doğrulanamadı.");
        }
    }

    public static String[] parseQRData(String qrData) {
        // QR verisini ayrıştır
        // Bu örnekte, veriyi hardcoded olarak ayrıştırıyorum. Gerçek duruma göre bu fonksiyonu düzenleyin.
        return new String[] {"123456", "654321", "100.0"};
    }
```

```
25
26   public static boolean validateQRData(String fromAccount, String toAccount, double amount) {
27       // QR verisini doğrula
28       return true; // Gerçek duruma göre bu değeri değiştirin
29   }
30
31   public static void initiatePayment(String fromAccount, String toAccount, double amount) {
32       // Ödemeyi başlat
33       System.out.println("Ödeme başlatıldı: " + amount + " TL, " + fromAccount + " hesabından " + toAccount + "
34   }
35
36   public static void sendErrorNotification(String message) {
37       // Hata bildirimi gönder
38       System.out.println("Hata bildirimi gönderildi: " + message);
39   }
40 }
```

- **Sadece Kayıtlı İş Yerlerine Ödeme:** Ödemeler yalnızca sistemde kayıtlı iş yerlerine yapılabilir.
  - **İşlem:** İş yeri bilgileri doğrulaması.

```
1  public class BusinessPaymentProcessor {
2    public static void main(String[] args) {
3        String fromAccount = "123456";
4        String toAccount = "654321";
5        double amount = 100.0;
6
7        if (isRegisteredBusiness(toAccount)) {
8            initiatePayment(fromAccount, toAccount, amount);
9        } else {
10           sendErrorNotification("Geçersiz iş yeri.");
11       }
12   }
13
14   public static boolean isRegisteredBusiness(String toAccount) {
15       // İş yerinin kayıtlı olup olmadığını kontrol et
16       // Bu örnekte, iş yeri kayıtlı olarak varsayılıyor
17       return true; // Gerçek duruma göre bu değeri değiştirin
18   }
19
20   public static void initiatePayment(String fromAccount, String toAccount, double amount) {
21       // Ödemeyi başlat
22       System.out.println("Ödeme başlatıldı: " + amount + " TL, " + fromAccount + " hesabından " + toAccount + "
23   }
24
25   public static void sendErrorNotification(String message) {
26       // Hata bildirimi gönder
27       System.out.println("Hata bildirimi gönderildi: " + message);
28   }
29 }
```

- **Başarısız İşlemde Geri Ödeme:** Başarısız işlemlerde para, from hesabına geri döner.
  - **İşlem:** Geri ödeme işlemi ve bildirim gönderme.

```
1  public class PaymentProcessor {
2    public static void main(String[] args) {
```

```
      String fromAccount = "123456";
      String toAccount = "654321";
       double amount = 100.0;

      if (paymentFailed()) {
           reversePayment(fromAccount, toAccount, amount);
           sendErrorNotification("Ödeme başarısız, geri ödeme yapıldı.");
       }
   }

   public static boolean paymentFailed() {
      // Ödemenin başarısız olup olmadığını kontrol et
      // Bu örnek kodda, ödeme başarısız olarak varsayılıyor
       return true; // Gerçek duruma göre bu değeri değiştirin
   }

   public static void reversePayment(String fromAccount, String toAccount, double amount) {
       // Ödemeyi geri al
      System.out.println("Geri ödeme yapıldı: " + amount + " TL, " + toAccount + " hesabından " + fromAccount + "
   }

   public static void sendErrorNotification(String message) {
      // Hata bildirimi gönder
      System.out.println("Hata bildirimi gönderildi: " + message);
   }
}
```