💻 Teknik Analiz

1. Kullanıcı Hesapları Yönetimi (Account Management)

Account Sınıfı:

- `USER_ID` : Kullanıcı kimliği, hesapla ilişkilendirilir.
- `LIST<CARD>` : Bu hesaba bağlı kartların listesi.

User Sınıfı:

- `LIST<ACCOUNT>` : Kullanıcının sahip olduğu hesapların listesi.

Teknik Detaylar:

- **Kullanıcı kayıt ve giriş işlemleri:** Kullanıcı, uygulamaya kaydolduğunda yeni bir `Account` ve `User` nesnesi oluşturulur.
- **Kimlik doğrulama:** JWT (JSON Web Tokens) kullanılarak kimlik doğrulama sağlanır.
- **Hesap bilgileri güncelleme:** Kullanıcı, kendi hesap bilgilerini güncelleyebilir.

2. Cüzdan Yönetimi (Wallet Management)

Wallet Sınıfı:

- `LIST<ACCOUNT>` : Cüzdana bağlı hesaplar.
- `LIST<PAYMENT>` : Cüzdandan yapılan ödemeler.
- `LIST<TRANSACTION>` : Cüzdanda gerçekleşen işlemler.
- `TYPE` : Cüzdanın türü (kişisel, grup).

MilPoint Sınıfı:

- `ACCOUNT_ID` : Mil puanlarının bağlı olduğu hesap kimliği.
- `IND_WALLET_ID` : Mil puanlarının bağlı olduğu bireysel cüzdan kimliği.
- `AMOUNT` : Mil puanı miktarı.

Teknik Detaylar:

- **Cüzdan oluşturma:** Kullanıcı, yeni bir cüzdan oluşturabilir. Bu işlem sırasında cüzdanın türü belirlenir.
- **Grup cüzdanı yönetimi:** Grup lideri, grup cüzdanı oluşturabilir ve üyeleri davet edebilir. Davet bağlantıları `LinkGenerator` sınıfı ile yönetilir.
- **Mil puanları:** Kullanıcılar, harcamaları karşılığında mil puanı kazanabilir ve bunları `MilPoint` sınıfında yönetebilir.

3. Ödeme İşlemleri (Payment Transactions)

Payment Sınıfı:

- `WALLET` : Ödemenin yapıldığı cüzdan.
- `RECEIVER` : Ödemenin alıcısı.

Transaction Sınıfı:

- `ACCOUNT_ID` : İşlemin yapıldığı hesap.
- `WALLET_ID` : İşlemin yapıldığı cüzdan.

Receiver Sınıfı:

- `ACCOUNT` : Alıcının hesabı.

Teknik Detaylar:

- **Ödeme oluşturma ve işleme:** Kullanıcılar, cüzdandan alıcıya ödeme yapabilir. Ödeme işlemleri `Payment` sınıfı ile temsil edilir.
- **İşlem yönetimi:** Hesaplar ve cüzdanlar arasındaki tüm işlemler `Transaction` sınıfı ile yönetilir.
- **Ödeme ağ geçidi entegrasyonu:** Stripe veya Iyzico gibi bir ödeme ağ geçidi kullanılarak ödemeler güvenli bir şekilde işlenir.

4. Kart Yönetimi (Card Management)

Card Sınıfı:

- `ACCOUNT`: Kartın bağlı olduğu hesap.

Teknik Detaylar:

- **Kart ekleme:** Kullanıcı, hesabına yeni bir kart ekleyebilir. Kart bilgileri `Card` sınıfında saklanır.
- **Kart bilgileri güncelleme:** Kullanıcı, kart bilgilerini güncelleyebilir.

5. Bağlantı ve Davet Yönetimi (Link and Invitation Management)

LinkGenerator Sınıfı:

- `LINK`: Kullanıcıya özel oluşturulan bağlantı.
- `USER_ID`: Bağlantıyı oluşturan kullanıcının kimliği.
- `WALLET_ID`: Bağlantının bağlı olduğu cüzdan kimliği.

Teknik Detaylar:

- **Davet bağlantıları oluşturma:** Grup lideri, yeni üyeleri davet etmek için `LinkGenerator` kullanarak bağlantı oluşturur.
- **Bağlantıların yönetimi:** Bağlantılar, kullanıcı kimlikleri ve cüzdan kimlikleri ile ilişkilendirilir.

Veri Akışı ve Kullanım Senaryoları

Kullanıcı Hesabı Oluşturma

1. Kullanıcı, kayıt formunu doldurur ve gönderir.
2. Backend, yeni bir `User` ve `Account` nesnesi oluşturur ve veritabanına kaydeder.
3. Kullanıcıya başarı bildirimi gönderilir.

Grup Cüzdanı Oluşturma

1. Grup lideri, grup cüzdanı oluşturma formunu doldurur.
2. Backend, yeni bir `Wallet` nesnesi oluşturur ve `TYPE` değerini grup cüzdanı olarak ayarlar.
3. `LinkGenerator` kullanılarak davet bağlantısı oluşturulur.
4. Davet bağlantısı grup liderine döner ve grup lideri bu bağlantıyı grup üyeleriyle paylaşır.

Ödeme Yapma

1. Kullanıcı, ödeme formunu doldurur ve gönderir.
2. Backend, `Payment` nesnesi oluşturur ve ilgili `WALLET` ve `RECEIVER` değerlerini ayarlar.
3. Ödeme ağ geçidi üzerinden ödeme işlemi gerçekleştirilir.
4. Ödeme başarılı olursa, işlem kaydedilir ve kullanıcıya bildirim gönderilir.

İşlem Geçmişi Takibi

1. Kullanıcı, işlem geçmişini görüntülemek istediğinde backend'den ilgili `Wallet` nesnesi ve `LIST<TRANSACTION>` istenir.
2. Backend, ilgili işlemleri veritabanından çeker ve kullanıcıya döner.
3. Kullanıcı, işlem geçmişini arayüzde görüntüler.