

YAZILIM İNŞAASI DERSI

**Banka Otomasyon Sistemi Analiz Raporu**

Java programlama dili ile geliştirilen güvenli banka otomasyon

sisteminin teknik analiz dokümanı

[](https://gamma.app/?utm_source=made-with-gamma)

**Proje Kapsamı ve Amaç**

Bu sistem, modern bankacılık ihtiyaçlarını karşılamak üzere tasarlanmış

kapsamlı bir otomasyon çözümüdür. Kullanıcılar güvenli bir giriş paneli üzerinden hesaplarına erişerek temel bankacılık işlemlerini gerçekleştirebilir.

**Temel Özellikler:**

* Güvenli kullanıcı kimlik doğrulama sistemi
* Bakiye yönetimi (yatırma/çekme işlemleri)
* Para transfer modülü (Havale/EFT)
* Dijital dekont oluşturma ve kayıt sistemi



**Platform:** Java konsol uygulaması

**Veri Saklama:** TXT dosya sistemi

**Fonksiyonel Gereksinimler**

**1**

**Güvenlik ve Giriş Yönetimi**

* Kullanıcı adı ve şifre doğrulama mekanizması
* Şifre güncelleme fonksiyonu
* Oturum yönetimi ve güvenli çıkış

**2**

**Hesap Hareketleri**

* Anlık bakiye sorgulama
* Para yatırma ile bakiye artışı
* Para çekme ve yetersiz bakiye kontrolü

**3**

**Para Transfer Yönetimi**

* IBAN, GSM ve Hesap No ile transfer
* Transfer öncesi bakiye kontrolü
* Transfer onay mekanizması

**4**

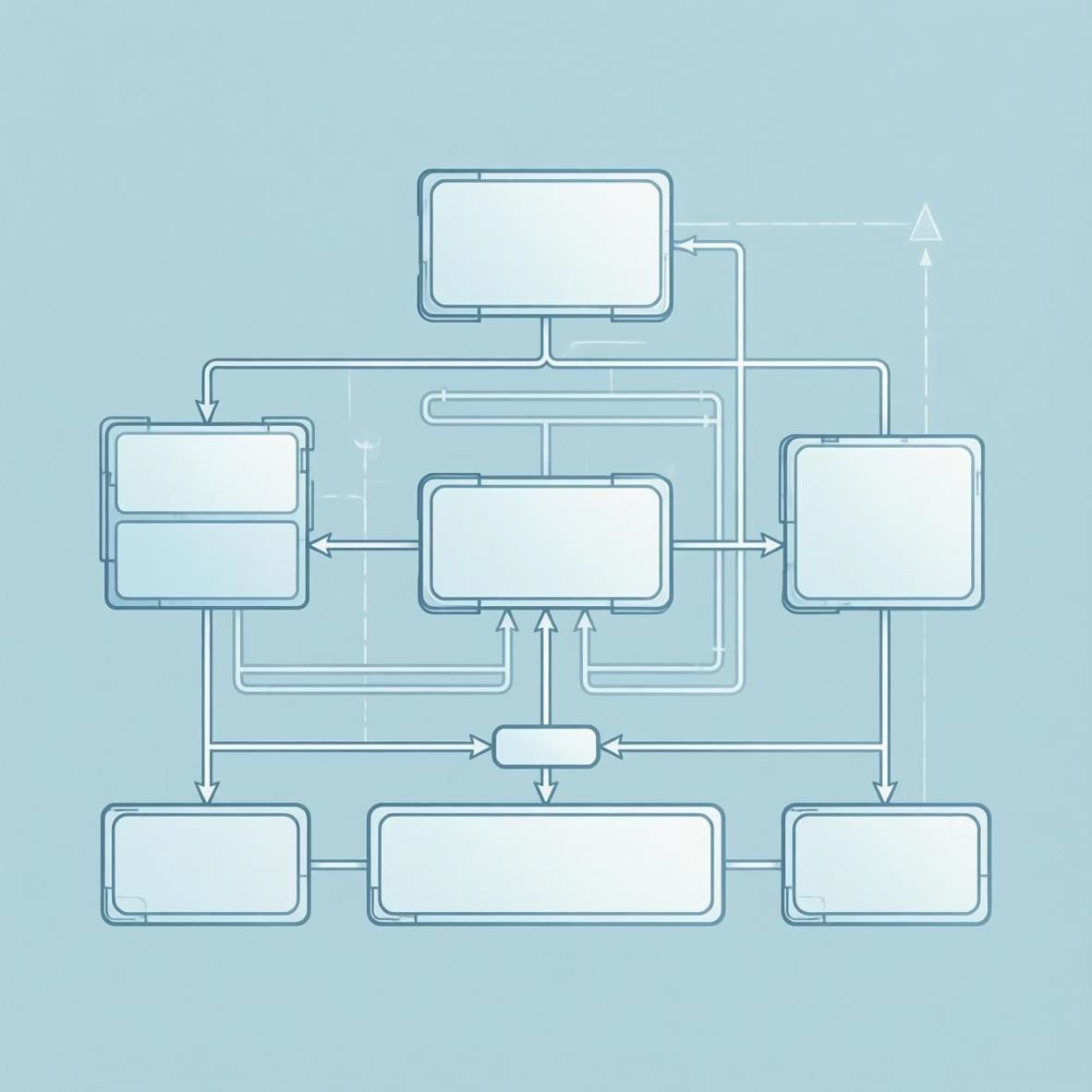
**Veri ve Kayıt Yönetimi**

* Her işlem için dekont.txt oluşturma
* Geçmiş işlem listeleme
* İşlem tarih ve saat kayıtları

**Fonksiyonel Olmayan Gereksinimler ve Kısıtlar**

##### Kalite Gereksinimleri

**Sistem Kısıtları**



**Teknolojik Sınırlamalar:**

* Veritabanı yerine dosya sistemi (TXT formatı)



**Güvenilirlik**

Hatalı girişlerde ve yetersiz bakiye durumlarında sistem kararlılığı. Kullanıcı doğru bilgilendirilir ve hata yönetimi sağlanır.



**Kullanılabilirlik**

Anlaşılır konsol menüleri, görsel çerçeveler ve kullanıcı dostu arayüz tasarımı ile kolay navigasyon.



**Veri Bütünlüğü**

Her işlemde dekont.txt güncellenir, veriler tutarlı saklanır ve veri kaybı önlenir.

* RAM tabanlı koleksiyonlar (ArrayList yapıları)
* Komut satırı arayüzü (CLI) zorunluluğu
* Tek kullanıcılı çalışma ortamı

UML DİYAGRAMI – BANKA SİSTEMİ

metin, ekran görüntüsü, diyagram, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

## Kullanım Senaryoları (Use Case)



### UC1: Sisteme Giriş

Kullanıcı, sistemde kayıtlı kullanıcı adı ve şifresini girerek

kimlik doğrulaması yapar. Başarısız girişlerde tekrar deneme hakkı tanınır.

### UC2: Para Transferi

IBAN, GSM numarası veya hesap numarası kullanarak EFT/Havale işlemi gerçekleştirilir. Transfer öncesi bakiye kontrolü yapılır.



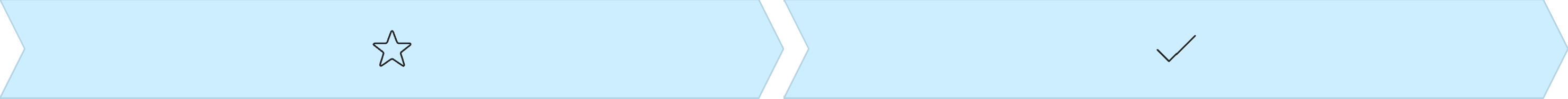
### UC3: Bakiye Sorgulama

Kullanıcı mevcut bakiyesini görüntüler ve geçmiş tüm işlemlerini tarih sırasına göre listeleyebilir.

### UC4: Şifre Değiştirme

Mevcut şifre kontrolü yapıldıktan sonra yeni şifre belirlenir. Güvenlik için şifre doğrulama adımı uygulanır.

**Sistem İşlem Akışı**

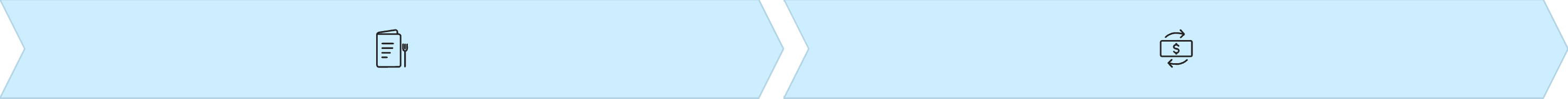


##### Program Başlangıcı

GirisPaneli sınıfı çağrılır, sistem başlatılır

##### Kimlik Doğrulama

Kullanıcı adı ve şifre kontrol edilir

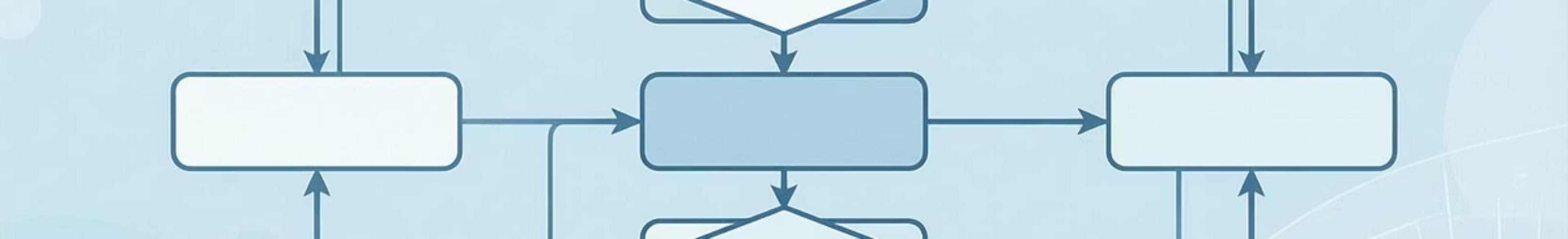


##### Ana Menü

İşlem seçimi ekranı gösterilir

##### İşlem Gerçekleştirme

Seçilen işlem yürütülür, dekont üretilir



Transfer işlemlerinde ek adımlar: Yöntem seçimi → Hedef bilgi girişi → Miktar belirleme → Bakiye kontrolü → İşlem onayı → Dekont

oluşturma. Her adımda kullanıcıya geri dönüş imkanı sunulur.

**Sınıf Yapısı ve Mimari**

**GirisPaneli**

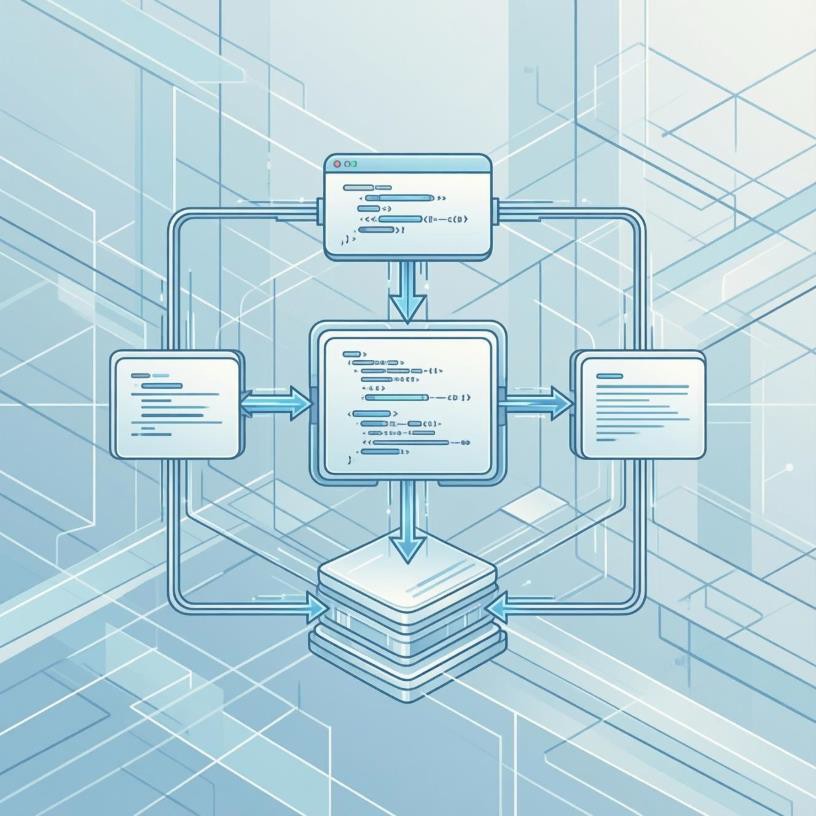


**Sorumluluk:** Kullanıcı kimlik doğrulama

**Ana Metot:**

* Giris(): Döngüsel doğrulama kontrolü
* sifreKontrol(): Şifre validasyonu

**Hesap**



**Sorumluluk:** İş kuralları yönetimi

**Ana Metotlar:**

* paraYatir(): Bakiye artırma
* paraCek(): Bakiye azaltma
* paraGonder(): Transfer işlemi
* dekontOlustur(): Kayıt oluşturma

**BankaUygulamasi**



**Sorumluluk:** Sistem orkestrasyonu **Ana Metot:**

* main(): Program başlatma ve menü

yönetimi

* menuGoster(): Kullanıcı arayüzü

**Modüler Mimari:** Sistem üç katmanlı yapı ile tasarlanmıştır: Giriş katmanı (Login), İş mantığı katmanı (Business Logic), ve Kullanıcı arayüzü katmanı (UI). Bu yapı, kodu sürdürülebilir ve genişletilebilir kılar.

## Arayüz Tasarımı

**Ana Menü Yapısı**

╔════════════════════════════════════╗

║ ANA MENÜ ║

╠════════════════════════════════════╣

║ 1.

Kullanıcı deneyimini optimize etmek için görsel çerçeveli konsol tasarımı uygulanmıştır. Her menü

öğesi emoji ikonları ile desteklenerek görsel hiyerarşi

sağlanmıştır.

###### Menü Özellikleri:

* Çerçeveli ve düzenli görünüm
* Anlamlı emoji ikonları
* Net seçenek numaralandırması
* Kolay navigasyon

**Teknik Özellikler ve Veri Yönetimi**

# 3

#### Katmanlı Mimari

Login, Logic ve UI katmanları ile modüler yapı

# 100%

#### Veri Bütünlüğü

Her işlemde dekont garantisi

**4**

#### Ana Sınıf

GirisPaneli, Hesap, BankaUygulamasi ve

yardımcı sınıflar



**Nesne Modeli Örneği:** Çalışma zamanında RAM'de benimHesabim nesnesi (bakiye: 1000.0 TL, isim: "Ali Yılmaz") ve hareketler ArrayList'i (işlem geçmişi: "09/01/2026 - Hesap açıldı", "09/01/2026 - Para Yatırma: +500 TL") aktif olarak tutulur.

**Kapsülleme:** Hesap sınıfı üzerinden sağlanan encapsulation ile veri güvenliği korunur. Private değişkenler ve public metotlar ile kontrollü erişim sağlanır.

**Sonuç ve Değerlendirme**

###### preencoded.pngProje Başarıları

**Modüler Mimari**

Login, Logic ve UI katmanları net ayrımı ile sürdürülebilir kod yapısı

**Veri Güvenliği**

Kapsülleme ve dosya yönetimi ile veri bütünlüğü korunması

**Savunmacı Programlama**

Yetersiz bakiye, yanlış şifre gibi hatalara karşı defensive

programming prensipleri



**Önemli Not:** Bu sistem sadece bir kod bloğu değil, profesyonel yazılım mühendisliği prensiplerinin uygulandığı kapsamlı bir otomasyon çözümüdür.

Sistem, modern yazılım geliştirme standartlarına uygun olarak tasarlanmış, kullanıcı hatalarına karşı dayanıklı ve genişletilebilir bir yapıya sahiptir. Dosya tabanlı veri saklama ile basitlik sağlanırken, nesne yönelimli programlama prensipleri ile profesyonel bir mimari

oluşturulmuştur.