

KÜTÜPHANE OTOMASYON SİSTEMİ PROJE RAPORU

Hazırlayanlar: Yusuf Bozbeşparmak
Sude Naz Çelik

223 405 023
233 405 042

Bölüm: Bilgisayar Mühendisliği

Tarih: 31.12.2025

Ders/Kapsam: Nesne Tabanlı Programlama

1. ÖZET

Bu proje, kütüphane iş süreçlerinin dijitalleştirilmesi, veri tutarlılığının sağlanması ve işlem hızının artırılması amacıyla geliştirilmiş bir Kütüphane Otomasyon Sistemi'dir. Proje kapsamında Java programlama dili kullanılmış, Nesne Yönelimli Programlama prensipleri temel alınmış ve veri yönetimi için optimize edilmiş veri yapıları entegre edilmiştir. Sistem; kitap takibi, üye yönetimi ve ödünç alma/iade süreçlerini başarılı bir şekilde yürütmektedir.

2. GİRİŞ VE PROJENİN AMACI

Geleneksel kütüphane yönetimindeki manuel kayıt sistemleri, veri kaybına ve işlem hatalarına açıktır. Bu projenin temel amacı:

- Kütüphane envanterinin dinamik olarak yönetilmesi,
- Üye işlemlerinin ve kitap hareketlerinin (ödünç/iade) kayıt altına alınması,
- Arama ve sorgulama işlemlerinin algoritmik olarak hızlandırılmasıdır.

Proje, teorik bilgisayar mühendisliği kavramlarının (OOP, Kapsülleme, Polimorfizm, Algoritmalar) pratik bir uygulama üzerinde somutlaştırılmasını hedeflemiştir.

3. KULLANILAN TEKNOLOJİLER VE YÖNTEMLER

3.1. Yazılım Dili ve Ortamı

- **Programlama Dili:** Java SE (Standard Edition)
- **Geliştirme Ortamı (IDE):** IntelliJ IDEA
- **Versiyon Kontrol:** GitHub

3.2. Nesne Yönelimli Programlama (OOP) Prensipleri

Sistemin modülerliği ve sürdürülebilirliği için aşağıdaki prensipler uygulanmıştır:

- **Encapsulation (Kapsülleme):** *Book* ve *Member* sınıflarındaki hassas veriler *private* erişim belirleyicileri ile korunmuş, erişim *getter* / *setter* metotları ile sağlanmıştır.
- **Inheritance (Kalıtım):** *LibraryItem* ve *LibraryPerson* temel sınıfından *Dvd*, *Book*, *Magazine* ve *Members*, *Officers* sınıfları türetilerek kod tekrarı önlenmiştir.
- **Abstraction (Soyutlama):** Kütüphane işlemleri için arayüzler (Interfaces) tanımlanarak sistemin genel çatısı oluşturulmuştur.

4. SİSTEM MİMARİSİ VE VERİ YAPILARI

Projenin performansını belirleyen en önemli unsur, kullanılan veri yapılarıdır. Bilgisayar mühendisliği perspektifiyle aşağıdaki yapılar tercih edilmiştir:

4.1. Veri Yapıları Analizi

1. ArrayList:

Kitap ve üye listelerinin dinamik olarak tutulması için kullanılmıştır.

Boyutun önceden belirsiz olduğu durumlarda esneklik sağladığı için tercih edilmiştir.

2. HashMap (veya HashTable):

a. Kullanım Alanı: Ürünlerin, kullanıcı ve görevlilerin ID'ye göre aranması.

b. Mühendislik Gerekçesi: ArrayList üzerinde arama işlemi $O(n)$ zaman karmaşıklığına sahipken, HashMap kullanılarak bu işlem anahtar-değer (key-value) eşleşmesi sayesinde ortalama $O(1)$ karmaşıklığına düşürülmüş ve sistem performansı optimize edilmiştir.

4.2. Sınıf Tasarımı (Class Design)

Sistem temel olarak aşağıdaki sınıflardan oluşmaktadır:

- **Book (Kitap):** Kitap adı, yazar, ID ve durum (rafta/ödünçte) özelliklerini barındırır.
- **DVD:** Dvd adı, ID, markası ve durum (rafta/ödünçte) özelliklerini barındırır.
- **Magazine (Magazin):** Magazin adı, ID, sayfa sayısı, markası ve durum(rafta/ödünçte) özelliklerini barındırır.

- **Members (Üye):** Üye adı, soyadı, ID ve telefon numarası bilgilerini tutar.
- **Officers (Görevli):** Görevli adı, soyadı, ID ve telefon numarası bilgilerini tutar.
- **Request (İstek):** İstek ID, isteği yapan kişi ID, istenen eşya ID, istek tipi, durum bilgilerini tutar.
- **LibraryItem (Kütüphanedeki eşyalar):** Eşya ID, isimlerini, soyisimlerini bilgilerini tutar.
- **LibraryPerson (Kütüphanedeki kişiler):** Kişi ID, isimlerini, soyisimlerini, telefon numarası bilgilerini tutar.

5. FONKSİYONEL GEREKSİNİMLER VE ALGORİTMALAR

Sistem aşağıdaki temel fonksiyonları yerine getirmektedir:

- **Kitap Ekleme/Silme:**
Envantere yeni nesneler eklenir veya mevcut nesneler listeden çıkarılır.
- **Ödünç Verme (Lending Logic):**
 - a.** Kullanıcının ödünç alma limiti kontrol edilir.
 - b.** Kitabın stok durumu (*isAvailable*) kontrol edilir.
 - c.** Şartlar sağlanıyorsa kitabın durumu güncellenir ve kullanıcı hesabına işlenir.

6. SONUÇ VE KAZANIMLAR

Bu proje ile birlikte, bir yazılımın yaşam döngüsü (analiz, tasarım, kodlama) deneyimlenmiştir. Özellikle Java Collection Framework yapısının etkin kullanımı ve algoritmik karmaşıklığın (*Time Complexity*) performansa etkisi üzerine pratik bilgi edinilmiştir. Geliştirilen sistem, temel kütüphane ihtiyaçlarını karşılayacak düzeyde olup, ileride veritabanı (*SQL*) entegrasyonu ve grafik arayüz (*GUI*) eklenerek geliştirilmeye açıktır.

7. EKLER

GitHub Linki : <https://github.com/Ykpkara/LibraryAutomation>