

OTEL YÖNETİM SİSTEMİ

FİNAL RAPORU (IMPLEMENTATION REPORT)

Proje: Otel Yönetim Sistemi

Bu rapor, Java programlama dili kullanılarak geliştirilen Otel Yönetim Sistemi yazılımının final aşamasını kapsamaktadır. Proje, Analiz Raporu ve oluşturulan Tasarım Raporu temel alınarak gerçekleştirilmiştir.

Final raporunun temel amacı; sistemin gerçekleştirim (implementation) sürecini detaylı biçimde belgelemek, analiz ve tasarım aşamalarında belirlenen hedeflerle ne ölçüde uyum sağlandığını değerlendirmek, kodlanan özellikleri açıklamak ve mevcut durumu bütüncül bir bakış açısıyla ortaya koymaktır.

Programlama Dili: Java

1. GİRİŞ

Bu kapsamda raporda şu konular ele alınmıştır:

- Gerçekleştirim süreci ve mimari yapı
- Kodun analiz ve tasarım raporlarıyla uyumu
- Sistemin fonksiyonel özellikleri (Oda, Müşteri, Rezervasyon, Check-in/Out)
- Kullanıcı kılavuzu ve işlem akışları
- Karşılaşılan teknik zorluklar ve çözümleri

Proje boyunca nesne yönelimli yazılım mühendisliği prensipleri (özellikle SOLID, Katmanlı Mimari ve Separation of Concerns) titizlikle uygulanmış; modülerlik, okunabilirlik ve genişletilebilirlik ön planda tutulmuştur.

2. GERÇEKLEŞTİRİM SÜRECİ

Otel Yönetim Sistemi, Java (JDK 8+) sürümü kullanılarak geliştirilmiştir. Uygulama, analiz raporunda belirtilen sistem kısıtlarına (veritabanı kullanılmaması, CLI arayüz) tam uyumlu şekilde komut satırı tabanlı olarak kodlanmıştır.

Gerçekleştirim süreci şu adımlar izlenerek yürütülmüştür:

Analiz ve Tasarım İncelemesi:

Tasarım raporunda belirlenen 3 katmanlı mimari (Sunum – İş Mantığı – Veri) yapıtaşı olarak kabul edilmiş ve paket yapısı buna göre kurgulanmıştır.

Katmanlı Kodlama:

- **Sunum Katmanı:**
MenuController sınıfı ile kullanıcı etkileşimi, menü navigasyonu ve Scanner tabanlı veri girişi kodlanmıştır. Main sınıfı sadece başlatıcı görevini üstlenmiştir.
- **İş Mantığı Katmanı:**
ReservationService sınıfı oluşturularak; tarih çakışma kontrolleri, fiyat hesaplama ve oda durumu yönetimi bu sınıfta merkezleştirilmiştir.
- **Veri Katmanı:**
FileManager sınıfı ile .txt dosyalarına (rooms, customers, reservations) okuma/yazma işlemleri ve “Auto-Increment ID” mantığı kodlanmıştır.
- **Model Sınıfları:**
Room, Customer ve Reservation sınıfları, veriyi taşıyan ve kapsülleyen (Encapsulation) varlıklar olarak tasarlanmıştır.

Test ve Doğrulama:

- ReservationService.isDateAvailable metodu farklı tarih aralıklarıyla test edilerek çakışma kontrolü doğrulanmıştır.
- Dosya okuma/yazma işlemleri manuel testlerle (programı kapatıp açarak verinin korunup korunmadığı) kontrol edilmiştir.

3. ANALİZ VE TASARIM RAPORLARI İLE UYUM

Sistem, analiz ve tasarım raporlarında tanımlanan gereksinimleri eksiksiz karşılayacak şekilde geliştirilmiştir.

3.1 Fonksiyonel Gereksinimlerin Karşılanması

Gereksinim	Karşılanma Durumu	Açıklama
FR1 – Oda Yönetimi	Tamamen Karşılındı	Yeni oda ekleme, özellikleri (tip, kapasite, fiyat) belirleme ve listeleme (UC1, UC2).
FR2 – Müşteri Yönetimi	Tamamen Karşılındı	Müşteri kaydı oluşturma, iletişim bilgilerini saklama ve listeleme (UC3, UC4).
FR3 – Rezervasyon	Tamamen Karşılındı	Tarih aralığı kontrolü, çakışma engelleme, fiyat hesaplama ve kayıt oluşturma (UC5, UC6).

Gereksinim	Karşılanma Durumu	Açıklama
FR4 – Check-in/Out	Tamamen Karşılandı	Oda durumunu (DOLU/BOŞ) ve rezervasyon durumunu (AKTIF/TAMAMLANDI) senkronize güncelleme (UC7, UC8).
FR5 – Sistem Kontrolü	Tamamen Karşılandı	Menü tabanlı seçim, hatalı girişlerde programın çökmesini engelleyen yapılar.

3.2 Tasarım Uyumu

- **MenuController:**
Sadece UI işlerinden sorumludur. İş mantığı (örneğin fiyatın gün sayısına göre hesaplanması) içermez. → Sunum Katmanı Uyumu.
- **ReservationService:**
Tüm kritik kararları (Oda müsait mi? Tarih geçerli mi? Check-in yapılabilir mi?) veren sınıftır. → İş Mantığı Katmanı Uyumu.
- **FileManager:**
Veri erişimini izole etmiştir. Main veya MenuController dosya yollarını bilmez, sadece FileManager bilir. → Veri Katmanı Uyumu.
- **Veri Yapısı:**
Tasarımda belirtildiği gibi veriler .txt dosyalarında saklanmakta ancak performans için ArrayList içinde işlenmektedir.

4. GERÇEKLEŞTİRİLEN SİSTEM ÖZELLİKLERİ

4.1 Oda Yönetimi

- Kullanıcıdan Oda Tipi (Single, Double vb.), Kapasite ve Fiyat bilgisi alınır.
- Her oda için benzersiz bir ID (1, 2, 3...) otomatik atanır.
- Odalar listelenirken “BOŞ” veya “DOLU” durumları anlık olarak gösterilir.

4.2 Müşteri Yönetimi

- Ad, Soyad ve Telefon bilgisi ile kayıt oluşturulur.
- Müşteriler ID ile sistemde takip edilir ve rezervasyonlarda bu ID kullanılır.

4.3 Rezervasyon İşlemleri

- **Tarih Kontrolü:**
LocalDate kullanılarak, giriş tarihinin bugünden önce olmaması ve çıkış tarihinden küçük olması sağlanır.
- **Müsaitlik Kontrolü:**
ReservationService içindeki algoritma, seçilen tarihlerde o odanın başka bir “AKTIF” veya “REZERVE” kaydıyla çakışıp çakışmadığını denetler.
- **Fiyatlandırma:**
(Çıkış - Giriş) * Günlük Fiyat formülü ile otomatik hesaplama yapılır.

4.4 Check-in ve Check-out (Giriş/Çıkış)

- **Check-in:**
Sadece rezervasyon tarihi “bugün” olan işlemler için yapılır. Odayı “DOLU”, rezervasyonu “AKTIF” yapar.
- **Check-out:**
Müşteri çıkış yaptığında odayı tekrar “BOŞ” statüsüne çeker ve rezervasyonu “TAMAMLANDI” olarak kapatır.
- **Veri Güncelleme:**
Bu işlemler sırasında FileManager.updateAll... metotları ile dosya içeriği güncellenir.

5. KULLANICI KILAVUZU

5.1 Programın Başlatılması

1. Kaynak kodların (Main.java, MenuController.java vb.) bulunduğu dizine gidiniz.
2. Komut satırında (Terminal/CMD)
javac Main.java
komutu ile derleyiniz.
3. java Main
komutu ile uygulamayı başlatınız.
4. Eğer .txt dosyaları yoksa sistem otomatik oluşturacaktır.

5.2 Ana Menü

5. Uygulama açıldığında kullanıcıyı aşağıdaki menü karşılar:

=== OTEL YÖNETİM SİSTEMİNE HOŞ GELDİNİZ ===

--- ANA MENÜ ---

1. Oda Ekle
2. Oda Listele
3. Müşteri Ekle
4. Müşteri Listele
5. Rezervasyon Yap
6. Rezervasyon Listele
7. Check-in (Giriş)
8. Check-out (Çıkış)
9. Çıkış

Seçiminiz:

5.3 Örnek İşlem Akışı

- **Oda Ekle:**
1 seçilir → Tip: “Kral Dairesi”, Kapasite: 2, Fiyat: 1500 girilir. (ID: 101 oluşur)
- **Müşteri Ekle:**
3 seçilir → Ad: “Ahmet”, Soyad: “Yılmaz”, Tel: “0555...” girilir. (ID: 1 oluşur)
- **Rezervasyon Yap:**
5 seçilir → Müşteri ID: 1, Oda ID: 101 →
Giriş: 2023-12-01, Çıkış: 2023-12-05 girilir.
- **Check-in:**
Giriş günü geldiğinde 7 seçilir → Rezervasyon ID girilir → İşlem onaylanır.

6. TASARIMDAN SAPMALAR VE ALINAN KARARLAR

Proje kapsamında, gerçek dünya senaryolarından eğitim amaçlı olarak şu noktalarda bilinçli sapmalar yapılmıştır:

- **Veritabanı Yerine Dosya:**
SQL veritabanı kurulumunun karmaşıklığından kaçınmak ve dosya işleme (I/O) yeteneklerini göstermek için .txt dosyaları kullanılmıştır.

- **Senkron Çalışma:**
Çoklu kullanıcı desteği (Multi-thread) eklenmemiş, işlemler sıralı (Synchronous) yapılmıştır.
- **RAM Önbellekleme:**
Dosya okuma performansını artırmak için veriler her işlemde ArrayList içine alınmış, işlem bitince dosyaya yazılmıştır.

7. KARŞILAŞILAN ZORLUKLAR VE ÇÖZÜMLER

- **Tarih Formatlama:**
Kullanıcının “01-01-2023” veya “01/01/2023” gibi farklı formatlar girmesi DateTimeParseException hatasına yol açıyordu.

Çözüm:

Kullanıcıya “YYYY-MM-DD” formatını kullanması gerektiği belirtildi ve MenuController içinde try-catch blokları ile hatalı girişler yakalandı.

- **Veri Güncelleme:**
.txt dosyasındaki belirli bir satırı (örneğin sadece oda durumunu) değiştirmenin zorluğu.

Çözüm:

Güncelleme işlemlerinde (Check-in/Out) listenin tamamının bellekte güncellenip, dosyanın üzerine tamamen yeniden yazılması (overwrite) yöntemi uygulandı.

- **Çakışma Kontrolü:**
Tarih aralıklarının kesişimini bulmak.

Çözüm:

ReservationService içinde mantıksal bir algoritma
(giris1 < cikis2 && cikis1 > giris2)
kullanılarak çakışmalar tespit edildi.

8. EKSİKLER VE GELECEKTEKİ GELİŞTİRME ÖNERİLERİ

Mevcut sistem temel gereksinimleri başarıyla karşılamaktadır. Gelecek sürümler için şu geliştirmeler planlanabilir:

- Veritabanı Entegrasyonu (MySQL / SQLite)
- Grafik Arayüz (JavaFX veya Swing)
- Raporlama (Aylık Gelir, En Çok Tercih Edilen Oda Tipi vb.)
- Kimlik Doğrulama (Yönetici / Resepsiyonist Login)

9. SONUÇ

Bu çalışma kapsamında geliştirilen Otel Yönetim Sistemi, yazılım mühendisliği yaşam döngüsünün analiz, tasarım ve gerçekleştirim aşamalarını eksiksiz ve tutarlı biçimde içeren bütüncül bir projedir.

Kodlama aşamasında Java'nın nesne yönelimli yetenekleri kullanılarak; Katmanlı Mimari, Tek Sorumluluk Prensipli (SRP) ve Façade tasarım desenleri başarıyla uygulanmıştır. ReservationService gibi merkezi kontrol yapıları sayesinde iş kuralları arayüzden ayrıştırılmış, FileManager sayesinde veri erişimi izole edilmiştir.