

Tolunay AKDENİZ 20220305032

<https://github.com/tolunay23/nesne2>

1. Projenin Amacı

Projenin temel amacı; otel odaları, misafir bilgileri ve rezervasyon süreçlerinin yönetilebildiği, Java tabanlı, nesne yönelimli ve çok katmanlı bir masaüstü yazılım geliştirmektir. Sistem, Swing GUI ile kullanıcı dostu bir arayüz sunmaktadır.

2. Kullanılan OOP Prensipleri

Projede aşağıdaki OOP kavramları kullanılmıştır:

- Inheritance (Kalıtım):**

Guest ve Staff sınıfları, ortak özelliklerin tutulduğu soyut Person sınıfından türetilmiştir.

- Interface:**

Identifiable arayüzü tüm temel varlıklara (Room, Reservation, Person) ortak bir getId() sözleşmesi sağlar.

Repository<T> arayüzü de veri katmanı için sözleşme görevi görür.

- Polymorphism (Çok Bİçimlilik):**

Person referansları üzerinden Guest ve Staff nesneleri yönetilebilmekte, Repository arayüzü üzerinden farklı türlerde depolar aynı şekilde kullanılabilmektedir.

- Generic Class & Method:**

Repository<T> ve InMemoryRepository<T> sınıfları, Java generics yapısını kullanarak tip güvenli ve tekrar kullanılabilir bir veri erişim katmanı sunar.

- Generic Collections:**

Uygulama boyunca List, Set ve Map koleksiyonları kullanılmaktadır.

Örneğin; tüm odalar List<Room> içinde tutulurken, dolu oda ID'leri Set<Integer> ile, oda önbelleği ise Map<Integer, Room> yapısıyla yönetilmektedir.

- Lambda Functions:**

Rezervasyon sorgulama ve filtreleme işlemlerinde Java Stream API ile lambda ifadeleri kullanılmış,

ayrıca Swing butonlarının ActionListener tanımlamalarında da lambda kullanılarak kod sadeleştirilmiştir:

- btnAll.addActionListener(e -> loadAllRooms());
-

3. Sınıf Yapısı ve UML Class Diagram Açıklaması

Projede kullanılan başlıca sınıflar:

- **Identifiable (interface)**
 - Metot: int getId()
Tüm varlıkların bir kimlik numarasına sahip olmasını garanti eder.
- **Person (abstract class)**
 - Alanlar: id, name, phone
 - Davranış: Temel kişi bilgilerini içerir. Doğrudan örneklenmez, misafir ve personel için temel sınıf görevi görür.
- **Guest (Person'dan kalıtım)**
 - Ek alan: email
Otel misafirlerini temsil eder. Rezervasyonlar bu sınıf üzerinden ilişkilendirilir.
- **Staff (Person'dan kalıtım)**
 - Ek alan: position
Otel çalışanlarını temsil eder (ör. resepsiyonist, müdür).
- **Room**
 - Alanlar: id, number, type, capacity, pricePerNight
Odaların numara, kapasite, fiyat ve tip bilgilerini içerir.
- **Reservation**
 - Alanlar: id, guest, room, checkIn, checkOut
Bir misafir ile bir oda arasındaki konaklama ilişkisini ve tarih aralığını tutar.
- **Repository<T> (interface)**
 - Metotlar: findAll(), findById(int), save(T), deleteById(int)
Veri erişimi için generik bir sözleşme sunar.
- **InMemoryRepository<T> (Repository'den kalıtım)**
 - Uygulama boyunca bellekte çalışan basit bir veri deposu sağlar.
Gerçek bir veritabanına geçiş yapılmak istendiğinde bu sınıf yerine veritabanı tabanlı bir implementasyon yazılabilir.
- **HotelService**
 - Tüm iş kurallarını içerir:

- Rezervasyon oluşturma
 - Rezervasyon iptali
 - Belirli tarihlerde boş odaları bulma
- UI katmanı sadece bu servis ile konuşur, doğrudan veri katmanına erişmez.
-

4. Use Case Diagram (Açıklama)

Use Case diyagramında tek bir ana aktör bulunmaktadır:

- **Aktör:** Kullanıcı (resepsiyon görevlisi veya sistem kullanıcısı)

Bu aktör aşağıdaki senaryoları gerçekleştirir:

1. Oda Listeleme

Tüm odaları ve odaların kapasite/fiyat bilgilerini görüntüler.

2. Rezervasyon Oluşturma

Misafir bilgilerini ve tarih aralığını girerek uygun bir oda için rezervasyon oluşturur.

3. Rezervasyon İptali

Var olan bir rezervasyonu seçerek iptal eder.

Diyagramda, “Kullanıcı” aktöründen bu üç use case’e doğru ilişki (association) çizilmiştir.

5. Class Diagram (Açıklama)

Class diyagramında:

- Person sınıfı üstte yer alır; altına Guest ve Staff için kalıtım okları çizilir.
- Room ve Reservation sınıfları ayrı kutular halinde gösterilir; Reservation sınıfından Guest ve Room sınıflarına doğru bir “has-a” (composition/aggregation) ilişkisi bulunmaktadır.
- HotelService sınıfı, Repository<Room>, Repository<Guest> ve Repository<Reservation> arayüzlerini kullanan bir servis sınıfı olarak gösterilir.

Bu diyagram, sistemdeki temel nesnelerin birbirleriyle olan ilişkilerini özetler.

6. Katmanlı Mimari (Multitier Design)

Uygulama üç temel katmandan oluşmaktadır:

1. UI Layer (Kullanıcı Arayüzü Katmanı)

- HotelAppGUI sınıfı ve Swing bileşenlerinden oluşur.
- Kullanıcıyla etkileşime girer, buton tıklamaları gibi olayları yakalar ve HotelService üzerinden iş mantığını tetikler.

2. Business Layer (İş Mantığı Katmanı)

- HotelService sınıfı bu katmanı temsil eder.
- Rezervasyon oluşturma, iptal etme ve uygun oda bulma gibi kuralları içerir.
- Tarih çakışması kontrolü gibi kritik kontroller burada yapılır.

3. Data Layer (Veri Katmanı)

- Repository<T> ve InMemoryRepository<T> sınıfları ile temsil edilir.
- Verilerin bellekte saklanması ve geri getirilmesini sağlar.
- İleride gerçek bir veritabanına geçmek için uygun bir soyutlama sunar.

Bu yapı sayesinde katmanlar birbirinden ayrılmış, bakımı ve test edilmesi kolay bir mimari elde edilmiştir.

7. Uygulama Özellikleri

- Tüm otel odalarını listeleme
 - Oda kapasitesi ve gecelik fiyat bilgilerini gösterme
 - Belirli giriş–çıkış tarihleri için uygun (boş) odaları bulma
 - Misafir bilgisi girerek yeni rezervasyon oluşturma
 - Mevcut rezervasyonları listeleme
 - Seçilen rezervasyonu iptal etme
 - Swing tabanlı, sekmeli (TabbedPane) yapıda bir masaüstü arayüz
-

8. Kullanıcı Arayüzü (GUI) Tasarımı

Arayüz üç ana sekmeden oluşur:

1. Odalar Sekmesi

- Tüm odaların listelendiği tablo
- “Tüm Odaları Listele” ve “Belirli Tarihte Boş Odalar” butonları

2. Rezervasyonlar Sekmesi

- Tüm rezervasyonların listelendiği tablo
- “Rezervasyonları Listele” ve “Seçili Rezervasyonu İptal Et” butonları

3. Yeni Rezervasyon Sekmesi

- Misafir adı, telefon, e-posta, giriş–çıkış tarihi ve oda ID’sini alan form
- “Bu Tarihlerde Boş Odaları Göster” ve “Rezervasyon Oluştur” butonları

Renk Paleti:

- #EDE7C7 – Arka plan
- #8B0000 – Başlıklar ve vurgu renkleri
- #5B0202 – Daha koyu kırmızı tonları
- #200E01 – Yazı ve çizgi rengi

Bu palet sayesinde sade ve göz yormayan bir arayüz elde edilmiştir.

9. Sonuç

Otel Rezervasyon Sistemi; OOP prensiplerini doğru, anlaşılır ve genişletilebilir bir mimari ile uygulayan, kullanıcı dostu arayüze sahip bir Java masaüstü uygulamasıdır. Proje, derste belirtilen:

- Inheritance
- Interface
- Polymorphism
- Generic class/method
- List–Set–Map koleksiyonları
- Lambda functions
- Katmanlı mimari (UI, Business, Data)

gibi tüm teknik gereksinimleri karşılamaktadır. Ayrıca yapı, gerçek bir veritabanı entegrasyonu veya ek modüller ile ileride kolaylıkla genişletilebilecek esnekliktedir.

