Araç Kiralama Uygulaması, kullanıcıların araç kiralama işlemlerini kolay ve verimli bir şekilde gerçekleştirmelerini sağlayan bir yazılımdır. Bu uygulama, araçların eklenmesi, güncellenmesi, silinmesi ve listelenmesi gibi temel CRUD (Create, Read, Update, Delete) işlemlerini destekler.

**Proje Yapısı:**

Uygulama, iki projeden oluşmaktadır:

1. **Class Library (AracKiralamaLib):** Bu kütüphane, iş mantığı ve veri erişim katmanını içerir. Araç verilerinin yönetimi ve depolanması burada gerçekleştirilir. Örneğin, AracRepository sınıfı, araçların eklenmesi, güncellenmesi, silinmesi ve listelenmesi gibi işlemleri sağlar.
2. **Windows Forms Uygulaması (AracKiralamaUygulamasi):** Bu proje, kullanıcı arayüzünü sağlar ve kullanıcıların araç kiralama işlemlerini gerçekleştirmelerine olanak tanır. Formlar aracılığıyla kullanıcılar araç bilgilerini girebilir, güncelleyebilir veya silebilirler.

**Kullanılan Teknolojiler ve Prensipler:**

* **Nesne Yönelimli Programlama (OOP) Prensipleri:** Uygulama, OOP prensiplerine uygun olarak tasarlanmıştır. Bu sayede kodun yeniden kullanılabilirliği ve sürdürülebilirliği artırılmıştır.
* **Dependency Inversion Prensibi:** Bu prensip, üst seviye modüllerin alt seviye modüllere doğrudan bağımlı olmamasını sağlar. Bunun yerine, her iki modül de ortak bir arayüze bağımlıdır. Bu sayede, bağımlılıklar tersine çevrilir ve esneklik artırılır. Örneğin, IAracRepository arayüzü, veri erişim işlemlerini tanımlar ve AracRepository sınıfı bu arayüzü uygular. Kullanıcı arayüzü, doğrudan AracRepository sınıfına bağımlı olmak yerine, IAracRepository arayüzüne bağımlıdır. Bu sayede, veri erişim katmanı kolayca değiştirilebilir veya farklı bir implementasyon kullanılabilir.

**Temel Sınıflar ve Arayüzler:**

1. **Arac**: Araç bilgilerini temsil eder.
   * Özellikler:
     + Id: int
     + Marka: string
     + Model: string
     + Yil: int
     + Fiyat: decimal
2. **Musteri**: Müşteri bilgilerini temsil eder.
   * Özellikler:
     + Id: int
     + Isim: string
     + Soyisim: string
     + Telefon: string
3. **IKiralama**: Kiralama işlemlerini temsil eden arayüz.
   * Metodlar:
     + AracKirala(Musteri musteri, Arac arac, DateTime baslangic, DateTime bitis): void
     + KiralamaIptal(int kiralamaId): void
4. **IAracRepository**: Araç veri erişim işlemlerini temsil eden arayüz.
   * Metodlar:
     + AracEkle(Arac arac): void
     + AracSil(int aracId): void
     + AracGetir(int aracId): Arac
     + TumAraclariGetir(): List<Arac>
5. **IMusteriRepository**: Müşteri veri erişim işlemlerini temsil eden arayüz.
   * Metodlar:
     + MusteriEkle(Musteri musteri): void
     + MusteriSil(int musteriId): void
     + MusteriGetir(int musteriId): Musteri
     + TumMusterileriGetir(): List<Musteri>
6. **AracRepository**: IAracRepository arayüzünü uygulayan somut sınıf.
7. **MusteriRepository**: IMusteriRepository arayüzünü uygulayan somut sınıf.
8. **KiralamaServisi**: IKiralama arayüzünü uygulayan somut sınıf.

**Sınıflar Arası İlişkiler:**

* **Bağımlılık İlişkisi (Dependency)**: Bir sınıfın başka bir sınıfı kullanması durumunda oluşur.
  + KiralamaServisi sınıfı, IAracRepository ve IMusteriRepository arayüzlerine bağımlıdır.
* **Gerçekleştirme İlişkisi (Realization)**: Bir sınıfın bir arayüzü uygulaması durumunda oluşur.
  + AracRepository sınıfı, IAracRepository arayüzünü gerçekleştirir.
  + MusteriRepository sınıfı, IMusteriRepository arayüzünü gerçekleştirir.
  + KiralamaServisi sınıfı, IKiralama arayüzünü gerçekleştirir.
* **SQL Server:** Veri depolama ve yönetimi için SQL Server kullanılmıştır. Veritabanı bağlantısı, ConfigurationManager aracılığıyla sağlanır. Bağlantı dizesi, uygulamanın yapılandırma dosyasında tanımlanmıştır.

Örnek: \_connectionString = ConfigurationManager.ConnectionStrings["AracKiralamaDB"].ConnectionString;

**Örnek Kod:**

Aşağıda, AracRepository sınıfında bir aracın eklenmesi işlemi örneklenmiştir:

public void AracEkle(Arac arac)

{

using (SqlConnection connection = new SqlConnection(\_connectionString))

{

string query = "INSERT INTO Araclar (Marka, Model, Yil, Fiyat, KiradaMi) VALUES (@Marka, @Model, @Yil, @Fiyat, @KiradaMi)";

SqlCommand command = new SqlCommand(query, connection);

command.Parameters.AddWithValue("@Marka", arac.Marka);

command.Parameters.AddWithValue("@Model", arac.Model);

command.Parameters.AddWithValue("@Yil", arac.Yil);

command.Parameters.AddWithValue("@Fiyat", arac.Fiyat);

command.Parameters.AddWithValue("@KiradaMi", arac.KiradaMi);

connection.Open();

command.ExecuteNonQuery();

}

}

Örneğin, bağlantı dizesi şu şekilde olabilir:

<connectionStrings>

<add name="AracKiralamaDB" connectionString="Data Source=DESKTOP-K77RIQS;Initial Catalog=AracKiralamaDB;Integrated Security=True" />

</connectionStrings>

**Sonuç:**

Bu doküman, Araç Kiralama Uygulaması'nın genel yapısını, kullanılan teknolojileri ve karşılaşılan olası sorunları özetlemektedir. Uygulama, kullanıcıların araç kiralama işlemlerini kolaylaştırmak amacıyla tasarlanmış olup, OOP prensiplerine ve modern yazılım geliştirme tekniklerine uygun olarak geliştirilmiştir.