YILDIZ TEKNİK ÜNİVERSİTESİ
ELEKTRİK-ELEKTRONİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Yazılım Mühendisliği Dersi Dönem Projesi

Motorlu Araç Filo Yönetim Sistemi

Prof. Dr. Oya KALIPSIZ

Grup Üyeleri:
21011007 Elif Gülerarslan
22011602 Esad Engin Işık
21011092 Bilal Oğuz Erdoğdu
Ç21069012 Ceyda Babuz

İçindekiler

*Proje kodu sisteme kütüphaneler dahil olmayarak yüklendi online yildiz yüklemesinin boyut sınırı aşılıyor kütüphaneler dahil edildiğinde

1. Proje Planı

1.1. Proje Alan Tanımı

Bu proje, farklı şirketlere filo hizmeti sağlayan bir firmanın filo yönetimini dijital ortamda takip edebilmesini sağlayan bir yazılım sistemini oluşturmayı amaçlar. Şirketler, araç filosu yönetimini daha verimli bir şekilde gerçekleştirebilmek için bu tür sistemlere ihtiyaç duymaktadır. Motorlu Araç Filo Yönetim Sistemi de firmalara kiralanan araçların bakım, kullanım, yakıt harcama gibi süreçlerini takip edecek bir bilgi sistemi olarak dijital ortamda iş süreçlerinin izlenebilirliğini sağlar. Aynı zamanda sisteme girilen verilerden faydalanılarak raporlama, geçmiş verilerden faydalanarak maliyet tahmini gibi analizleri yapmaya olanak sağlar.
Sistem aşağıdaki temel işlevleri kapsamaktadır:

1. Araç Yönetimi
2. Bakım ve Servis Takibi
3. Kilometre takibi
4. Harcama Yönetimi
5. Araç- Çalışan atamaları
6. Raporlama
7. Maliyet Tahmini

1.2. Fizibilite

Teknik Fizibilite

Projenin geliştirilmesi için web, mobil veya masaüstü uygulama gibi çeşitli platform seçenekleri bulunmaktadır. Programlama dili olarak JavaScript, HTML/CSS, Java, C#, Python gibi alternatifler değerlendirilmiştir. Bu seçenekler arasından, teknik ekip tarafından hem performans hem de kullanım kolaylığı açısından daha uygun bulunan nesneye dayalı olan Java programlama dili tercih edilmiştir. Geliştirme süreci Visual Studio Code üzerinde yürütülecektir.

Donanım Fizibilitesi

Donanım fizibilitesi kapsamında, projede kullanılacak bilgisayarın özellikleri yeterli işlem gücünü sağlamaktadır. AMD Ryzen 5 5600 6-Core İşlemci ve 16,0 GB RAM donanımları kullanılacaktır. Aynı zamanda güçlü bir GPU olan RTX 4060 (128-bit, 8GB VRAM) kullanılmaktadır.

Ekonomik Fizibilite

Projenin ekonomik fizibilitesi göz önünde bulundurularak, maliyetleri en aza indirmek amacıyla açık kaynaklı ve ücretsiz yazılım araçları ile kütüphaneler kullanılacaktır.

Personel Giderleri	Ücretsiz
Yazılım Araçları	Ücretsiz
Bilgisayar	20000

Yasal Fizibilite

Sistem, 6698 sayılı Kişisel Verilerin Korunması Kanunu (KVKK) ile tam uyumlu olarak tasarlanacaktır.Çalışanlara ait bilgiler (sürücü bilgileri, araç kullanım verileri) kanun kapsamında işlenecek ve korunacaktır.Açık rıza beyanları sisteme entegre edilecek ve gerekli aydınlatma metinleri hazırlanacaktır.213 sayılı Vergi Usul Kanunu kapsamında araç harcamalarına ait finansal kayıtlar elektronik ortamda tutulacaktır.Bakım, tamir ve diğer hizmetlere ilişkin faturaların elektronik ortamda saklanması, ilgili mevzuata uygun olarak gerçekleştirilecektir. Veri güvenliği kapsamında ise ISO 27001 Bilgi Güvenliği Yönetim Sistemi standartlarına uygun veri güvenliği önlemleri alınacaktır.Kullanıcı erişim yetkilendirmeleri ve veri şifreleme uygulamaları mevzuata uygun şekilde yapılandırılacaktır.

1.3. Kabul ve Kısıtlar

Kabuller

1. Dış kaynak firmaları ve kullanıcılar sisteme yapacakları tüm veri girişlerini eksiksiz, doğru ve zamanında gerçekleştirecektir.
2. Sistemi kullanacak tüm kullanıcılar (firma yetkilileri, dış kaynak firması çalışanları, sürücüler vb.) temel bilgisayar kullanım bilgisine sahiptir.
3. Sistemin sorunsuz çalışması için gerekli altyapı sağlanmıştır.
4. Sistemde yer alacak araçlara ait bilgiler (model, plaka, kilometre, bakım geçmişi vb.) ilgili kullanıcılar/paydaşlar tarafından eksiksiz ve doğru bir şekilde sağlanacaktır.
5. Uygulama masaüstü tabanlı çalışacak şekilde tasarlanacaktır.
6. Sistemdeki kullanıcı rolleri ve yetkilendirme düzeyleri aşağıda belirtildiği şekilde belirlenmiştir.
7. Harcama ve bakım gibi işlemlerin sisteme kaydı, yalnızca dış kaynak firması veya yetkili kişiler tarafından onaylandıktan sonra yapılır.
8. Kullanıcılar sistemi aktif olarak kullanacaktır.
9. Harcama tahmini için hareketli ortalama yöntemi kullanılacaktır.
10. Harcamalar (kasko, bakım, yakıt, vb.) doğru kategorilere ayrılarak sisteme girildiği kabul edilir.
11. Tüm harcamalar için para birimi olarak yalnızca TL kullanılacağı varsayılmıştır.
12. Firma ve dış kaynak kullanıcıları sadece yetkili oldukları verilere erişebilir.

Kullanıcı Yetkileri

Bu bilgi sistemini kullanan farklı yetkilere sahip 3 tipte sistem kullanıcı grupları vardır. Bunlar,

1. Firma yetkilileri : Filo hizmetini firma içinde yöneten yetkili
2. Dış Kaynak Firması Yetkilileri: Kiralanan araçların bakımı, sigortası ve harcama yönetimiyle ilgilenen Araç sizde Yükü bizde A.Ş. firması
3. Şirket Çalışanları: Araçları belirli bir süre kullanmakla görevlendirilen şirket çalışanlarıdır.

Firma yetkilileri, şirket içinde kiralanan araçlar için çalışan atama ve rapor görüntüleme işlemlerini yaparlar.

Dış kaynak firması yetkilileri ise araç bakım, tamir ve yakıt harcamalarını sisteme girme, çalışanların girdiği kilometre bilgilerini doğrulama gibi işlemlerin yanında harcamalar için raporlar oluşturma ve görüntüleme yetkilerine sahiptir.

Son olarak şirket çalışanları bu sistemi kendilerine atanan araç bilgilerini görüntüleme, kullanım süresi boyunca yaptığı kilometre bilgisini ve yakıt harcamalarını sisteme girme amacıyla kullanırlar.

Kısıtlar

1. Proje, belirlenen teslim zamanına kadar tamamlanmak zorundadır. Ek geliştirme süresi yoktur.
2. Proje yalnızca 4 kişilik bir ekiple tamamlanacaktır.
3. Uygulama, kişisel bilgisayarlar ile sınırlı donanım ve altyapı kaynakları ile tasarlanacaktır.
4. Sisteme mobil uygulama yoluyla erişim sağlanamaz.
5. Veri girişleri manuel olarak olarak elle yapılacaktır.
6. Sistem yalnızca basit raporlamalar ve geçmiş veriler üzerinden analizler sunacaktır.
7. Gerçek zamanlı analiz yapılmayacaktır.
8. Arayüz dili yalnızca Türkçe olacak; sistem çoklu dil desteğine sahip olmayacaktır.
9. Dış kaynak firmalarının işlem süreçleri sistem geliştirme sırasında değişmeyecek kabul edilir. Süreç değişiklikleri sistemde anlık uyarlama gerektirmez.

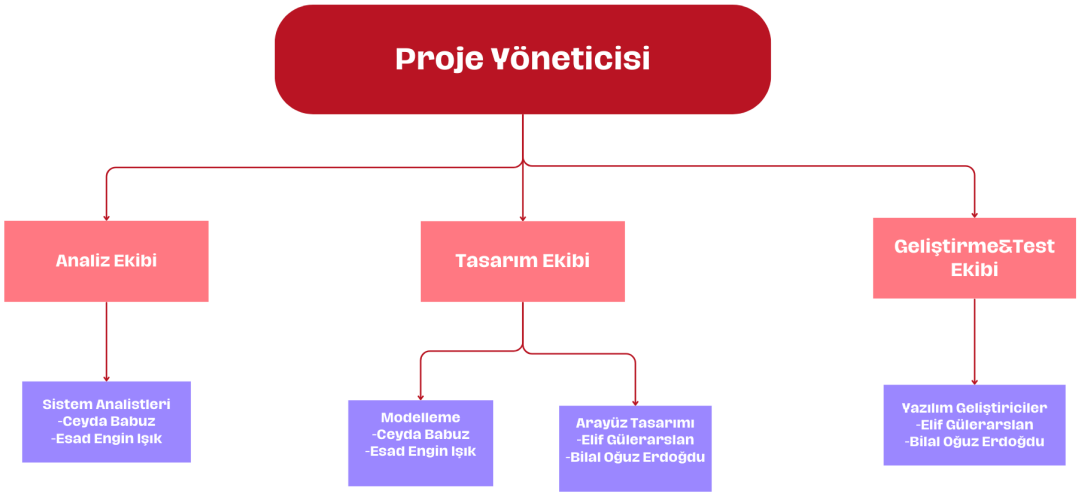
10. Kullanıcılar yalnızca önceden tanımlanmış rapor türlerini alabilir. Özgün rapor tasarımı desteği sunulmamaktadır.

1.4. Proje İş-Zaman Çizelgesi (Gantt Diyagramı)

Gantt şeması, bir projede belirli görevlerin ve kaynakların planlanmasını, yönetilmesini ve izlenmesini kolaylaştıran önemli bir görsel araçtır. Bu projede de proje sürecinin daha kolay bir şekilde takip ve kontrol edilebilmesi için proje zaman çizelgesi olarak Gantt şeması hazırlandı. Öncelikle proje için tamamlanması gereken iş paketleri belirlendi. Temelde analiz, tasarım, geliştirme ve test başlıkları altında toplanan iş tanımları ve alt başlıkları, projenin o aşamasından sorumlu kişiler, her bir aktivitenin başlangıç ve bitiş zamanları gösterildi. Aynı zamanda her aşamanın tamamlanma yüzdesi ve hafta bazındaki zaman öncelik durumları aşağıda görüldüğü üzere çubuk grafikler yardımıyla görselleştirilerek proje zaman çizelgesinin son hali oluşturuldu.

No	Job Description		Start Date	Due Date	Complete percent	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7
1	**1.Analiz**											
	1.1	Fizibilite Çalışması	22.03.2025	5.04.2025	*100%*							
	1.2	Proje Gereksinimlerinin belirlenmesi	29.03.2025	5.04.2025	*100%*							
	1.3	Proje zaman çizelgesinin oluşturulması	29.03.2025	5.04.2025	*100%*							
	1.4	Organizasyon yapısının oluşturulması	29.03.2025	5.04.2025	*100%*							
	1.5	Kullanım senaryolarının oluşturulması	29.03.2025	5.04.2025	*100%*							
	1.6	Analiz Bitimi Toplantı	5.04.2025	12.04.2025	*100%*							
2	**2.Tasarım**											
	2.1	İzlenebilirlik tablosunun oluşturulması	12.04.2025	19.04.2025	100%							
	2.2	Ön tasarım	12.04.2025	19.04.2025	100%							
	2.3	UML diyagramının çizilmesi	12.04.2025	19.04.2025	100%							
	2.4	Ardışıl diyagramın oluşturulması	19.04.2025	26.04.2025	100%							
	2.5	Tasarım Bitimi Toplantı	26.04.2025	26.04.2025	100%							
3	**3.Geliştirme&Test**											
	3.1	Veritabanı oluşturma	12.04.2025	26.04.2025	100%							
	3.2	Ön yüz geliştirme	12.04.2025	26.04.2025	100%							
	3.3	Modül Kodlama	12.04.2025	26.04.2025	100%							
	3.4	Veritabanı bağlantısı	19.04.2025	26.04.2025	100%							
	3.5	Birim Test	26.04.2025	3.05.2025	100%							
	3.6	Sistem Testi	26.04.2025	3.05.2025	100%							
	3.7	Kullanıcı Testleri	26.04.2025	10.05.2025	100%							
	3.8	Geliştirme ve test aşaması final toplantısı	3.05.2025	10.05.2025	100%							
4	**4.Proje teslimi**											
	4.1	Proje Raporunun tamamlanması	3.05.2025	16.05.2025	100%							
	4.2	Proje Teslim Toplantısı	10.05.2025	16.05.2025	100%							
	4.3	Proje sunumu	10.05.2025	16.05.2025	0%							

1.5. Ekip Organizasyon Şeması ve Görev Dağılımları



Proje 4 kişilik bir ekip tarafından yürütülmüştür. Ekip, iş bölümü esas alınarak analiz, tasarım ve geliştirme & test olmak üzere üç temel gruba ayrılmıştır.
Projenin başlangıç aşamasında toplantı yapılarak proje planı oluşturulmuş, proje kapsamı ve gereksinimler netleştirilmiştir. Ardından ekip üyelerinin yetkinliklerine göre görev dağılımları yapılmıştır.

Analiz ekibi, 2 kişiden oluşmaktadır. Bu ekip, sistemin işleyişine yönelik ihtiyaç analizini yapmış ve gereksinimleri sistem analizi düzeyinde belgelemiştir. Ayrıca kullanım senaryolarına uygun UML diyagramlarının temellerini oluşturmuştur.

Tasarım aşaması 2 bölümde ele alınmıştır. Bunlar modelleme ve arayüz tasarımıdır. Modelleme kısmında görevli kişiler sistemin sınıf yapısını, ilişkilerini modellemiştir. Gerekli etkinlik, durum diyagramları oluşturulmuştur. Arayüz tasarımı kısmında ise tasarlanan model detaylandırılarak modele uyumlu olacak şekilde kullanıcı arayüzünün işlevsel tasarımı, menü yapısı ve etkileşim akışları belirlenmiştir.

Geliştirme ve test ekibi sistemin Java dilinde kodlanmasından, gerekli fonksiyonların gerçeklenmesinden ve test edilmesinden sorumludur. Arka plan sınıf yapıları, ekran işlemleri ve giriş/kayıt mekanizmaları geliştirilmiş; proje sonunda birim testlerle doğrulanmıştır.

1.6. Risk Tablosu

Risk ID	Risk tanımı	Olasılık	Etki	Muhtemel Nedenler	Azaltma Stratejisi
1	Zamanında teslim edilememe	Orta	Yüksek	Planlama ve takip eksikliği	Haftalık ilerleme toplantıları ve Gantt şeması kullanarak ilerlemenin takibi
2	Gereksinimlerin yanlış yorumlanması	Orta	Yüksek	Gereksinimlerin net tanımlanmamış olması	Net olmayan tanımlamalarla ilgili ilgili kişiye danışılması
3	Yazılım performans sorunları	Düşük	Orta	Yüksek veri hacmi, sistemin yoğun kullanımda yavaşlaması	Performans testi yapılmalı, sistem optimize edilmeli
4	Proje kapsamının genişlemesi	Düşük	Orta	Müşteri tarafından yeni taleplerin gelmesi	Kapsam dışı istekler kabul edilmemeli, sözleşme buna uygun yapılmalı
5	Hatalı veri girişi yapılması	Orta	Yüksek	Kullanıcı hataları	Veri doğrulama ve hata kontrol sistemleri eklenmeli
6	Yazılımsal hatalar	Düşük	Yüksek	Sistem testi eksikleri, yanlış kodlama	Tüm fonksiyonlar test edilmeli
7	Kullanıcı eğitim eksiklikleri	Orta	Orta	Kullanıcıların sistemin tüm fonksiyonlarını öğrenememesi	Kullanıcı eğitimleri planlanmalı
8	Teknik/Altyapı Problemleri	Orta	Orta	Yazılımın mevcut donanımla uyumsuz olması, eski yazılım teknolojileri	Donanım uyumluluğunun proje öncesinde kontrol edilmesi
9	Veri güvenliği ihlali	Düşük	Yüksek	Güvenlik önlemlerinin yetersiz olması	Kullanıcı erişim yetkileri sınırlanmalı
10	Sistem entegrasyon sorunları	Düşük	Orta	Sistem ve diğer yazılımlar arasında entegrasyon problemleri	Ortak sınıf yapısı ve veri modelleri önceden netleştirilir.

Bu projede, yazılım geliştirme sürecinde karşılaşılması muhtemel riskler önceden belirlenmiş ve bunlar bir tablo hâlinde sistematik olarak analiz edilmiştir. Riskler; olası etkileri, oluşma ihtimalleri, muhtemel nedenleri ve azaltma stratejileri ile birlikte değerlendirilmiştir.
Tablo toplamda 10 temel riski içermektedir ve proje boyunca teknik, organizasyonel ve kullanıcı kaynaklı aksaklıkları kapsayacak şekilde hazırlanmıştır.

Risk olasılık ve etki dereceleri aşağıdaki gibi ölçeklendirilmiştir.

Olasılık dereceleri
Yüksek
Orta
Düşük

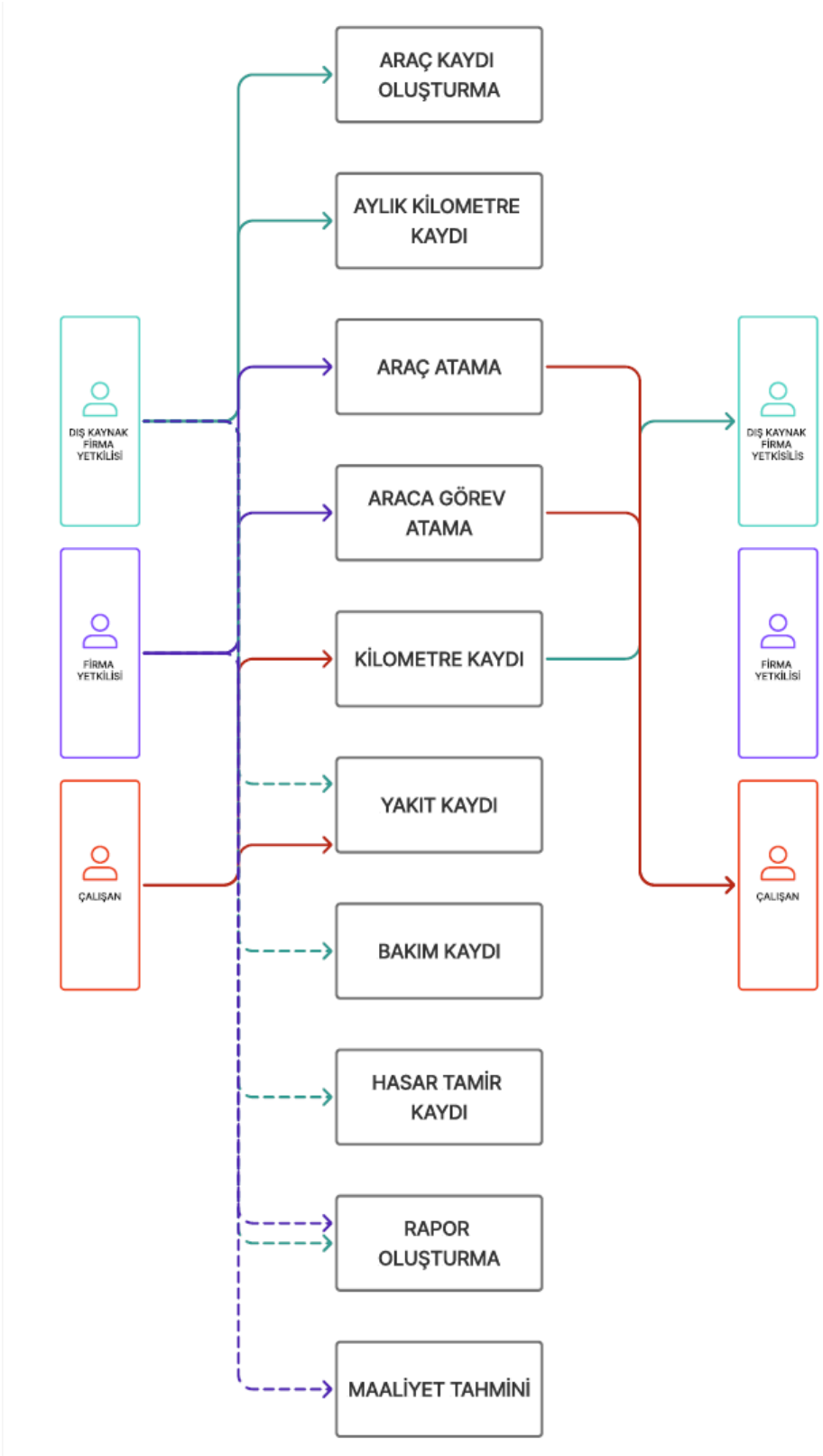
Etki Dereceleri
Büyük
Orta
Küçük

En kritik riskler, “Zamanında teslim edilememe”, “Gereksinimlerin yanlış yorumlanması” ve “Hatalı veri girişi yapılması” olarak öne çıkmaktadır. Bu risklerin hem olasılığı hem de etkisi orta–yüksek düzeyde olduğundan, azaltma stratejileri detaylı biçimde belirlenmiştir.
Düşük olasılıklı ancak yüksek etkili riskler arasında “Yazılımsal hatalar” ve “Veri güvenliği ihlali” yer almaktadır. Bu risklerin nadiren gerçekleşmesi beklenmektedir, ancak gerçekleştiğinde proje çıktıları üzerinde ciddi etkiler oluşturabileceği için gerekli önlemler alınmıştır.
“Performans sorunları”, “Proje kapsamının genişlemesi”, “Teknik altyapı problemleri” gibi bazı riskler ise düşük ya da orta etkili olmasına rağmen takip gerektiren durumlardır. Bu riskler için optimizasyon, kapsam sınırlama ve donanım uyumluluğu kontrolleri gibi önleyici stratejiler uygulanacaktır.

Kullanıcı eğitimi eksikliği de proje başarısını etkileyebilecek bir unsurdur. Bu risk için sistemin kullanıcı dostu tasarlanması ve kullanım kılavuzlarının oluşturulması planlanmıştır.

2. İsteklerin Modellenmesi

2.1. Kullanım Senaryosu Diyagramı



2.2. Kullanım Senaryosu Metinleri

1. Araç Kaydı Oluşturma

Aktör: Dış Kaynak Firması Yetkilisi

Amaç: Filoya yeni bir aracın kaydedilmesi

Ana Akış:

1. Yetkili, "Yeni Araç Ekle" bölümüne girer
2. Araç bilgilerini (plaka, marka, model, üretim yılı) girer
3. Aracın tipini belirler (kiralık/özmal)
4. Kiralık araç ise kira başlangıç ve bitiş tarihlerini, aylık kira bedelini girer
5. Başlangıç kilometre bilgisini girer
6. Araç kaydını tamamlar
7. Sistem araca benzersiz bir ID atar ve kayıt tamamlanır

2. Aylık Kilometre Kaydı

Aktör: Dış Kaynak Firması Yetkilisi

Amaç: Her ay başında araçların güncel kilometre bilgilerinin güncellenmesi

Ana Akış:

1. Yetkili "Kilometre Kayıt" ekranına girer
2. Filodaki araçların listesini görür
3. Her bir araç için güncel kilometre bilgisini girer
4. Sistem önceki ayın kilometre değerini otomatik olarak saklar
5. Sistem aylık kat edilen mesafeyi hesaplar
6. Yetkili tüm kayıtları onaylar

3. Araç-Çalışan Atama

Aktör: Firma Yetkilisi

Amaç: Bir aracın belirli bir çalışana atanması

Ana Akış:

1. Yetkili "Araç Atama" ekranına girer
2. Atanabilir araçlar listesinden bir araç seçer
3. Çalışan listesinden bir çalışan seçer
4. Atama başlangıç tarihini belirler
5. Atama bitiş tarihi (varsa) belirler
6. Atama nedenini girer
7. Atamayı onaylar
8. Sistem atama kaydını oluşturur ve ilgili taraflara bildirim gönderir

4. Havuz Aracı Görev Atama

Aktör: Firma Yetkilisi

Amaç: Havuzda bulunan bir araca görev ve sürücü atanması

Ana Akış:

1. Yetkili "Havuz Araç Yönetimi" ekranına girer
2. Havuzdaki müsait araçları görüntüler
3. Bir araç seçer
4. Görevi tanımlar (görev adı, türü, beklenen süre)
5. Sürücü olarak atanacak çalışanı seçer
6. Görevin başlangıç ve tahmini bitiş tarihini belirler
7. Atamayı kaydeder
8. Sistem ilgili çalışana bildirim gönderir

5. Çalışan Kilometre Kaydı

Aktör: Çalışan

Amaç: Çalışanın kullandığı araçla kat ettiği mesafenin kaydedilmesi

Ana Akış:

1. Çalışan sisteme giriş yapar
2. "Kilometre Bildirimi" bölümüne girer
3. Kendisine atanmış aracı görür
4. Seyahat bilgilerini girer (tarih, başlangıç km, bitiş km, rota)
5. Bilgileri onaylar ve gönderir
6. Sistem kaydı dış kaynak firmasına bildirir

6. Kilometre Doğrulama

Aktör: Dış Kaynak Firması Yetkilisi

Amaç: Çalışanın bildirdiği kilometre bilgilerinin doğrulanması

Ana Akış:

1. Yetkili "Kilometre Onay" ekranına girer
2. Bekleyen bildirimleri görüntüler
3. Bildirimi inceler
4. Doğruysa onaylar
5. Sistem onaylanan kilometre bilgisini kaydeder Alternatif Akış: 4a. Yetkili bilgilerde tutarsızlık tespit eder 4b. Düzeltme talebiyle çalışana geri gönderir

7. Bakım Kaydı

Aktör: Dış Kaynak Firması Yetkilisi

Amaç: Araç için yapılan bakım işleminin kaydedilmesi

Ana Akış:

1. Yetkili "Bakım Yönetimi" ekranına girer
2. İlgili aracı seçer
3. Bakım tipini belirler (periyodik bakım, lastik değişimi, vs.)
4. Bakımı yapan servis bilgilerini girer
5. Bakım tarihini ve detaylarını girer
6. Maliyet bilgilerini girer
7. Değişen parça bilgilerini kaydeder (kasko kapsamında olup olmadığını belirtir)
8. Bakım kaydını tamamlar

8. Yakıt Harcaması Kaydı

Aktör: Çalışan/Dış Kaynak Firması Yetkilisi

Amaç: Yakıt alımlarının kaydedilmesi

Ana Akış:

1. Kullanıcı "Yakıt Girişi" ekranına girer
2. İlgili aracı seçer
3. Yakıt alım tarihini girer
4. Alınan yakıt miktarını ve tutarını girer
5. Kilometre bilgisini girer
6. Yakıt fişi/fatura bilgilerini girer
7. Kaydı tamamlar
8. Sistem ortalama yakıt tüketimini hesaplar

9. Hasar/Tamir Kaydı

Aktör: Dış Kaynak Firması Yetkilisi

Amaç: Araçta oluşan hasarın ve tamirin kaydedilmesi

Ana Akış:

1. Yetkili "Hasar/Tamir Yönetimi" ekranına girer
2. İlgili aracı seçer
3. Hasar tarihini, tipini ve detaylarını girer
4. Hasarın kasko kapsamında olup olmadığını belirler
5. Tamir bilgilerini girer (servis, tarih, maliyet)
6. Değişen parça bilgilerini kaydeder

10. Harcama Raporu Oluşturma

Aktör: Firma Yetkilisi/Dış Kaynak Firması Yetkilisi

Amaç: Araç harcamalarının raporlanması

Ana Akış:

1. Yetkili "Raporlama" ekranına girer
2. Rapor tipini seçer (bakım, kasko, yakıt, tamir, genel toplam)
3. Rapor parametrelerini belirler (tarih aralığı, araç/araç grubu)
4. Rapor formatını seçer (tablo ve/veya grafik)
5. Raporu oluştur butonuna basar
6. Sistem raporu oluşturur ve gösterir
7. Yetkili raporu PDF, Excel vb. formatlarda dışa aktarabilir.

11. Maliyet Tahmini Oluşturma

Aktör: Firma Yetkilisi

Amaç: Gelecek dönem için maliyet tahmini yapılması

Ana Akış:

1. Yetkili "Maliyet Tahmin" ekranına girer
2. Tahmin edilecek harcama kategorisini seçer
3. Tahmin yöntemini seçer (Hareketli Ortalama)
4. Hesaplamada kullanılacak geçmiş dönem sayısını belirler
5. Tahmin butonuna basar
6. Sistem Hareketli Ortalama Yöntemi ile tahmini hesaplar ve gösterir
7. Yetkili sonuçları tablo ve grafik olarak görüntüler

2.3. İzlenebilirlik Matrisi

İzlenebilirlik matrisi, bir yazılım projesinde tanımlanan sistem gereksinimlerinin, bu gereksinimleri karşılayan kullanım senaryoları, testler veya sistem bileşenleri ile olan bağlantısını gösteren tablosal bir modeldir.

Amaç, her bir gereksinimin proje içinde nasıl ve nerede karşılandığını izlenebilir ve takip edilebilir hâle getirmektir.

Temel Sistem Gereksinimleri	KS-1: Araç Kaydı Oluşturma	KS-2: Aylık Kilometre Kaydı	KS-3: Araç-Çalışan Atama	KS-4: Havuz Aracı Görev Atama	KS-5: Çalışan Kilometre Kaydı	KS-6: Kilometre Doğrulama	KS-7: Bakım Kaydı	KS-8: Yakıt Harcaması Kaydı	KS-9: Hasar/ Tamir Kaydı	KS-10: Harcama Raporu Oluşturma	KS-11: Maliyet Tahmini Oluşturma
Araç bilgilerinin kaydedilmesi	X										
Araç tipinin belirlenmesi	X										
Kilometre takibi	X	X			X	X		X		X	X
Aylık kilometre hesaplama		X								X	X
Araç-çalışan ilişkilendirme			X	X							
Atama süresi belirleme			X	X							
Görev tanımlama				X							
Seyahat bilgisi kaydetme					X						
Kilometre doğrulama						X					
Bakım işlemlerinin kaydı							X			X	X
Maliyet bilgilerinin kaydı							X	X	X	X	X
Yakıt tüketimi hesaplama								X		X	X
Hasar/tamir kaydı									X	X	X
Rapor oluşturma										X	
Maliyet tahmini yapma											X

Bu tabloda, sistemin gerçekleştirmesi gereken temel işlevler ile bu işlevleri karşılayan kullanım senaryoları (KS-1 ila KS-11) arasında kurulan bağlantılar gösterilmektedir. Bu yapı, projenin gereksinimlerinin hangi senaryolarla doğrudan karşılandığını analiz edebilmek için oluşturulmuştur. Aynı zamanda yazılımın gereksinimlere olan uygunluğunu izlemeye olanak sağlar.

Satırlarda, sistemin karşılaması gereken temel işlevsel gereksinimler yer almaktadır. Bunlar, örneğin “Araç bilgilerinin kaydedilmesi”, “Kilometre takibi”, “Görev tanımlama”, “Rapor oluşturma” gibi modül ya da alt işlev düzeyindeki gereksinimlerdir.

Sütunlarda, önceden belirlenmiş olan 11 adet temel kullanım senaryosu (KS-1: Araç Kaydı Oluşturma, KS-2: Aylık Kilometre Kaydı, ... KS-11: Maliyet Tahmini Oluşturma) yer almaktadır.

Her bir hücrede yer alan “X” işareti, ilgili gereksinimin belirtilen senaryo kapsamında doğrudan ele alındığını veya gerçekleştirildiğini ifade etmektedir.

3. Nesneye Dayalı Modelleme

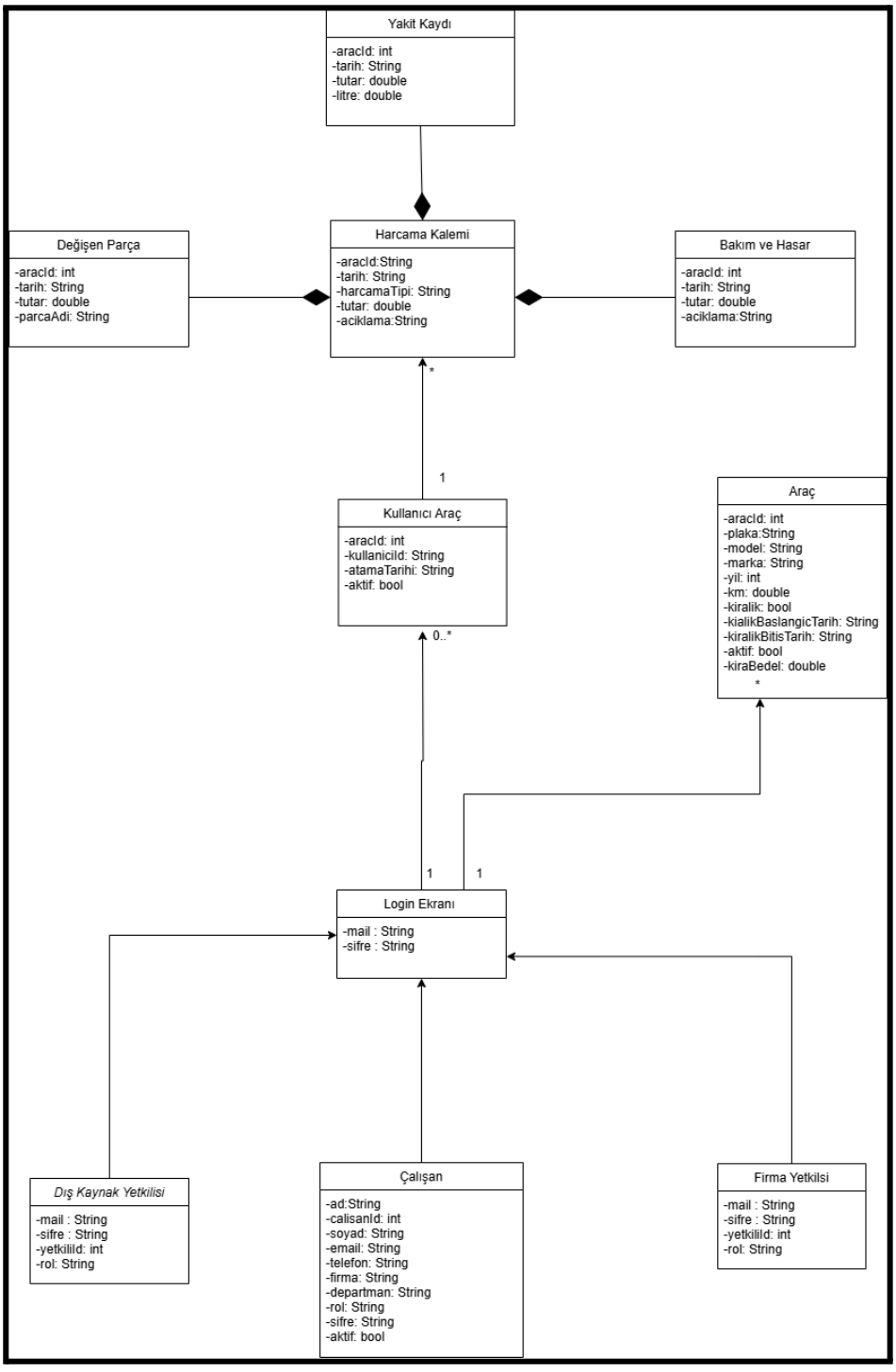
3.1. Sınıf Diyagramı

Firma Yetkilisi: Araç kiralayan firmanın yetkilisi. Rapor ve tahmin görüntüleme yapar.

Dış Kaynak Yetkilisi: Filo Yönetim Sisteminin bir yöneticisidir.

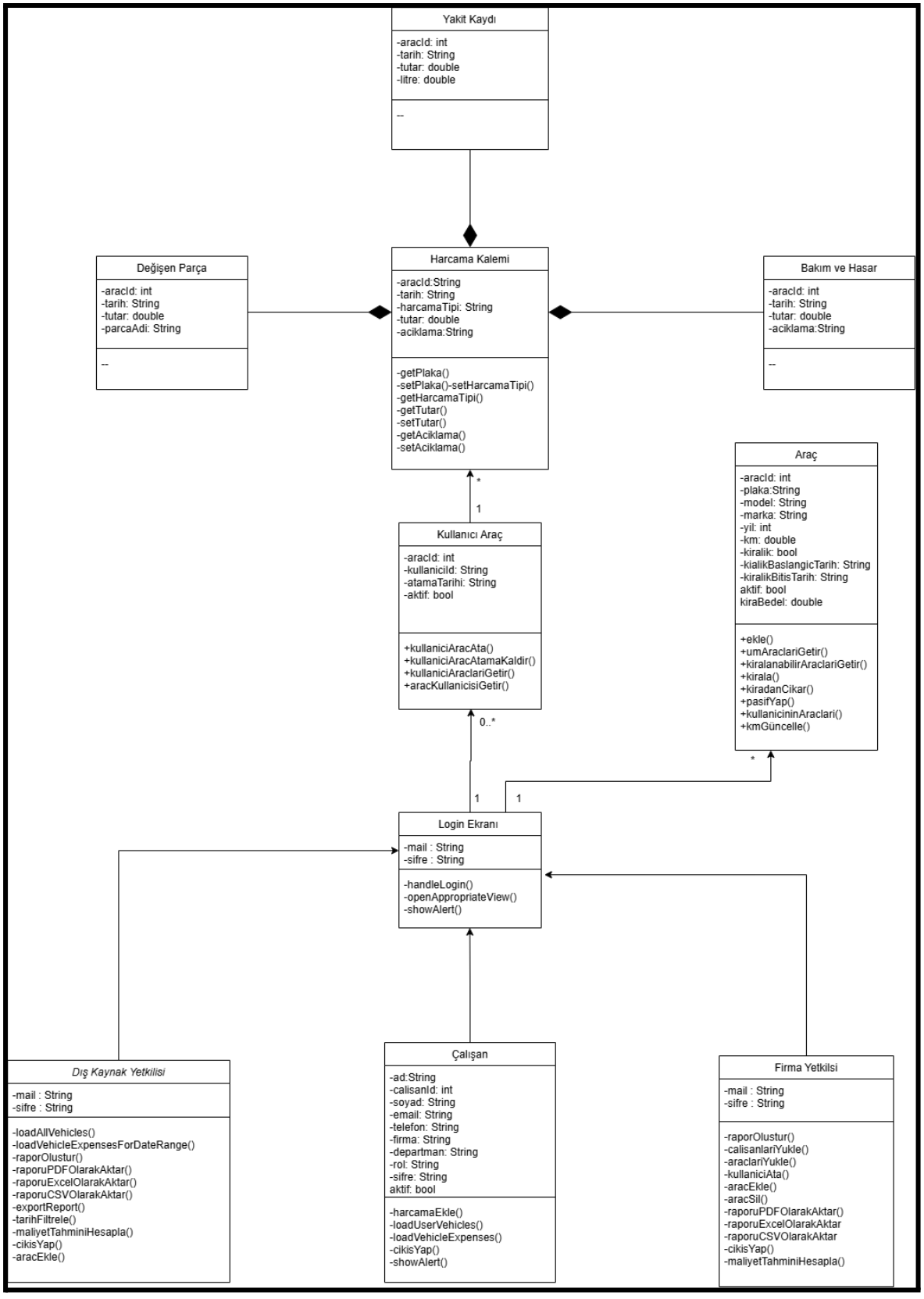
Kullanıcı Araç Geçmişi: Bir çalışana atanmış arabayı temsil eder

Çalışan: İlgili aracın km (varsa hasar) bildirimini yapar.



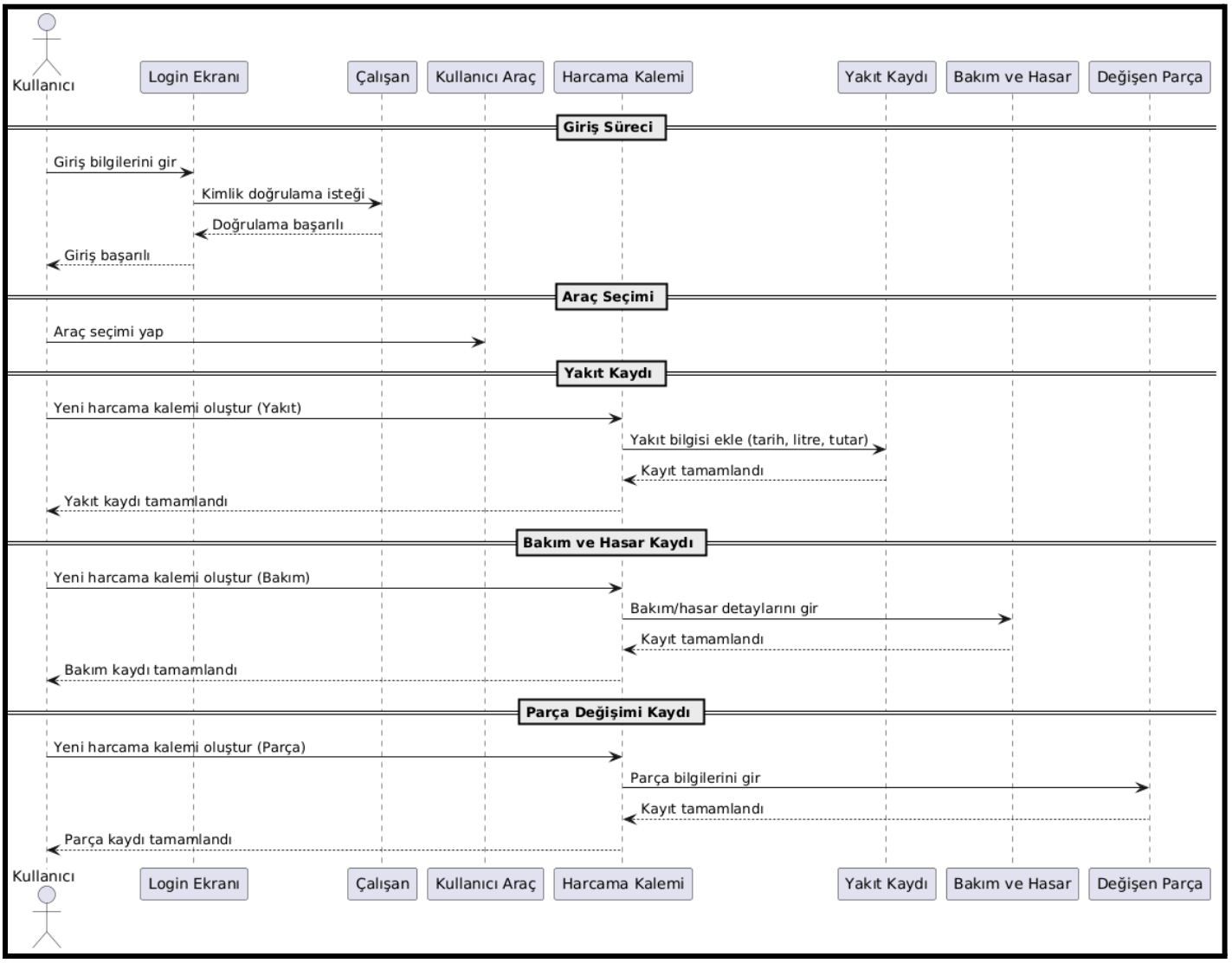
4. Nesneye Dayalı Tasarım

4.1. Tasarım Sınıf Diyagramı



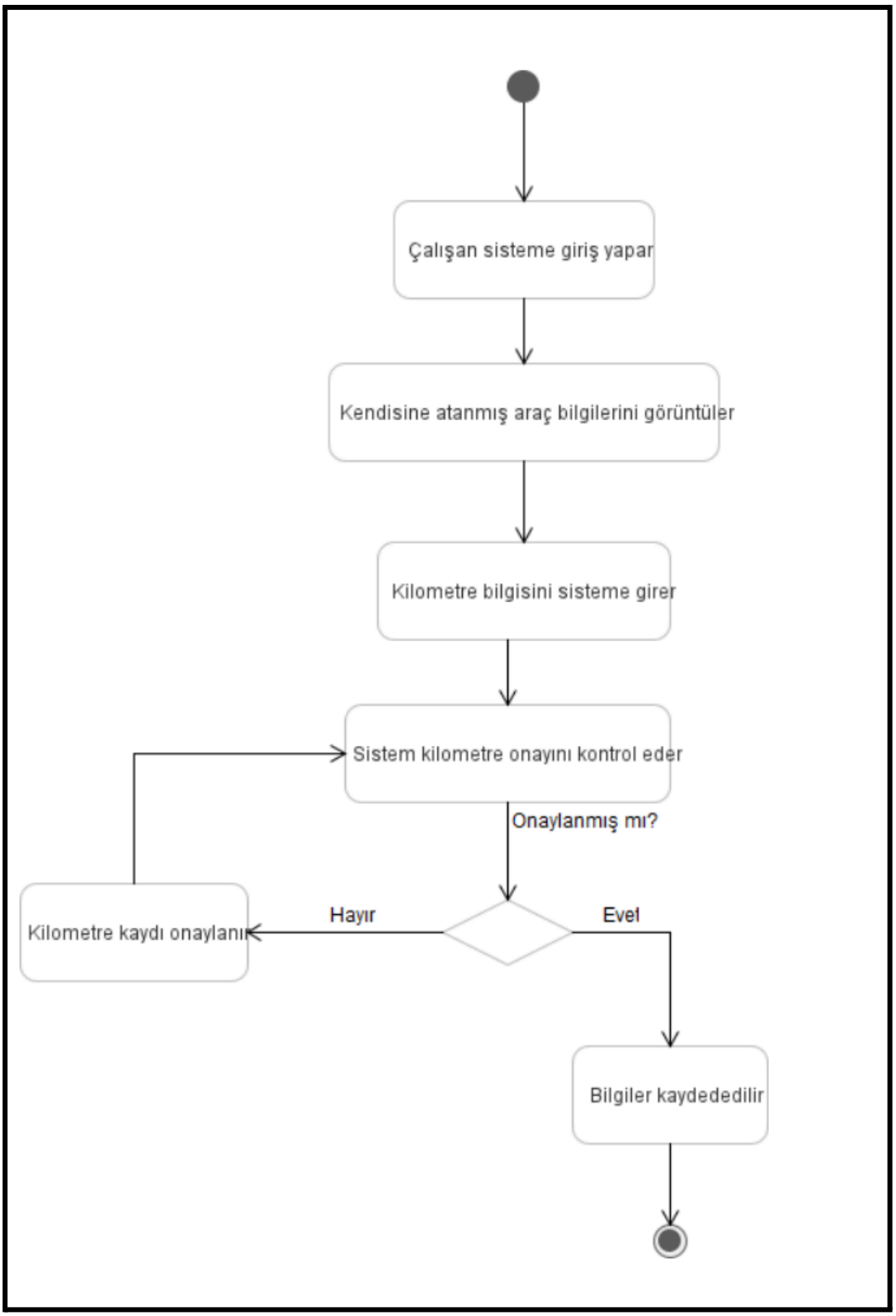
4.2. Sıralama (Sequence) Diyagramı

Aşağıda Çalışanın üzerine atanan aracı kullanarak gerekli bildirimleri yapma süreci şekillendirilmiştir.

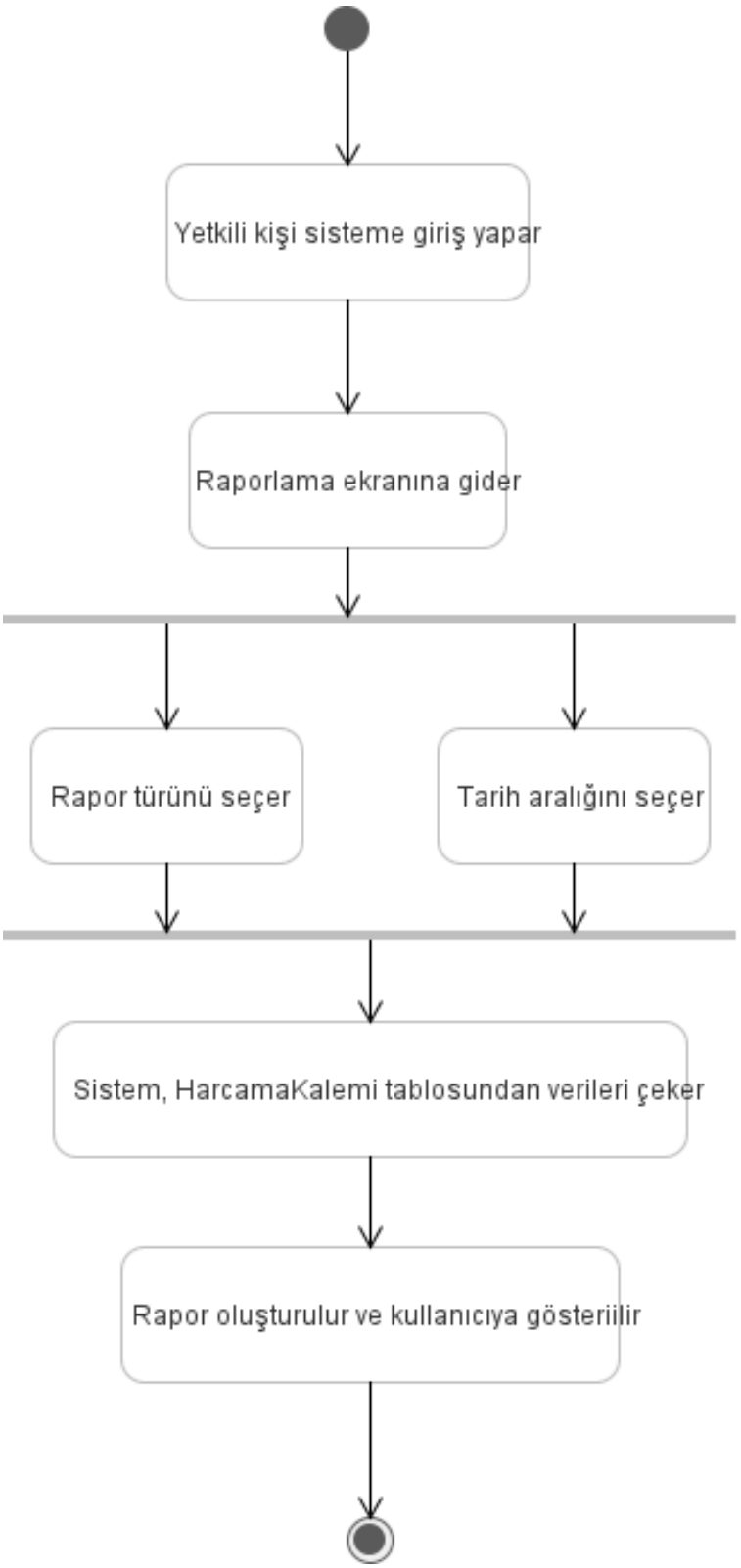


4.3. Etkinlik (Activity) Diyagramı

Activity Diyagram 1: Çalışanın atanmış aracı kullanarak kilometre bildirimi yapma süreci



Activity Diyagram 2: Firma yetkilisinin rapor oluşturma süreci



4.4. Durum (State) Diyagramı

Aşağıdaki durum diyagramı, bir aracın filo yönetim sistemi içerisinde çalışana atanması ve kullanım sürecindeki durum değişimlerini gösterir.

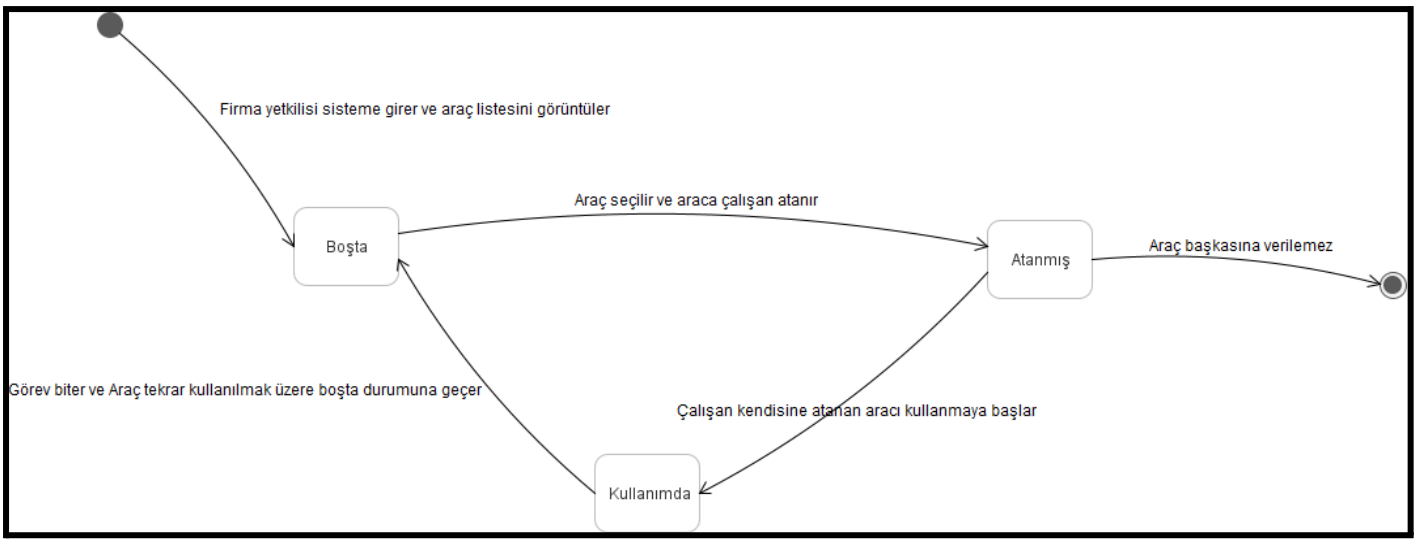
Araçlar için 3 farklı durum bulunmaktadır. Bunlar boşta, atanmış ve kullanımda durumlarıdır.

Atama işlemi boşta durumundaki araçlar için yapılır. Firma yetkilisi sisteme giriş yapar ve araç listesini görüntüler. Burada henüz bir çalışana atanmamış boşta olan araçlardan birini seçer ve araca çalışan atar. Bu işlem sonrasında araç "atanmış" durumuna geçer. Atanmış durumundaki araç belirli bir çalışan için ayrılmıştır ve başka bir çalışana verilemez.

Atanmış araç, çalışan tarafından teslim alınıp kullanılmaya başlandığında aracın durumu "kullanımda" durumuna geçer.

Görev süresi tamamlandığında araç tekrar "boşta" durumuna geçer ve yeni bir atama için uygun hale gelir.

Bu diyagram genel olarak araçların sistemde hangi koşullarda kullanılabilir olduğunu ve her bir kullanım adımının sistem açısından nasıl izlendiğini ifade eder. Ayrıca, aynı anda birden fazla kullanıcıya atanamaması gibi kuralları da göz önünde bulundurur.

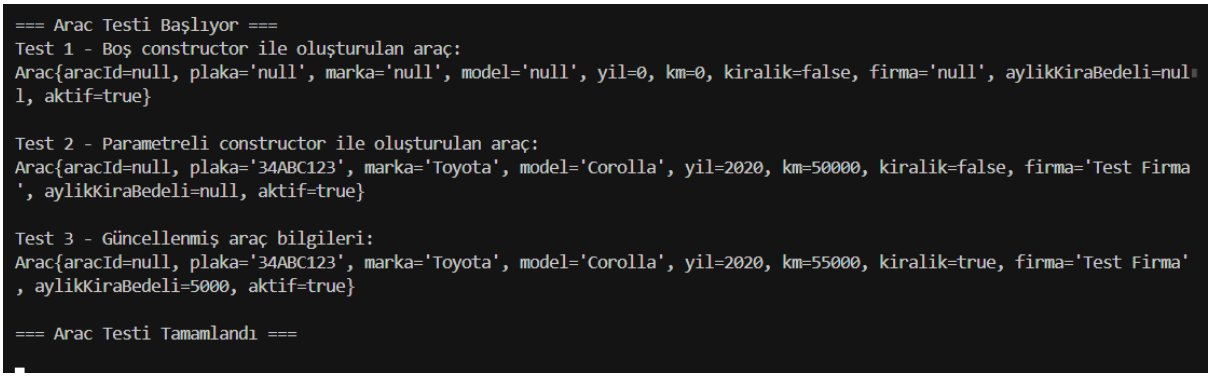


5. Birim Testleri

5.1. Test Senaryoları ve Kaynak Kodlar

Elif Gülerarslan:

```
public class TestArac {
    private static boolean testExecuted = false;

    public static void testAracOlustur() {
        if (testExecuted) {
            return;
        }
        testExecuted = true;

        System.out.println("\n=== Arac Testi Başlıyor ===");

        // Test 1: Boş constructor ile araç oluşturma
        Arac arac1 = new Arac();
        System.out.println("Test 1 - Boş constructor ile oluşturulan araç:");
        System.out.println(arac1);

        // Test 2: Parametreli constructor ile araç oluşturma
        Arac arac2 = new Arac("34ABC123", "Toyota", "Corolla", 2020, 50000,
"Test Firma");
        System.out.println("\nTest 2 - Parametreli constructor ile
oluşturulan araç:");
        System.out.println(arac2);

        // Test 3: Setter metodları ile araç bilgilerini güncelleme
        arac2.setKm(55000);
        arac2.setKiralik(true);
        arac2.setKiraBaslangicTarihi(LocalDate.now());
        arac2.setKiraBitisTarihi(LocalDate.now().plusMonths(12));
        arac2.setAylikKiraBedeli(new BigDecimal("5000"));

        System.out.println("\nTest 3 - Güncellenmiş araç bilgileri:");
        System.out.println(arac2);

        System.out.println("\n=== Arac Testi Tamamlandı ===\n");
    }
}
```



Esad Engin Işık:

```
public static void testFirmaYetkilisiOlustur() {
        if (testExecuted) {
            return;
        }
        testExecuted = true;

        System.out.println("\n=== Firma Yetkilisi Testi Başlıyor
===");

        // Test 1: Boş constructor ile firma yetkilisi oluşturma
        FirmaYetkilisi yetkili1 = new FirmaYetkilisi();
        System.out.println("Test 1 - Boş constructor ile
oluşturulan firma yetkilisi:");
        System.out.println(yetkili1);

        // Test 2: Parametreli constructor ile firma yetkilisi
oluşturma
        FirmaYetkilisi yetkili2 = new FirmaYetkilisi(
            "Ahmet",
            "Yılmaz",
            "ahmet@firma.com",
            "5551234567",
            "Test Firma A.Ş.",
            "sifre123"
        );
        System.out.println("\nTest 2 - Parametreli constructor ile
oluşturulan firma yetkilisi:");
        System.out.println(yetkili2);

        // Test 3: Rapor oluşturma fonksiyonlarını test etme
        testRaporOlusturma(yetkili2);

        System.out.println("\n=== Firma Yetkilisi Testi Tamamlandı
===\n");
    }
```

```
=== Firma Yetkilisi Testi Başlıyor ===
Test 1 - Boş constructor ile oluşturulan firma yetkilisi:
null null (null)

Test 2 - Parametreli constructor ile oluşturulan firma yetkilisi:
Ahmet Yılmaz (ahmet@firma.com)

Test 3 - Rapor Oluşturma Testi Başlıyor
```

Bilal Oğuz Erdoğdu:

```
public static void testRaporOlusturma(FirmaYetkilisi yetkili) {
        System.out.println("\nTest 3 - Rapor Oluşturma Testi Başlıyor");

        LocalDate baslangicTarihi = LocalDate.now().minusMonths(1);
        LocalDate bitisTarihi = LocalDate.now();

        // Test 3: Bakım raporu oluşturma
        HarcamaRaporu bakimRaporu =
yetkili.bakimRaporuOlustur(baslangicTarihi, bitisTarihi, null);
        System.out.println("\nTest 3 - Bakım Raporu:");
        System.out.println("Rapor Tipi: " + bakimRaporu.getRaporTipi());
        System.out.println("Başlangıç Tarihi: " +
bakimRaporu.getBaslangicTarihi());
        System.out.println("Bitiş Tarihi: " + bakimRaporu.getBitisTarihi());

        // Test 4: Yakıt raporu oluşturma
        HarcamaRaporu yakitRaporu =
yetkili.yakitRaporuOlustur(baslangicTarihi, bitisTarihi, null);
        System.out.println("\nTest 4 - Yakıt Raporu:");
        System.out.println("Rapor Tipi: " + yakitRaporu.getRaporTipi());
        System.out.println("Başlangıç Tarihi: " +
yakitRaporu.getBaslangicTarihi());
        System.out.println("Bitiş Tarihi: " + yakitRaporu.getBitisTarihi());

        // Test 5: Genel toplam raporu oluşturma
        HarcamaRaporu genelRapor =
yetkili.genelToplamRaporuOlustur(baslangicTarihi, bitisTarihi, null);
        System.out.println("\nTest 5 - Genel Toplam Raporu:");
        System.out.println("Rapor Tipi: " + genelRapor.getRaporTipi());
        System.out.println("Başlangıç Tarihi: " +
genelRapor.getBaslangicTarihi());
        System.out.println("Bitiş Tarihi: " + genelRapor.getBitisTarihi());
```

```
        // Test 6: Rapor dışa aktarma
        System.out.println("\nTest 6 - Rapor Dışa Aktarma:");
        String dosyaYolu = "raporlar/test_raporu.pdf";
        boolean pdfSonuc = yetkili.raporuPDFOlarakAktar(genelRapor,
dosyaYolu);
        System.out.println("PDF Aktarım Sonucu: " + (pdfSonuc ? "Başarılı" :
"Başarısız"));

        System.out.println("\n Rapor Oluşturma Testleri Tamamlandı");
    }
```

```
Test 3 - Bakım Raporu:
Rapor Tipi: Bakım
Başlangıç Tarihi: 2025-04-21
Bitiş Tarihi: 2025-05-21

Test 4 - Yakıt Raporu:
Rapor Tipi: Yakıt
Başlangıç Tarihi: 2025-04-21
Bitiş Tarihi: 2025-05-21

Test 5 - Genel Toplam Raporu:
Rapor Tipi: Genel Toplam
Başlangıç Tarihi: 2025-04-21
Bitiş Tarihi: 2025-05-21

Test 6 - Rapor Dışa Aktarma:
PDF raporu oluşturuluyor: raporlar/test_raporu.pdf
PDF Aktarım Sonucu: Başarılı
```

Ceyda Babuz:

```
public static void testHarcamaRaporuOlustur() {
        if (testExecuted) {
            return;
        }
        testExecuted = true;

        System.out.println("\n=== Harcama Raporu Testi Başlıyor ===");

        // Test 1: Boş constructor ile rapor oluşturma
        HarcamaRaporu rapor1 = new HarcamaRaporu();
        System.out.println("Test 1 - Boş constructor ile oluşturulan
rapor:");
        System.out.println(rapor1);
```

```
        // Test 2: Rapor bilgilerini ayarlama
        HarcamaRaporu rapor2 = new HarcamaRaporu();
        rapor2.setRaporTipi("Aylık Harcama Raporu");
        rapor2.setBaslangicTarihi(LocalDate.now().minusMonths(1));
        rapor2.setBitisTarihi(LocalDate.now());
        rapor2.setAracId(1);

        // Test 3: Harcama kalemleri ekleme
        HarcamaKalemi harcama1 = new HarcamaKalemi(1, "34ABC123",
LocalDate.now(), "Yakıt", new BigDecimal("500.50"), "Benzin alımı");
        HarcamaKalemi harcama2 = new HarcamaKalemi(1, "34ABC123",
LocalDate.now(), "Bakım", new BigDecimal("1000.00"), "Periyodik bakım");

        rapor2.harcamaEkle(harcama1);
        rapor2.harcamaEkle(harcama2);

        System.out.println("\nTest 2 & 3 - Harcama kalemleri eklenmiş
rapor:");
        System.out.println("Rapor Tipi: " + rapor2.getRaporTipi());
        System.out.println("Başlangıç Tarihi: " +
rapor2.getBaslangicTarihi());
        System.out.println("Bitiş Tarihi: " + rapor2.getBitisTarihi());
        System.out.println("Araç ID: " + rapor2.getAracId());
        System.out.println("Toplam Tutar: " + rapor2.getToplamTutar() + "
TL");
        System.out.println("Kategori Tutarları: " +
rapor2.getKategoriTutarMapi());

        System.out.println("\n=== Harcama Raporu Testi Tamamlandı ===\n");
    }
```

```
=== Harcama Raporu Testi Başlıyor ===
Test 1 - Boş constructor ile oluşturulan rapor:
HarcamaRaporu{raporTipi='null', başlangıçTarihi=null, bitişTarihi=null, toplamTutar=0}

=== Harcama Kalemi Testi Başlıyor ===
Test 1 - Boş constructor ile oluşturulan harcama kalemi:
HarcamaKalemi{id=0, aracId=0, plaka='null', harcamaTipi='null', tarih=null, tutar=null, aciklama='null'}

Test 2 - Parametreli constructor ile oluşturulan harcama kalemi:
HarcamaKalemi{id=0, aracId=1, plaka='34ABC123', harcamaTipi='Yakıt', tarih=2025-05-21, tutar=500.50, aciklama='Benzi
n alımı'}

Test 3 - Güncellenmiş harcama kalemi bilgileri:
HarcamaKalemi{id=0, aracId=1, plaka='34ABC123', harcamaTipi='Yakıt', tarih=2025-05-21, tutar=550.75, aciklama='Benzi
n ve yağ alımı'}

=== Harcama Kalemi Testi Tamamlandı ===
```