

**ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**



BLM 4537 PROJE RAPORU

SERVİS TAKİP SİSTEMİ

iOS (FLUTTER) MOBİL UYGULAMA PROJE RAPORU

Mustafa Erdem ÖZTÜRK

21290480

Enver Bağcı

Ocak, 2026

Video linki: <https://youtu.be/zo7lgftQ3Jk>

ÖZET

Bu çalışmada, servis taşımacılığı süreçlerinin kullanıcılar tarafından mobil cihaz üzerinden takip edilmesini sağlayan “Servis Takip Sistemi”nin **iOS (Flutter) mobil uygulama** bileşeni incelenmiştir. Uygulama; kullanıcıların sisteme giriş yapması, role bağlı olarak farklı ekranlara yönlendirilmesi (yolcu/şoför/yönetici), harita üzerinde canlı servis takibi, durak yönetimi ve rota görüntüleme gibi temel fonksiyonları sunmaktadır. Flutter ile geliştirilen mobil uygulama, backend ile **REST API** üzerinden iletişim kurmakta; gerçek zamanlı konum güncellemelerini ise **SignalR** üzerinden almaktadır.

Uygulamanın başlangıcında Splash ekranı üzerinden mevcut oturum kontrolü yapılarak kullanıcı tekrar giriş yapmak zorunda bırakılmamaktadır. Başarılı giriş sonrası rol bazlı Dashboard ekranları görüntülenmektedir. Yolcu tarafında aktif rotalar listelenebilmekte ve harita sayfası üzerinden rota/duraklar ile servis aracının anlık konumu izlenebilmektedir. Şoför tarafında rotalar listelenebilmekte, rota başlatma/tamamlama işlemleri yapılabilmekte ve ayrıca şoför konum takibi başlatılarak konum bilgisi SignalR hub’a gönderilebilmektedir. Yönetici tarafında ise temel bir dashboard ve harita ekranı iskeleti sunulmuş olup, yönetim modüllerinin genişletilmesine uygun bir temel hazırlanmıştır.

Sonuç olarak mobil uygulama; servis takibinin sahada kullanılmasına uygun, rol bazlı yetkilendirme ve harita/konum özelliklerini entegre eden bir mimari ile tasarlanmış ve uygulanmıştır.

İÇİNDEKİLER

ÖZET.....	1
İÇİNDEKİLER.....	2
1. GİRİŞ.....	1
1.1 Projenin Amacı.....	1
1.2 Projenin Kapsamı.....	1
2. MOTİVASYON VE KURAMSAL ARKA PLAN.....	2
2.1 Proje Motivasyonu ve Operasyonel İhtiyaç.....	2
2.2 Mevcut Yönetim Yöntemlerinin Yetersizlikleri.....	2
2.3 Web Tabanlı Yönetici Panelinin Rolü ve Katkıları.....	2
2.4 Gerçek Zamanlı Takip ve Anlık Karar Verme Mekanizmaları.....	3
2.5 Proje Kapsamının Operasyonel Planlamaya Entegrasyonu.....	3
2.6 Güvenlik, Yetkilendirme ve Oturum Yönetimi.....	3
2.7 Kuramsal Çerçeve ve Çalışmanın Konumlandırılması.....	4
2.8 Bölümün Genel Değerlendirmesi.....	4
3. KULLANILAN TEKNOLOJİLER VE ARAÇLAR.....	4
3.1 Mobil Uygulama.....	4
3.2 Harita ve Konum.....	4
3.3 API ve Veri.....	5
3.4 Gerçek Zamanlı İletişim.....	5
3.5 Yerel Depolama.....	5
4. UYGULAMA MİMARİSİ VE KATMANLAR.....	5
5. EKRANLAR VE KULLANICI AKIŞLARI.....	6
5.1 Splash ve Oturum Kontrolü.....	6
5.2 Giriş (Login).....	6
5.3 Kayıt (Register).....	7
5.4 Rol Bazlı Dashboard Yapısı.....	7
5.5 Harita (MapPage) – Temel Mantık.....	8
5.7 Şoför Akışları.....	9
6. API ENTEGRASYONU VE SERVİS KATMANI.....	10
7. GERÇEK ZAMANLI KONUM TAKİBİ (SIGNALR).....	10
7.1 Şoför Tarafı (Konum Gönderme):.....	10
8. GÜVENLİK, YETKİLENDİRME VE OTURUM YÖNETİMİ.....	11
8.1 Token ve Kullanıcı Bilgileri.....	11
8.2 Yetkilendirme.....	11
8.3 Güvenlik Notları.....	11
9. HATA YÖNETİMİ, PERFORMANS VE İYİLEŞTİRME ÖNERİLERİ.....	12
9.1 Hata Yönetimi.....	12
9.2 Performans.....	12
9.3 Önerilen Geliştirmeler.....	12
11. SONUÇ VE GELECEK ÇALIŞMALAR.....	12
Kaynaklar.....	13

1. GİRİŞ

Servis taşımacılığı süreçlerinde kullanıcıların en kritik ihtiyacı, servis aracının nerede olduğunu ve rota/durak durumlarını anlık olarak görebilmektir. Mobil uygulamalar, bu ihtiyaca en hızlı yanıt verebilen arayüzlerdir. Özellikle yolcuların servis aracını bekleme sürecinde konum takibi, şoförlerin rota yönetimi, yöneticilerin genel durum izleme ihtiyacı aynı sistem çatısı altında birleşmektedir.

Bu proje kapsamında geliştirilen Servis Takip Sistemi'nin mobil bileşeni Flutter ile hazırlanmış olup iOS ve Android için ortak bir kod tabanı hedeflemektedir. Bu rapor, mobil uygulamanın iOS tarafına odaklanarak uygulama mimarisini, ekranlarını, API entegrasyonunu ve gerçek zamanlı konum takibi yaklaşımını açıklamaktadır.

1.1 Projenin Amacı

Mobil uygulamanın amacı; kullanıcıların (yolcu/şoför/yönetici) servis takip sistemine mobil cihaz üzerinden erişebilmesi ve rolüne uygun fonksiyonları kullanabilmesidir. Sistem ile:

- Yolcu, aktif rotaları görebilmeli ve servis aracını haritada takip edebilmelidir.
- Şoför, kendisine atanmış rotaları görüntüleyebilmeli, rota başlatma/tamamlama işlemlerini yapabilmeli ve konumunu canlı olarak paylaşabilmelidir.

1.2 Projenin Kapsamı

Bu rapor, Flutter mobil uygulamadaki şu bileşenleri kapsar:

- Splash → oturum kontrolü ve otomatik yönlendirme
- Giriş/Kayıt ekranları
- Rol bazlı Dashboard'lar (Passenger/Driver/Admin)
- Harita ekranı (Google Maps + rota/durak çizimi + canlı konum)
- Yolcu için aktif rotalar listesi ve durakların yönetimi
- Şoför için rotalarını, rota başlat/tamamla, konum takibi başlat/durdur

- API servis katmanı (Auth/Route/Reservation/Notification)
- SignalR ile gerçek zamanlı iletişim
- SharedPreferences ile token ve kullanıcı verisi saklama

2. MOTİVASYON VE KURAMSAL ARKA PLAN

2.1 Proje Motivasyonu ve Operasyonel İhtiyaç

Günümüzde servis taşımacılığı operasyonları (okul, personel vb.) dinamik ve çok bileşenli süreçleri yönetmeyi gerektirmektedir. Bu süreçler; araç takibi, rota planlama, şoför-arac-yolcu eşleştirmesi ve anlık durum izleme gibi kritik görevleri içerir. Bu görevlerin manuel veya dağınık sistemlerle yürütülmesi, operasyonel verimsizliğe, gecikmelere ve hatalı bilgi akışına neden olmaktadır. Projenin temel motivasyonu, tüm bu operasyonel süreçleri dijital, merkezi ve verimli bir web tabanlı yönetici arayüzü ile tek noktadan yönetme ihtiyacından doğmuştur.

2.2 Mevcut Yönetim Yöntemlerinin Yetersizlikleri

Geleneksel servis yönetimi yöntemleri; konum verilerini anlık izleyememe, rota değişikliklerini hızlıca sisteme yansıtamama ve operasyonel varlıkların (araç/şoför) kayıtlarını güncel tutmada zorlanma gibi yetersizliklere sahiptir. Bu durum, özellikle acil durumlarda veya yoğun saatlerde karar alma süreçlerini yavaşlatır. Ayrıca, yetkisiz erişim riskleri barındıran veya tutarsız veri girişine açık olan dağınık kayıt sistemleri, operasyonun güvenilirliğini düşürmektedir. Geliştirilen bu web paneli, bu yetersizlikleri ortadan kaldırmayı ve operasyonun merkezi bir API üzerinden beslenen güvenilir bir arayüz ile yönetilmesini hedeflemiştir.

2.3 Web Tabanlı Yönetici Panelinin Rolü ve Katkıları

Web yönetici paneli, Servis Takip Sistemi'nin çok katmanlı mimarisinde yönetici rolündeki kullanıcılar için kritik bir arayüz görevi görür. Temel rolü, operasyonel varlık yönetimi (Araç, Şoför, Rota, Yolcu - CRUD işlemleri), Canlı Harita ile gerçek zamanlı konum verisi görselleştirmesi ve Dashboard ile özet metriklerin sunulmasıdır. Panel, arka plandaki API servisinden veri alarak çalışır; bu sayede iş kuralları ve veri

güvenliği API katmanında merkezileşirken, panel sadece yönetim ve görselleştirme katmanı olarak hizmet eder. Bu yaklaşım, sistemin ölçeklenebilirliğini ve sürdürülebilirliğini artırmaktadır.

2.4 Gerçek Zamanlı Takip ve Anlık Karar Verme Mekanizmaları

Sistemin "Canlı Harita" modülü ve SignalR teknolojisi entegrasyonu, operasyonun en önemli özelliklerindendir. Araç konumlarının anlık olarak harita üzerinde izlenmesi, operasyon yöneticisinin gecikmeleri veya rota dışı sapmaları saniyeler içinde fark etmesini sağlar. Bu anlık geri bildirim mekanizması, yöneticinin gerektiğinde hızlıca müdahale etmesini sağlayarak operasyonun kalitesini ve yolcu memnuniyetini doğrudan etkiler. Rota oluşturma ekranındaki harita tabanlı durak seçimi ve önizleme özellikleri de planlama aşamasında etkin karar vermeyi destekler.

2.5 Proje Kapsamının Operasyonel Planlamaya Entegrasyonu

Proje kapsamında yer alan Rota Yönetimi, Rota Oluşturma, Araç Yönetimi ve Şoför Yönetimi modülleri, servis operasyonunun temel planlama ve yürütme süreçlerini dijitalleştirmektedir. Özellikle harita tabanlı rota oluşturma ekranı, durakların sürükle-bırak ile sıralanabilmesi ve rota önizlemesi sunması sayesinde karmaşık rota planlama sürecini görsel ve kullanıcı dostu hale getirir. Bu entegre yaklaşım, kağıt tabanlı veya manuel yöntemlere kıyasla operasyonel planlama hızını ve doğruluğunu önemli ölçüde artırmaktadır.

2.6 Güvenlik, Yetkilendirme ve Oturum Yönetimi

Yönetici panelinin güvenliği, operasyon verilerinin hassasiyeti nedeniyle kritik öneme sahiptir. Giriş ekranı üzerinden sağlanan kimlik doğrulama sonrasında, API'den alınan Bearer Token'ın sonraki tüm isteklerde yetkilendirme bilgisi olarak taşınması esastır. Bu yaklaşım ile API ve web paneli arasında güvenli bir iletişim kanalı oluşturulmuştur. Ayrıca, sisteme erişim yalnızca "yönetici rolüne sahip" kullanıcılara sınırlandırılmış olup, rol tabanlı erişim kontrolü sayesinde operasyonel işlemlere

sadece yetkili personelin ulaşması sağlanmıştır. Oturum yönetimi, token'ın web oturumunda güvenli bir şekilde tutulmasıyla sağlanır.

2.7 Kuramsal Çerçeve ve Çalışmanın Konumlandırılması

Bu çalışma, Bilgi Sistemleri (Yönetici Paneli), Dağıtık Sistemler (Mobil İstemci, API ve Web Panelden oluşan çok katmanlı mimari) ve Gerçek Zamanlı Veri İletişimi (SignalR) alanlarında konumlanmaktadır. Web Yönetici Paneli, bir REST API İstemcisi olarak görev yaparak Veritabanı ve İş Kuralları katmanından ayrılmıştır. Bu mimari, sistemin her bir bileşeninin bağımsız olarak geliştirilebilmesi, bakımı yapılabilmesi ve ölçeklenebilmesi için modern yazılım mühendisliği prensiplerine uygun bir çerçeve sunmaktadır.

2.8 Bölümün Genel Değerlendirmesi

Bu bölümde ele alınan motivasyon ve kuramsal arka plan, web tabanlı yönetici panelinin, Servis Takip Sistemi'nin merkezi bir yönetim aracı olarak operasyonel verimliliği ve güvenilirliği artırmayı amaçladığını göstermektedir. Proje, sadece bir arayüz geliştirmekle kalmamış, aynı zamanda servis taşımacılığı operasyonlarının zorluklarına modern teknolojilerle (ASP.NET Web Forms, Google Maps API, SignalR) çözüm sunan bütüncül bir yönetim katmanı olarak tasarlanmıştır.

3. KULLANILAN TEKNOLOJİLER VE ARAÇLAR

3.1 Mobil Uygulama

- **Flutter (Dart SDK 3.9.x)**: Tek kod tabanıyla çoklu platform desteği.
- **Material Design / Material 3**: Uygulama arayüz bileşenleri.

3.2 Harita ve Konum

- **google_maps_flutter**: Harita görüntüleme, marker ve polyline çizimi.

- **location / geolocator / geocoding**: Konum izni, konum verisi alma, adres dönüşümleri için altyapı.

- **flutter_polyline_points**: Google Directions API'den gelen polyline kodlarını decode etmek için.

3.3 API ve Veri

- **http / dio**: REST API istekleri için istemci kütüphaneleri (projede ağırlıklı `http` kullanılıyor).

- **JSON encode/decode**: Model dönüşümleri.

3.4 Gerçek Zamanlı İletişim

- **signalr_netcore**: SignalR hub bağlantısı kurma; konum güncellemelerini gönderme/alma.

3.5 Yerel Depolama

- **shared_preferences**: Token ve kullanıcı bilgilerini cihazda tutmak için.

4. UYGULAMA MİMARİSİ VE KATMANLAR

Uygulama, tipik Flutter proje düzeni ile katmanlı bir yapıdadır:

- **pages/**: UI ekranları (auth, passenger, driver, admin, map vb.)

- **services/**: API çağrıları ve gerçek zamanlı servisler (AuthService, RouteService, LocationTrackingService vb.)

- **models/**: Veri modelleri (UserModel, DailyRouteModel, RouteStopModel, ReservationModel vb.)

- **utils/**: Sabitler ve yardımcıları (AppConstants gibi)

Bu mimari, UI'nın servis çağrılarından ayrıştırılmasını sağlar. Ekranlar servis katmanını kullanarak veri çekip gösterir; token yönetimi AuthService üzerinden merkezi olarak yürütülür.

5. EKРАНLAR VE KULLANICI AKIŞLARI

5.1 Splash ve Oturum Kontrolü

Uygulama açıldığında Splash ekranı:

- 1) kısa bir bekleme (UI/UX amaçlı) uygular,
- 2) `SharedPreferences` üzerinden token ve kullanıcı bilgilerini yüklemeye çalışır,
- 3) oturum varsa kullanıcı rolüne göre ilgili Dashboard'a yönlendirir,
- 4) oturum yoksa Login ekranını açar.

Bu sayede kullanıcı deneyimi iyileşir; her açılışta yeniden giriş zorunluluğu azaltılır.

5.2 Giriş (Login)

Login ekranında:

- e-posta ve şifre validasyonları yapılır,
- AuthService üzerinden `POST /auth/login` çağrısı yapılır,
- başarılı olursa UserModel oluşturulur ve saklanır,
- kullanıcı rolüne göre yönlendirme yapılır.

Rol bazlı yönlendirme:

- **Passenger** → **PassengerDashboard**
- **Driver** → **DriverDashboard**

- **Admin → AdminDashboard**

5.3 Kayıt (Register)

Register ekranı:

- ad-soyad, e-posta, telefon, şifre alanlarını içerir,
- rol seçimi (yolcu/şoför) seçilebilir durumdadır,
- AuthService ile `POST /auth/register` çağrısı yapar,
- kayıt başarılı olursa kullanıcıyı Login'e geri yönlendirir.

Not: Role seçimi UI'da bulunmasına rağmen backend register isteğinde rol parametresi gönderilmiyor ise bu raporda “geliştirilebilir” olarak belirtilmelidir (projedeki mevcut API sözleşmesine göre uyarlanır).

5.4 Rol Bazlı Dashboard Yapısı

Uygulamada her rol için ayrı dashboard yapısı vardır:

PassengerDashboard

- Bottom navigation: Ana Sayfa / Harita (rotalar) / Bildirimler / Profil
- “Duraklarım” sayfasına hızlı erişim
- Bildirimler ve bazı ekranlar “yakında” placeholder olabilir

DriverDashboard

- Bottom navigation: Ana Sayfa / Rotalarım / Harita / Profil
- Konum takibi başlat/durdur kontrolü (LocationTrackingService)


- Aktif rota seçildiğinde Harita ekranında rota izleme

AdminDashboard

- Bottom navigation: Dashboard / Harita / Kullanıcılar / Araçlar
- Kullanıcılar/Araçlar yönetmek için kullanılabilir
- Harita ekranı temel izleme için kullanılabilir

5.5 Harita (MapPage) – Temel Mantık

Harita sayfası, isteğe bağlı olarak bir rota parametresi alır:

- Eğer rota verilmişse duraklar API'den yüklenir,
- SignalR üzerinden rota grubuna katılarak şoför konum güncellemeleri dinlenir,
- Marker'lar: duraklar (numaralı), şoför/servis aracı () , kullanıcının mevcut konumu,
- Polyline çizimi:
- Öncelikli olarak Google Directions API'den gerçek yol çizimi,
- Hata durumunda durakları birleştiren basit dashed polyline.

Bu yaklaşım, kullanıcıya hem rota geometrisini hem de servis aracının anlık konumunu aynı harita üzerinde gösterir.

5.6 Yolcu Akışları

Aktif Rotalar (PassengerRoutesPage)

- API'den tüm aktif rotaları çeker (`GET /routes/active`),
- listeden seçilen rota MapPage'e aktarılır,
- yolcu, herhangi bir aktif rotayı takip edebilir.

Duraklarım (MyStopsPage)

- Kullanıcının duraklarını listeler (`GET /stops/my`),
- durak silme (`DELETE /stops/{id}`),
- yeni durak ekleme sayfasına yönlendirme (AddStopPage).

Bu modül, yolcunun servise binmek için kullandığı durak noktalarını yönetebilmesini sağlar.

5.7 Şoför Akışları

Rotalarım (DriverRoutesPage)

- Şoföre atanmış rotaları çeker (`GET /routes/driver/{driverId}`),
- rota başlatma (`PUT /routes/{id}/start`),
- rota tamamlama (`PUT /routes/{id}/complete`),
- rota başlatıldığında DriverDashboard üzerinde “aktif rota” ayarlanır ve Harita sekmesine geçiş yapılır.

Konum Takibi

- DriverDashboard üzerinde “Takibi Başlat/Durdur” butonu vardır.
- Konum izni ve servis açık mı kontrolleri yapılır.
- Konum değişikçe hareket/hareketsizlik analizi yapılarak status üretilir:
 - belirli metre üstü hareket → moving
 - uzun süre hareketsiz → stopped

- SignalR hub'a konum güncellemesi gönderilir: `UpdateDriverLocation(userId, lat, lng, status)`.

Bu sayede web panel veya yolcu uygulaması, şoför konumunu canlı takip edebilir.

6. API ENTEGRASYONU VE SERVİS KATMANI

Uygulama, API base URL bilgisini sabitlerden alır:

- `apiBaseUrl` örnek: `http://10.0.2.2:5000/api` (emülatör/yerel geliştirme için)
- iOS gerçek cihazda bu adresin ağ erişimine göre güncellenmesi gerekir.

Temel servisler:

- **AuthService**: login/register/logout + token saklama + auth header üretimi
- **RouteService**: şoför rotaları, aktif rotalar, rota durakları, rota başlat/tamamla
- **ReservationService**: rezervasyon listeleme/oluşturma/güncelleme/iptal + haftalık rezervasyon (API'ye bağlı)
- **NotificationService**: bildirimleri çekme/okundu işaretleme (geliştirme durumuna göre)
- **ServiceApiService**: günlük servisler vb. (varsa raporda yer verilir)

Bu yapı ile ekranlar, HTTP ayrıntılarını minimum bilerek servis katmanını çağırır.

7. GERÇEK ZAMANLI KONUM TAKİBİ (SIGNALR)

Sistemde iki yönlü gerçek zamanlı iletişim bulunur:

7.1 Şoför Tarafı (Konum Gönderme):

Şoför uygulaması:

- SignalR hub'a bağlanır,
- konum güncellemelerini `UpdateDriverLocation` metodu ile sunucuya gönderir,
- status alanı ile (moving/stopped) hareket bilgisini de iletir.

7.2 Yolcu Tarafı (Konum Dinleme)

Yolcu MapPage:

- SignalR'a bağlanır,
- ilgili rota grubuna katılır (routeId bazlı),
- driver konum güncellemelerini dinleyerek harita marker'ını günceller.

Bu mimari, canlı harita kullanım senaryosunu mobilde mümkün kılar.

8. GÜVENLİK, YETKİLENDİRME VE OTURUM YÖNETİMİ

8.1 Token ve Kullanıcı Bilgileri

- Token ve kullanıcı bilgileri `SharedPreferences` içinde saklanır.
- Uygulama açılışında token yüklü ise kullanıcı direkt yönlendirilir.

8.2 Yetkilendirme

- API isteklerinde `Authorization: Bearer <token>` başlığı kullanılır.
- Rol bazlı yönlendirme UI seviyesinde yapılır; asıl yetkilendirme API tarafında da enforced edilmelidir.

8.3 Güvenlik Notları

- Üretimde HTTPS zorunludur.

- Token saklama, daha güçlü güvenlik gerekirse secure storage ile güçlendirilebilir.
- Google Maps API key gibi değerlerin yönetimi ortam bazlı yapılmalıdır.

9. HATA YÖNETİMİ, PERFORMANS VE İYİLEŞTİRME ÖNERİLERİ

9.1 Hata Yönetimi

- Servislerde try/catch ile hata yakalanıp kullanıcıya SnackBar ile bilgi verilir.
- Bazı durumlarda “yakında” placeholder ekranlar mevcuttur.

9.2 Performans

- Harita üzerinde marker ve polyline güncellemeleri yoğun senaryoda optimize edilebilir.
- Directions API çağrıları maliyet/limit açısından kontrollü kullanılmalıdır (cache, sadece gerektiğinde çağırma).

9.3 Önerilen Geliştirmeler

- Tüm API URL yönetimini tek merkezden yapmak,
- Ortak bir API istemci katmanı yazıp tekrarları azaltmak,
- Yetkilendirme hatalarında otomatik logout/yeniden login yönlendirmesi,
- Yolcu modülüne düzenleme/silme işlemlerini eklemek,

11. SONUÇ

Bu raporda Servis Takip Sistemi'nin Flutter ile geliştirilen mobil uygulaması incelenmiştir. Uygulama; rol bazlı kullanıcı deneyimi, harita üzerinde rota/durak görüntüleme, SignalR ile gerçek zamanlı konum takibi ve temel durak yönetimi fonksiyonlarını sağlamaktadır. Mimari olarak ekranlar ile servis katmanı ayrıştırılmış, token yönetimi ve oturum sürekliliği SharedPreferences üzerinden desteklenmiştir.

Kaynaklar

1. Microsoft Docs – ASP.NET Web Forms
2. Microsoft Docs – HttpClient Kullanımı
3. Microsoft Docs – SignalR (JavaScript Client)
4. Google Maps Platform – Maps JavaScript API
5. Bootstrap 5 Dokümantasyonu
6. Newtonsoft.Json Dokümantasyonu