C-LENS (CLASSLENS) YAZILIM GEREKSİNİMLERİ BELGESİ (SRS)

BÖLÜM 1: GİRİŞ VE GENEL TANIM

1.1. Amaç, Kapsam ve Belge Hedefleri

Bu Yazılım Gereksinimleri Belgesi (SRS), ClassLens olarak adlandırılan yüz tanıma tabanlı mobil yoklama sisteminin kapsamlı gereksinimlerini tanımlamak amacıyla hazırlanmıştır. Sistemin temel amacı, öğretim görevlilerinin (hocaların) yoklama süreçlerini otomatikleştirerek zamandan tasarruf sağlamak ve öğrencilerin mobil cihazları üzerinden güvenli ve kolay bir şekilde kimliklerini doğrulayarak derse katılımını kaydetmektir.

Bu belgenin kapsamı, Flutter tabanlı mobil istemci uygulaması, ASP.NET Core Web API tabanlı arka uç hizmetleri ve PostgreSQL ilişkisel veritabanı yönetim sisteminin tüm fonksiyonel, fonksiyonel olmayan, mimari ve güvenlik gereksinimlerini içerir. Belge, geliştirme, kalite güvence (QA) ve proje yönetimi ekipleri için birincil teknik sözleşme görevi görecektir.

1.2. Ürün Perspektifi ve Teknolojik Kısıtlar

ClassLens, eğitim kurumları için tasarlanmış, iki ana kullanıcı grubuna hizmet eden mobil merkezli bir yoklama yönetim çözümüdür: Hocalar (yönetim ve raporlama) ve Öğrenciler (biyometrik katılım ve iletişim). Sistemin mimarisi ve teknoloji yığını, ürünün yüksek güvenlik ve gizlilik standartlarına uyumunu sağlamak üzere belirlenmiştir [2.4].

1.2.1. Kritik Teknolojik Yığın ve Mimarisi

Sistemin geliştirilmesinde kullanılacak temel bileşenler şunlardır:

1. **Mobil Uygulama:** Flutter (Dart) kullanılarak geliştirilecektir. Android 10 ve üzeri cihazlarda uyumluluk zorunluluğu vardır.
2. **Backend Servis:** ASP.NET Core (C#) kullanılarak geliştirilecek Web API mimarisi [2.4].
3. **Veritabanı:** İlişkisel veri yönetimi için PostgreSQL seçilmiştir [2.4].
4. **Yüz Tanıma Mekanizması:** Gizlilik ilkesi gereği, biyometrik işlemlerin yüksek performansta cihaz üzerinde (on-device) gerçekleşmesi zorunludur. Bu, Google ML Kit'in yüz tespiti (detection) yeteneği ile birlikte özelleştirilmiş bir TFLite modelinin (FaceNet gibi) eşleştirme (recognition) için kullanılmasıyla sağlanacaktır.

1.2.2. Temel Kısıtlar ve Varsayımlar

- **Gizlilik Kısıtlaması (NFR-3):** Öğrenciye ait ham yüz verisi veya hesaplanan biyometrik vektör temsili (Embedding), mobil cihaz dışına asla gönderilmez ve sunucuda saklanmaz. Sunucu sadece yoklama kaydı, doğruluk skoru ve canlılık durumu gibi metadataları işler.

Bu, özellikle GDPR ve CCPA gibi veri gizliliği düzenlemelerine uyum sağlamak için kritik bir mimari karardır [2.5].

- **Donanım Zorunluluğu:** Uygulamanın çalışacağı tüm mobil cihazlarda işlevsel kamera erişimi bulunmalıdır. İnternet bağlantısı, yoklama verilerinin sunucuya senkronize edilmesi için gereklidir, ancak biyometrik doğrulama işlemi çevrimdışı çalışabilmelidir [2.5, 2.6].

1.3. Tanımlar ve Kısaltmalar

Terim	Açıklama
ML Kit	Google'ın mobil cihazlarda çalışan hızlı yüz tespiti (detection) kütüphanesi.
JWT	JSON Web Token, hocaların kimlik doğrulama standardı.
Embedding	Bir yüzün benzersiz matematiksel vektör temsili. Bu, yüz tanıma modelinin çıktısıdır.
1:N Eşleştirme (Recognition)	Bir yüzün, yerel depolamada bulunan birden fazla kayıtlı yüzle karşılaştırılarak kimliğin belirlenmesi.
Liveness Detection	Sunulan yüz görüntüsünün gerçek bir kişiye mi ait olduğunu, yoksa sahte bir fotoğraf/video (spoofing) olup olmadığını belirleyen kritik kontrol mekanizması.
Geofencing	Dersin başlangıç konumuna göre belirlenen coğrafi sınır. Öğrencinin yoklamaya katılımı için bu sınır içinde bulunması zorunludur.
SSL Pinning	Mobil uygulamanın, API sunucusuna bağlanırken yalnızca önceden tanımlanmış (gömülü) sunucu sertifikalarına güvenmesini sağlayan gelişmiş güvenlik mekanizması.

BÖLÜM 2: ÖZEL GEREKSİNİMLERİN ANALİZİ

2.1. Fonksiyonel Gereksinimler (FR) Detaylandırması

2.1.1. Kullanıcı ve Kimlik Doğrulama

- **FR-1 (Hoca Girişi):** Öğretim üyesi, güvenli bir şekilde e-posta ve şifreyle giriş yapmalıdır. Başarılı doğrulama sonrası, ASP.NET Core API'den kısa ömürlü bir JWT Access Token ve uzun ömürlü bir Refresh Token almalıdır. Access Token, API çağrıları için kullanılırken; Refresh Token, kullanıcı deneyimini bozmadan yeni Access Token almayı sağlar. Hatalı giriş durumunda kullanıcı dostu hata mesajı sunulmalıdır (FR-3).
- **FR-2 (Öğrenci Girişi/Doğrulaması):** Öğrenci, ders yoklamasına katılmak için kamera üzerinden yüzünü taratır. Bu süreç, cihaz üzerinde (on-device) gerçekleşen yüz tespiti, canlılık kontrolü ve kimlik eşleştirmeyi (1:N) içerir. Eşleştirme, anlık yüz Embedding vektörünün, cihazda kayıtlı olan öğrenci Embedding'lerine karşı bir minimum `Threshold` değerini aşan bir `Face Verification Score` üretmesiyle başarılı kabul edilir.

2.1.2. Yoklama Yönetimi ve Kayıt İşleme

- **FR-4 (Yoklama Oturumu Başlatma):** Hoca, sisteme kayıtlı bir ders için yoklama oturumu başlatmalıdır. Oturumu başlatırken, yoklamanın geçerli olacağı coğrafi sınırları (geofence merkezi ve yarıçapı) belirlemek zorundadır. Bu, yoklamanın sadece tanımlı derslikte alınmasını sağlar.
- **FR-5 (Yoklama Kaydının Tetiklenmesi):** Öğrencinin mobil uygulaması, biyometrik (yüz tanıma ve canlılık) ve coğrafi (cihaz GPS konumu) tüm doğrulamaları başarıyla tamamladıktan sonra, yoklama kaydını tek, atomik bir API çağrısıyla sunucuya (ASP.NET Core) iletmelidir.
- **FR-6 (Kayıt Veri Alanları):** Yoklama kaydının veri bütünlüğünü ve raporlama yeteneklerini sağlamak için aşağıdaki alanları içermesi zorunludur:
    - `Student ID`
    - `Course ID`
    - `Timestamp_UTC`: Zaman serisi analizleri ve filtreleme için kritik olan, zaman dilimi bilgisi içeren zaman damgası.
    - `Verification_Score`: Yüz doğruluk skoru (minimum eşik değeri aşılmış olmalıdır).
    - `Location_Point`: Coğrafi konum verisi. Performanslı coğrafi sorgular için PostGIS uyumlu `GEOMETRY` tipi kullanılmalıdır.
    - `Liveness_Status`: Canlılık kontrolünün sonucu (Başarılı/Başarısız).
- **FR-7 (Filtreleme ve Raporlama):** Hoca, geçmiş yoklama kayıtlarını tarih aralığına ve derse göre etkin bir şekilde filtreleyebilmelidir. Bu gereksinim, büyük veri kümelerinde bile (yüz binlerce kayıt) anlık yanıt sağlamak için Bölüm 5'te detaylandırılan kompozit indeksleme stratejisine bağımlıdır.

2.1.3. İletişim, Paylaşım ve Sınıf İçi Özellikler

- **FR-8 & FR-9 (Duyurular/Paylaşım):** Hocalar, dersle ilgili kritik duyuruları ve kaynakları paylaşabilmelidir. Benzer şekilde, öğrencilerin de sınıf içinde not veya kaynak paylaşımına izin verilmelidir. Bu veriler `Announcements` tablosunda saklanacaktır.
- **FR-10 (Rastgele Öğrenci Seçici):** Hoca, sözlü veya tartışma başlatmak amacıyla sınıftan rastgele bir öğrenci seçebilmelidir.
- **FR-11, FR-12 (Gelecek Özellikleri):** Gelecekte eklenecek olan Mini Quiz, Chat ve Anlık Oylama/Anket gibi modüllerin, gerçek zamanlı iletişim yeteneklerini desteklemek üzere, mimarinin SignalR entegrasyonu için şimdiden hazır olması gerekmektedir.

2.2. Fonksiyonel Olmayan Gereksinimler (NFR) Detaylandırması

2.2.1. Performans ve Kullanılabilirlik

- **NFR-1 Performans:** Yüz tanıma işlemi, akıcı bir kullanıcı deneyimi sağlamak amacıyla 2 saniye içinde tamamlanmalıdır. Cihaz üzerinde çalışan modern ML/TFLite modelleri, ağ gecikmesini ortadan kaldırdığı için bu sürenin ideal olarak 100 milisaniyenin altında kalması beklenmektedir.
- **NFR-5 Kullanılabilirlik (Offline Desteği):** Mobil uygulama, internet bağlantısı kesintisi durumunda (örneğin dersliğin zayıf ağ koşulları) dahi yoklama sürecini başlatabilmeli, yüzü doğrulayabilmeli ve kayıt verilerini yerel olarak (bir Çevrimdışı Veritabanında)

güvenli bir şekilde saklayabilmelidir. Bağlantı geri geldiğinde, bu yerel kayıtlar otomatik olarak sunucuya senkronize edilmelidir (Sync Queue Pattern).

2.2.2. Güvenlik, Gizlilik ve Uyumluluk

- **NFR-2 Güvenlik Protokolleri:**
    - Tüm sunucu-istemci iletişimi, Man-in-the-Middle (MiTM) saldırılarını önlemek için zorunlu olarak HTTPS/TLS üzerinden şifrelenmelidir. API projelerinin HTTP dinlememesi veya HTTP isteklerini 400 (Bad Request) ile reddetmesi önerilir.
    - Mobil istemci, sunucunun sertifikasını kontrol etmek ve sahte sertifikaları engellemek için **SSL Pinning** mekanizmasını uygulamalıdır.
    - Hoca kimlik doğrulamasında kullanılan JWT Access ve Refresh Tokenlar, cihaz üzerinde **Flutter Secure Storage** (Android'de Keystore, iOS'te Keychain) kullanılarak şifrelenmiş biçimde saklanmalıdır. Hassas verilerin `SharedPreferences` veya dosya sisteminde düz metin olarak saklanmasından kesinlikle kaçınılmalıdır.
- **NFR-3 Veri Gizliliği:** Bu, sistemin temel direğidir. Yüz Embedding'leri yalnızca mobil cihazda şifreli kalır. Sunucu tarafında hiçbir biometrik veri saklanamaz veya işlenemez.
- **NFR-6 Hata Yönetimi:** API ve iş mantığı hataları, kullanıcının anlayabileceği açık ve kullanıcı dostu mesajlarla bildirilmelidir (örneğin, "Kamera erişimi engellendi," veya "Geofence sınırları dışında bulunuyorsunuz").

BÖLÜM 3: MİMARİ VE TEKNİK TASARIM

3.1. Genel Sistem Mimarisi ve Ayrım Prensibi

ClassLens sistemi, modern bir üç katmanlı mimariye (İstemci, Servis, Veri) dayanır. Uygulamanın tasarımı, her bileşenin bağımsız sorumlulukları olmasını sağlayan **İlgi Alanlarının Ayrılması (Separation-of-Concerns)** prensibine sıkı sıkıya uyar.

3.2. Mobil Uygulama Mimarisi (Flutter)

Mobil uygulama mimarisi, kodun sürdürülebilirliğini, test edilebilirliğini ve ölçeklenebilirliğini sağlamak için Model-View-ViewModel (MVVM) desenini benimseyecektir.

3.2.1. MVVM Uygulaması

- **Views:** Kullanıcı arayüzünü (UI) oluşturur.
- **View Models (veya Interactors/Use Cases):** İş mantığını ve durum yönetimini (state management) içerir. Veri manipülasyonu ve API çağrılarının koordinasyonundan sorumludur.
- **Repositories:** Veri Katmanını temsil eder. Uzaktan veri kaynakları (ASP.NET Core API) ve yerel veri kaynakları (Yerel Secure Storage, Offline DB) arasındaki erişimi soyutlar.
- **Kritik Bileşen: `FaceRecognitionRepository`:** Bu özel repository, on-device ML Kit/TFLite işlemlerini, yerel Embedding depolama ve biyometrik karşılaştırma mantığını yönetecek ve veri katmanından soyutlanmış olacaktır.

3.2.2. Çevrimdışı Veri Senkronizasyonu

NFR-5 gereksinimini karşılamak için, mobil uygulama içinde merkezi bir `SyncService` bulunmalıdır. Bu servis, öğrenci yoklama kaydı (FR-6) oluşturulduğunda:

1. Kaydı hemen yerel bir veritabanına kaydeder (hızlı kullanıcı geri bildirimi için).
2. Kaydı bir senkronizasyon kuyruğuna (`Sync Queue`) ekler.
3. Uygulama çevrimiçi olduğunda, kuyruktaki veriler tek tek API'ye gönderilir ve başarılı kayıt sonrası kuyruktan silinir.

3.3. Backend Servis Mimarisi (ASP.NET Core API)

Backend, yüksek performanslı, güvenli ve ölçeklenebilir bir RESTful API olarak tasarlanacaktır.

3.3.1. Güvenlik ve Kimlik Yönetimi

- **JWT ve Refresh Token Flow:** API, `JwtBearerHandler` kullanarak gelen Access Token'ları doğrulamalıdır. Access Tokenlar, kısa süreli (örneğin 5-10 dakika) olmalıdır.
- **Refresh Token Revocation:** Kullanıcının sürekli giriş yapmasını önlemek için, mobil uygulama Access Token süresi dolduğunda (401 yanıtı) Refresh Token ile `Token/Refresh` endpoint'ine çağrı yapmalıdır. Sunucu, bu çağrıda:
    1. Gelen Refresh Token'ın geçerliliğini ve süresini kontrol eder (veritabanında saklanan değere karşı).
    2. Başarılıysa, yeni bir Access Token ve yeni bir Refresh Token oluşturur.
    3. Güvenlik gereği, eski Refresh Token'ı derhal veritabanından siler (tek kullanımlık hale getirir/revocation).
- **API Performans İyileştirmeleri:** Veritabanından veri okuyan API uç noktaları, Entity Framework Core'un (EF Core) izleme mekanizmasını atlamak ve bellek tüketimini azaltmak için `AsNoTracking()` kullanmalıdır. Ayrıca, raporlama sorgularında büyük koleksiyonların yüklenmesini önlemek için sayfalandırma (Pagination) zorunlu hale getirilmelidir.

3.3.2. Gerçek Zamanlı İletişim Altyapısı

Gelecekteki genişletilebilirlik hedeflerini (Chat, Quiz, Anket - FR-12) desteklemek amacıyla, ASP.NET Core API'si **SignalR** teknolojisi kullanılarak tasarlanmalıdır. Geleneksel HTTP Polling yöntemleri, sunucu kaynaklarını hızla tüketme ve ölçeklenebilirlik sorunlarına yol açma riski taşır. SignalR, WebSockets'i kullanarak düşük gecikmeli, çift yönlü ve ölçeklenebilir gerçek zamanlı iletişim kanalları sağlar.

- **Hub Tanımları:** En az iki ana Hub tanımlanmalıdır:
    1. `AttendanceHub`: Hocaların yoklama oturumu durumunu veya bir öğrencinin başarıyla yoklamaya katıldığını anlık olarak bildirmek için.
    2. `ClassroomHub`: Gelecekteki sınıf içi sohbet ve oylama sistemleri için.

BÖLÜM 4: BİYOMETRİK VE CİHAZ ÜZERİ İŞLEM MİMARİSİ

Bu bölüm, NFR-3 (Gizlilik) ve NFR-1 (Performans) gereksinimlerini karşılamak için kritik olan, cihaz üzerinde yüz tanıma sürecini detaylandırır.

4.1. Yüz Tanıma İş Akışı Mimarisi

ML Kit, Google tarafından sunulan bir araç olmasına rağmen, genellikle sadece yüz tespiti (Face Detection) ve özellik analizi (gözler kapalı mı, gülümsüyor mu) yapar, bireyleri tanımaz (Recognition). Tam bir yoklama sistemi için 1:N eşleştirme gerektiğinden, özelleştirilmiş bir süreç zorunludur.

Aşama	İşlem	Amaç ve Çıktı	Teknoloji
1. Tespit (Detection)	Kameradan görüntü yakalama.	Görüntüde bir yüzün varlığını ve sınır kutusunu (bounding box) hızla bulmak.	Google ML Kit
2. Canlılık Kontrolü (Liveness)	Öğrencinin rastgele bir hareket yapmasını isteme.	Görüntünün sahte (spoof) olmadığını doğrulamak.	Özel Liveness Modülü/ SDK (veya Pose Estimation)
3. Embedding Üretimi (Recognition)	Tespit edilen yüzü kırpıp TFLite modeline besleme.	Yüzün 128 boyutlu benzersiz Embedding Vektörünü (V_anlık) üretmek.	TensorFlow Lite (TFLite)
4. Eşleştirme (Comparison)	V_anlık'ı yerel olarak saklanan V_kayıtlı vektörleriyle karşılaştırma.	Öklid Mesafesi kullanarak kimliği doğrulama ve doğruluk skoru (Verification Score) üretme.	Dart Kod Mantığı (Cihaz Üzerinde)

4.2. Liveness Detection ve Fraud Önleme

Canlılık tespiti, sistemin güvenilirliği için vazgeçilmezdir. Yüksek çözünürlüklü fotoğraflar veya videolar kullanılarak yapılan sahtecilik (spoofing) girişimlerini önler. Raporlama aşamasında FR-6 gereksinimi doğrultusunda, her yoklama kaydına `Liveness_Status` alanının eklenmesi, denetlenebilirliği artırır.

4.3. Ön Kayıt (Enrollment) Gereksinimleri

Yüz tanıma başarısızlık oranını (R-1 Riski) azaltmak için, ön kayıt (enrollment) aşamasında veri kalitesine ve çeşitliliğine odaklanılmalıdır. Algoritmik sapmayı (bias) ve farklı aydınlatma koşullarında performansı artırmak için, öğrencinin yalnızca önden değil, hafif açılı pozlarda ve çeşitli aydınlatma altında fotoğraflarını çekmesi zorunludur.

BÖLÜM 5: VERİ VE DEPOLAMA YÖNETİMİ (POSTGRESQL)

Veritabanı mimarisi, yüksek hacimli yoklama verilerini (Time-Series) yönetmek, hızlı raporlama yapmak (FR-7) ve coğrafi kontrolleri etkinleştirmek için PostgreSQL üzerine kurulmuştur.

5.1. İlişkisel Veritabanı Şeması ve PostGIS Entegrasyonu

Gereksinimler, konuma dayalı doğrulama (Geofencing) yapılmasını zorunlu kıldığından, standart PostgreSQL kurulumu yetersizdir. Etkili Geofencing sorguları için **PostGIS** uzantısının entegre edilmesi ve coğrafi veri tiplerinin kullanılması zorunludur.

5.1.1. Kritik Tablolar ve Veri Tipleri

Tablo	Kritik Alan	Veri Tipi	Amaç
`Users`	`password_hash`	CHAR(60)	Güçlü şifreleme hashlerini tutmak (Bcrypt/Argon2 uyumlu).
`Courses`	`geofence_point`	GEOMETRY(Point, 4326)	Dersin yoklama merkezi konumunu WGS 84 formatında saklamak.
`Attendance`	`timestamp_utc`	TIMESTAMP WITH TIME ZONE	Zaman serisi filtreleme için doğru zaman damgası.
`Attendance`	`location_point`	GEOMETRY(Point, 4326)	Yoklama alınan öğrencinin kesin konumu.

5.2. Performans Odaklı İndeksleme Stratejileri

`Attendance` tablosu, sistemin en yüksek yazma ve okuma trafiğine sahip tablosu olacaktır. Aşırı indeksleme (over-indexing) yazma (INSERT) işlemlerinin maliyetini artıracağından , indeksler yalnızca en sık kullanılan raporlama sorgularını (FR-7) hızlandırmak için oluşturulmalıdır.

5.2.1. Attendance Tablosu İndeks Optimizasyonları

1. **Bileşik B-Tree İndeksi:** En önemli raporlama ve filtreleme yolu `course_id` ve `timestamp_utc` alanları üzerinden gerçekleşecektir. Bu nedenle, `(course_id, timestamp_utc, student_id)` alanları üzerinde bileşik B-Tree indeksi oluşturulacaktır. PostgreSQL, bu bileşik indeksi, sorgular `course_id` ile başladığı sürece verimli bir şekilde kullanacaktır.
2. **GIST İndeksi (Coğrafi İndeks):** `location_point` alanında coğrafi sorguları (örneğin Geofence sınırları içinde mi? - `ST_DWithin` kontrolü) hızlandırmak için PostGIS'e özel GIST (Generalized Search Tree) indeksi kullanılacaktır. Standart B-Tree indeksleri bu tür uzamsal analizler için uygun değildir.

BÖLÜM 6: RİSK ANALİZİ VE AZALTMA PLANI

Sistemde tanımlanan en kritik riskler, biyometrik sahtecilik (Spoofing) ve ağ güvenliği zafiyetleridir. Bu riskler, çok katmanlı doğrulama ve sıkı protokol uygulamalarıyla azaltılacaktır.

6.1. Güvenlik Mekanizmalarının Entegrasyonu

NFR/FR	Mekanizma	Geliştirme Katmanı	Uygulama Detayı	Risk Azaltımı
NFR-2 (Güvenlik)	SSL Pinning	Flutter (Client)	Uygulama, önceden gömülü sertifika parmak izi ile sunucuyu doğrular.	MiTM (Man-in-the-Middle) saldırılarını engeller.
NFR-2 (Güvenlik)	Refresh Token Flow	ASP.NET Core API	Access Token'ları kısa tutar ve Refresh Token'ın tek kullanımlık olmasını sağlar (revocation).	Token çalınma süresini sınırlar.
FR-6, R-4 Azaltma	Geofence Server Doğrulaması	ASP.NET Core API / PostGIS	Mobil cihazdan gelen konum bilgisinin, dersin coğrafi sınırları içinde olup olmadığı sunucuda kontrol edilir.	Konum sahteciliği (GPS Spoofing) riskini azaltır.
R-2 Azaltma	Liveness Detection	Flutter (ML/TFLite)	Öğrencinin canlı bir kişi olduğunu kanıtlar.	Fotoğraf/video ile sahtecilik (Spoofing) girişimlerini engeller.

6.2. Yüz Tanıma ve Yoklama Kaydı İşleme Akışı Kontrol Noktaları

Yoklama kaydının güvenilirliğini sağlamak için her adımda kritik doğrulama kontrolleri (Kontrol Noktaları) uygulanmalıdır:

Adım	Sistem	Doğrulama Mekanizması	Başarısızlık Durumu Kodu	Gerekli Eylem
1	Mobil	Kamera Erişimi	ATT_101 (Kamera Hatası)	Kullanıcıdan izin istenmesi veya donanım kontrolü.
2	Mobil	Canlılık Tespiti	ATT_202 (Spoofing)	Yoklama kaydı reddedilir, hoca bilgilendirilebilir.
3	Mobil	Yüz Eşleştirme (1:N)	ATT_201 (Eşleşme Yok)	Yüzün yeniden taranması veya ışık koşullarının düzeltilmesi istenir.
4	Mobil	Konum Tespiti (GPS)	ATT_102 (GPS Verisi Yok)	Konum servisi izinleri kontrol edilmeli.
5	Backend	Yoklama Oturumu Aktif mi?	ATT_401 (Oturum Bitmiş)	Hoca tarafından oturumun yeniden başlatılması.

Adım	Sistem	Doğrulama Mekanizması	Başarısızlık Durumu Kodu	Gerekli Eylem
6	Backend	Geofence Doğrulaması	ATT_203 (Geofence Dışında)	Öğrenciye fiziksel olarak derse yaklaşması mesajı verilir.
7	Backend	Veritabanına Kayıt	ATT_500 (DB Hatası)	Veri bütünlüğü ve sunucu loglaması kontrolü.

SONUÇ VE ÖNERİLER

ClassLens projesi, modern mobil teknolojileri (Flutter/ASP.NET Core/PostgreSQL) kullanarak geleneksel yoklama süreçlerini otomatize etme hedefine ulaşırken, iki kritik mimari zorunluluğu çözmüştür: mutlak kullanıcı gizliliği ve yüksek performans.

Kritik Çıkarımlar ve Mimari Kararlar:

1. **Gizlilik Odaklı Biyometri:** Google ML Kit'in sınırlamaları aşılmış ve biometrik kimlik doğrulaması (recognition), sadece cihaz üzerinde çalışan TFLite modelleri ve lokal embedding karşılaştırması kullanılarak tasarlanmıştır. Bu tasarım, NFR-3 gereksinimini tam olarak karşılamakta ve sunucu tarafında biometrik veri saklama yükümlülüğünü ortadan kaldırmaktadır.
2. **Yüksek Güvenlik Standartları:** Hocaların kimlik doğrulamasında kısa ömürlü JWT Access Tokenlar ve veritabanında saklanıp yönetilen, tek kullanımlık Refresh Tokenlar kullanılması, yetkisiz erişim riskini en aza indirir. Buna ek olarak, MiTM saldırılarına karşı mobil istemcide **SSL Pinning** zorunluluğu getirilerek ağ iletişimi katmanının güvenliği pekiştirilmiştir.
3. **Veri Performansı ve Geofencing:** Yüksek hacimli zaman serisi verisi (Attendance) için PostgreSQL'in seçilmesi ve PostGIS uzantısının entegre edilmesi kararı, FR-4 (Geofencing) ve FR-7 (Hızlı Raporlama) gereksinimlerini doğrudan desteklemektedir. Özellikle `(course_id, timestamp_utc, student_id)` üzerindeki bileşik indeks, hoca raporlarının sorgu süresini önemli ölçüde hızlandıracaktır.
4. **Geleceğe Yönelik Hazırlık:** Mini Quiz ve Chat gibi gelecek modüller için ASP.NET Core API'sine SignalR entegrasyonunun şimdiden dahil edilmesi, projenin büyümesi ve gerçek zamanlı özelliklerin eklenmesi için sağlam ve ölçeklenebilir bir zemin oluşturur, bu da gelecekteki geliştirme maliyetlerini düşürür.