

BilanCompetence.AI - Eksiklikler ve Güncel Durum Raporu

Tarih: 23 Ekim 2025

Repo: <https://github.com/lekesiz/bilancompetence.ai>

Son Commit: cc6f6ab (23 Ekim 2025, 03:11)

Durum: Aktif Geliştirme - Sprint 7 Planlaması



Güncel Durum Özeti

Repo İstatistikleri

- Dil:** TypeScript (Ana), JavaScript
- Branch:** Sadece `main` (tek branch)
- Açık PR:** 0
- Açık Issue:** 0
- Son Aktivite:** 23 Ekim 2025 (bugün)
- Toplam Dosya:** 180 TypeScript, 6 JavaScript, 43 Test dosyası

Son Geliştirmeler

- ✓ **Sprint 7 - Task 1:** Qualiopi Compliance Module (Tamamlandı)
- 4 Phase'de tamamlanmış (Database, Services, API, Frontend)
- 16 yeniden kullanılabilir React komponenti
- E2E testler yazılmış
- 📅 **Sprint 7 - Task 2:** Scheduling/Appointment System (PLANLAMA AŞAMASINDA)
- Detaylı implementation planı hazırlanmış (1003 satır)
- Henüz implementasyon başlamamış
- Tahmini süre: 12-17 gün (~2.5 hafta)



Kritik Eksiklikler (Yüksek Öncelik)

1. Zamanlama/Randevu Sistemi - TAM EKSİK ⚠️

Durum: Sadece plan var, implementasyon YOK

Eksik Bileşenler:

- ✗ Backend routes (`/api/scheduling` , `/api/appointments` , `/api/availability`)
- ✗ Backend services (`schedulingService` , `appointmentService` , `availabilityService`)
- ✗ Database migrations (4 tablo: `availability_slots` , `bookings` , `session_records` , `reminders`)
- ✗ Frontend sayfalar (`consultant availability` , `beneficiary booking` , `calendar views`)
- ✗ Frontend komponentler (`AvailabilityCalendar` , `BookingForm` , `SessionDetails`)
- ✗ Email reminder sistemi
- ✗ Conflict detection logic

- ❌ Timezone handling
- ❌ Calendar integration (.ics export)

Etki: Kullanıcılar randevu alamıyor, danışmanlar müsaitlik belirleyemiyor

Önerilen Aksiyon: Sprint 7 Task 2 planını onaylayıp hemen implementasyona başla

2. Error Handling Eksiklikleri ⚠️

Etkilenen Dosyalar (12 dosya):

```
- apps/backend/src/services/csvService.ts
- apps/backend/src/services/fileService.ts
- apps/backend/src/services/analyticsService.ts
- apps/backend/src/services/userService.ts
- apps/backend/src/services/notificationService.ts
- apps/frontend/app/(auth)/login/page.tsx
- apps/frontend/app/(auth)/register/page.tsx
- apps/frontend/app/(protected)/recommendations/page.tsx
- apps/frontend/app/(protected)/saved-jobs/page.tsx
```

Sorun: Async fonksiyonlarda try-catch blokları eksik

Etki: Production'da hata durumunda uygulama crash olabilir, kullanıcıya anlamlı hata mesajı gösterilemiyor

Önerilen Aksiyon:

```
// Önce
async function getData() {
  const result = await api.fetch();
  return result;
}

// Sonra
async function getData() {
  try {
    const result = await api.fetch();
    return result;
  } catch (error) {
    logger.error('getData failed:', error);
    throw new AppError('Veri alınamadı', 500);
  }
}
```

3. Güvenlik Riskleri 🔒

Tespit Edilen Sorunlar (11 dosya):

a) Potansiyel Hardcoded Secrets:

- apps/backend/src/__tests__/routes/auth.integration.spec.ts
- apps/backend/src/__tests__/services/authService.spec.ts
- apps/frontend/e2e/assessment-wizard.e2e.ts
- apps/mobile/lib/api.ts

Sorun: Test dosyalarında ve kod içinde hardcoded API keys/tokens olabilir

Önerilen Aksiyon:

1. Tüm hardcoded secrets'ları `.env` dosyasına taşı
2. Test dosyalarında mock/fake credentials kullan
3. Git history'den sensitive data temizle (git-filter-repo)

4. Console.log Kullanımı (Production) 🐛

Etkilenen Dosyalar (13 dosya):

- apps/backend/src/index.ts
- apps/backend/src/services/satisfactionSurveyService.ts
- apps/backend/src/services/emailService.ts
- apps/backend/src/services/complianceReportService.ts
- apps/frontend/app/(protected)/dashboard/components/AdminDashboard.tsx
- apps/frontend/app/(protected)/recommendations/page.tsx

Sorun: Production kodunda console.log kullanımı

Etki:

- Performance düşüşü
- Sensitive data leak riski
- Log yönetimi zorluğu

Önerilen Aksiyon:

```
// Önce
console.log('User data:', userData);

// Sonra
import logger from '@lib/logger';
logger.info('User data fetched', { userId: userData.id });
```

⚠ Orta Öncelikli Eksiklikler

5. Boş Fonksiyonlar 📝

Tespit Edilen (11 dosya):

- apps/frontend/hooks/useAssessmentWizard.ts
- apps/frontend/hooks/useJobRecommendations.ts
- apps/backend/src/services/assessmentService.ts
- apps/backend/src/services/documentArchiveService.ts
- apps/backend/src/routes/recommendations.ts





Sorun: Placeholder/stub fonksiyonlar var, implementasyon eksik

Önerilen Aksiyon: Her boş fonksiyonu incele ve:




- Gerekliyse implement et
- Gereksizse sil
- TODO yorumu ekle

6. Dokümantasyon Eksiklikleri

Eksik Olanlar:

-  docs/ dizini yok
-  API dokümantasyonu eksik (Swagger/OpenAPI spec yok)
-  Deployment guide var ama güncel değil
-  Developer onboarding guide yok

Mevcut Olanlar:





-  README.md var (temel bilgiler)
-  Sprint raporları çok detaylı (100+ MD dosyası)
-  Migration guide var

Önerilen Aksiyon:






1. docs/ dizini oluştur
2. API dokümantasyonu ekle (Swagger UI)
3. Architecture diagram ekle
4. Contributing guide ekle
5. Troubleshooting guide ekle

7. Test Coverage Eksiklikleri

Mevcut Durum:

-  43 test dosyası var
-  Unit testler: Backend services, Frontend components
-  Integration testler: API endpoints
-  E2E testler: Qualiopi module, Assessment wizard

Eksik Testler:

-  Scheduling/Appointment sistemi testleri (sistem henüz yok)
-  Auth flow E2E testleri eksik
-  Mobile app testleri eksik
-  Performance testleri yok
-  Load testleri yok

Önerilen Aksiyon:

1. Scheduling sistemi implementasyonu ile birlikte testleri yaz
2. Auth E2E testleri ekle (login, register, password reset)
3. Jest coverage raporu çıkar: `npm test -- --coverage`

Düşük Öncelikli Eksiklikler

8. Kod Kalitesi İyileştirmeleri

a) Duplicate Code:

- Sprint rapor dosyaları arasında tekrar eden içerik (202510221.md - 2025102279.md)
- Dashboard komponentlerinde benzer pattern'ler

b) Naming Conventions:

- Bazı dosyalarda tutarsız naming (camelCase vs snake_case)

c) Code Organization:



- Root dizinde 100+ MD dosyası (docs/ altına taşınmalı)

Önerilen Aksiyon:




1. Duplicate MD dosyalarını arşivle veya sil
2. ESLint/Prettier kurallarını sıkılaştır
3. Folder structure refactor (docs/, archive/)

9. Dependency Management

Mevcut Durum:

-  package.json dosyaları var (root, backend, frontend)
-  Dependencies güncel görünüyor

Potansiyel Sorunlar:

-  Deprecated dependencies kontrolü yapılmamış
-  Security audit yapılmamış
-  Bundle size optimization yapılmamış

Önerilen Aksiyon:




```
# Deprecated packages kontrolü
npm outdated

# Security audit
npm audit
npm audit fix

# Bundle analysis
npm run build -- --analyze
```

10. Environment Configuration

Mevcut Durum:

-  .env.example var ve kapsamlı (200+ satır)
-  Email/SMTP config var
-  Scheduling/Calendar config YOK (sistem henüz yok)



Eksik Config:

```
# Scheduling System (eklenecek)
CALENDAR_INTEGRATION_ENABLED=true
CALENDAR_PROVIDER=google # google, outlook, ical
GOOGLE_CALENDAR_API_KEY=
GOOGLE_CALENDAR_CLIENT_ID=
REMINDER_EMAIL_ENABLED=true
REMINDER_HOURS_BEFORE=24,1
SESSION_DEFAULT_DURATION_MINUTES=120
```



Önerilen Aksiyon Planı

Faz 1: Kritik Eksiklikleri Gider (1-2 Hafta)

Hafta 1:




1.  Sprint 7 Task 2 planını onayla
2.  Scheduling sistemi Phase 1-2 (Backend + Database)
 - Migrations oluştur (4 tablo)
 - Services yaz (schedulingService, appointmentService)
 - API endpoints yaz (13+ endpoint)
 - Unit testler yaz

Hafta 2:

3.  Scheduling sistemi Phase 3-4 (Frontend + Testing)
 - Consultant availability UI
 - Beneficiary booking UI
 - Calendar views
 - E2E testler
4.  Security audit yap
 - Hardcoded secrets temizle
 - Environment variables düzenle
 - Git history temizle (gerekirse)


Faz 2: Orta Öncelikli İyileştirmeler (1 Hafta)

Hafta 3:


5.  Error handling ekle (12 dosya)
 - Try-catch blokları ekle
 - Error logging ekle
 - User-friendly error messages
6.  Console.log temizliği (13 dosya)
 - Logger service kullan
 - Production'da console.log kaldır
7.  Boş fonksiyonları tamamla (11 dosya)
 - Implement veya sil
 - TODO yorumları ekle

Faz 3: Dokümantasyon ve Optimizasyon (3-5 Gün)


Gün 1-2:

8.  Dokümantasyon oluştur
 - docs/ dizini oluştur
 - API dokümantasyonu (Swagger)
 - Architecture diagram
 - Developer guide

Gün 3-4:

9.  Test coverage artır
 - Auth E2E testler
 - Coverage raporu çıkar
 - Eksik testleri tamamla

Gün 5:

10.  Kod temizliği
 - Duplicate dosyaları arşivle
 - Root dizini düzenle
 - ESLint/Prettier çalıştır



Başarı Metrikleri

Tamamlanma Kriterleri:

Kritik (Faz 1):

- [] Scheduling sistemi %100 çalışır durumda
- [] 0 hardcoded secret
- [] 0 security vulnerability (npm audit)

Orta (Faz 2):

- [] Tüm async fonksiyonlarda error handling
- [] 0 console.log production kodunda
- [] 0 boş fonksiyon

Düşük (Faz 3):

- [] API dokümantasyonu %100
- [] Test coverage >80%
- [] Root dizinde <10 MD dosyası



Sonuç

Genel Değerlendirme:

• Güçlü Yönler:

- Qualiopi modülü mükemmel implement edilmiş
- Test coverage iyi (43 test dosyası)
- Sprint planlaması çok detaylı
- Deployment hazır

• Zayıf Yönler:

- Scheduling sistemi tamamen eksik (kritik özellik)
- Error handling eksiklikleri
- Security riskleri (hardcoded secrets)
- Production'da console.log kullanımı

Öncelik Sırası:

1. **EN ÖNCELİKLİ:** Scheduling sistemi implementasyonu (Sprint 7 Task 2)
2. **ÇOK ÖNEMLİ:** Security audit ve hardcoded secrets temizliği
3. **ÖNEMLİ:** Error handling ve console.log temizliği
4. **İYİLEŞTİRME:** Dokümantasyon ve test coverage

Tahmini Tamamlanma Süresi:

- **Minimum:** 2.5 hafta (sadece kritik)
- **Optimal:** 4 hafta (kritik + orta öncelikli)
- **İdeal:** 5 hafta (tüm eksiklikler)

Rapor Tarihi: 23 Ekim 2025

Hazırlayan: AI Code Analysis Tool

Versiyon: 1.0