Quality Assurance (QA) Plan

Proje Adı: TerraSense

Oluşturanlar (List of Contributors):

- Ahmet Yusuf ŞAHİN
- Furkan YAHŞİ
- Mehmet Ali USLU
- Yusuf Can BAHADIRLIOĞLU

İçindekiler (Table of Contents)

Görev Matrisi (Task Matrix)

Bölüm	Sorumlu Kişi	Açıklama
QA Plan Formatı	Mehmet Ali USLU	Genel şablon hazırlığı
QA Strategy	Yusuf Can BAHADIRLIOĞLU	Test stratejisi ve metodolojiler
Quality Factors & Metrics	Mehmet Ali USLU	Kalite faktörleri ve metriklerin tanımlanması
Test Plan	Yusuf Can BAHADIRLIOĞLU	Test senaryoları ve bug tracking yaklaşımı

1. Quality Assurance Strategy

1.1 Overview

Bu QA Planı, TerraSense projesinde yazılım kalitesini sağlamak ve sürdürülebilir kılmak için izleyeceğimiz süreçleri tanımlar. Amaç, **hataları erken aşamada bulmak**, kullanıcı deneyimini geliştirmek ve proje tesliminde yüksek kaliteli bir ürün sunmaktır.

1.2 Testing Methodologies

1. Unit Testler (Birim Testleri)

- Her bir fonksiyon ve modül seviyesinde testler yapılır.
    - "Toprak analizi verileri doğru işleniyor mu?"
    - Geliştiriciler tarafından yazılır ve otomatik olarak çalıştırılır.

2. Entegrasyon Testleri

- Farklı sistem bileşenlerinin birlikte doğru çalışıp çalışmadığını kontrol eder.
    - Toprak analizi ve öneri motorunun entegrasyonu.
    - Flutter mobil uygulaması ile backend API entegrasyonu.

3. Sistem Testleri

- Sistemin uçtan uca doğru çalışıp çalışmadığını doğrular.
    - Toprak analizi girişi, hava durumu uyarısı ve raporlama süreçlerinin doğruluğu.

4. Usability Test (Kullanılabilirlik Testleri)

- Kullanıcı deneyimini test eder.
    - Mobil arayüzdeki form girişlerinin anlaşılabilirliği.
    - Hata mesajlarının kullanıcı dostu olup olmadığı.

1.3 Automated vs. Manual Testing

1. **Otomatik Testler (Automated Testing)**

- **Unit Testler (Birim Testleri):**
  - Kodun en küçük parçalarını test eder.
    - Bir hesap makinesi uygulamasında toplama fonksiyonunun doğru sonuç verdiğini test etmek.
- **Entegrasyon Testleri:**
  - Farklı modüllerin veya servislerin birlikte doğru çalıştığını doğrular.
    - Kullanıcı girişi sonrası veritabanından doğru bilginin çekilip çekilmediğinin kontrolü.
- **CI/CD Pipeline Entegrasyonu:**
  - Unit testler ve bazı entegrasyon testleri, CI/CD sürecine entegre edilerek her yeni kod değişikliğinde otomatik çalıştırılır.
  - Başarısız olan testler geliştirme sürecinde tespit edilir ve kodun birleştirilmesi (merge) önlenir.
  - Araçlar: Selenium, Cypress

2. **Manuel Testler (Manual Testing)**

- **Usability Test (Kullanılabilirlik Testi):**
  - Kullanıcının arayüzü rahatça kullanıp kullanamadığını ölçmek için uygulanır.
    - Kullanıcı, kayıt olma işlemini kolayca tamamlayabiliyor mu? Butonlar, formlar anlaşılır mı?
- **Arayüz Kontrolleri (UI Testing):**
  - Web veya mobil uygulamanın tasarım unsurlarının doğru görüntülenip görüntülenmediğini test eder.
    - Farklı ekran boyutlarında veya cihazlarda UI öğelerinin hizalanması.

- **Özel Senaryo Testleri:**
  - Standart akışların dışında kalan, hata oluşturabilecek veya yanlış çalışabilecek uç senaryoların manuel olarak test edilmesi.
    - Kullanıcı farklı bir dil seçtiğinde çeviri ve yönlendirmeler doğru çalışıyor mu?
    - Veri tabanında hiç veri yokken raporlama ekranı nasıl tepki veriyor?
    - Aşırı büyük dosya yükleme işlemi nasıl yönetiliyor?

2. Quality Factors & Metrics

Kalite Faktörü	Açıklama	Ölçüm Metrik
Performans	Sistem yanıt süresi	Ortalama yanıt süresi (ms)
Güvenlik	Sistemi siber saldırılara karşı koruma düzeyi	Tespit edilen güvenlik açığı sayısı ve bu açıkların önemi
Kullanılabilirlik	Kullanıcı dostu arayüz	Kullanıcı memnuniyeti anket skoru (1-5)
Bakım Kolaylığı	Kod değiştirilebilitesi	Kod karmaşıklık ölçümü (Cyclomatic Complexity vb.)

3. Test Plan

3.1 Test Cases

Aşağıda 5 temel test senaryosu örneği verilmiştir:

Test Adı	Amaç	Giriş	Beklenen Çıktı
Toprak Analizi Girişi	Kullanıcının pH, azot, fosfor, potasyum verilerini girip kaydetmesi.	pH: 6.5, Azot: 10, Fosfor: 5, Potasyum: 3	"Öneri Listesi" JSON veya UI olarak dönmelidir.
Hava Durumu Uyarısı	Aşırı yağış durumunda bildirim gönderilmesi.	API sonucu (Yağış > 50 mm)	"Aşırı Yağış Uyarısı" bildirimi.
Su Yönetimi Önerisi	Domates bitkisi için sulama periyotlarının belirlenmesi.	Nem %40, Sıcaklık 25°C	Haftada 2 kez sulama.
Dönemsel Raporlama	Mısır 2025 Mart için aylık rapor.	Veritabanı + Mart 2025	PDF rapor oluşturulmalıdır.
Boş Veri Senaryosu	Veri olmadığında hata vermeden uyarı mesajı.	Kullanıcı kayıtlı değil	"Toprak analizi verileriniz bulunamadı." mesajı.

3.2 Bug Tracking

Araç: GitHub Issues

Adımlar:

1. **Hata Bildirimi:**
    - Geliştiriciler, test mühendisleri veya kullanıcılar tarafından bir hata tespit edildiğinde, GitHub Issues üzerinde yeni bir *issue* açılır.
    - Hata açıklayıcı bir başlık ve detaylı bir açıklama ile belgelenir. Açıklamada hatanın nasıl ortaya çıktığı, tekrar üretilebilir olup olmadığı ve varsa hata ile ilgili ekran görüntüleri veya hata mesajları eklenir, kısaca hata ile ilgili dokümantasyon yapılmış olur.
2. **Hata Seviyesinin Belirlenmesi:**
    - Hatanın etkisi ve aciliyeti değerlendirilerek seviyelendirilir:
        - **Kritik:** Sistemin temel işlevlerini durduran veya veri kaybına sebep olan hatalar.
        - **Yüksek:** Önemli fonksiyonların hatalı çalışmasına neden olan ancak sistemin tamamen çökmesine sebep olmayan hatalar.
        - **Orta:** Alternatif yollarla aşılabilen, kullanıcı deneyimini etkileyen hatalar.
        - **Düşük:** Küçük görsel hatalar veya kullanım kolaylığını etkileyen ama işlevselliği bozmayan sorunlar.
    - Hata seviyesine göre *issue* uygun etiketlerle işaretlenir.
3. **Sorumlu Atama ve Çözüm Süreci:**
    - Hatanın çözümü için bir geliştirici veya ekip sorumlu olarak atanır.
    - Çözüm süreci boyunca *issue* üzerinde güncellemeler paylaşılır.
    - Hatanın nedeni tespit edilerek gerekli kod değişiklikleri yapılır.
4. **Düzeltme Sonrası Test:**
    - Hata giderildikten sonra ilgili değişiklik test ekibi veya hata bildiren kişi tarafından tekrar test edilir.
    - Düzeltmenin başarılı olduğu doğrulanırsa *issue* kapanmaya hazır hale getirilir.
5. **Kapatma:**
    - Hata başarılı bir şekilde giderildiği ve testlerden geçtiği doğrulandıktan sonra *issue* kapatılır.
    - Eğer hata devam ediyorsa *issue* tekrar açılarak ek düzenlemeler yapılır.