Yazılım Gereksinim Belirtimi (SRS)

Joplin

Tuğcan Topaloğlu

Yıldız Teknik Üniversitesi

02.01.2025

İçindekiler

1. Giriş

1.1 Amaç

Bu doküman, Joplin uygulamasının geliştirilmesi ve kullanıcı gereksinimlerinin karşılanması amacıyla detaylı bir yazılım gereksinimleri açıklamasıdır. Joplin, not oluşturma, düzenleme, etiketleme ve senkronizasyon gibi özellikleri sunan bir açık kaynaklı uygulamadır.

1.2 Kapsam

Joplin, kişisel ve profesyonel kullanıcılar için geliştirilmiş, açık kaynak kodlu bir not alma ve organizasyon yazılımıdır. Uygulama, notları, yapılacaklar listelerini ve etiketleri organize etmeyi, farklı cihazlar arasında eşitlemeyi ve kullanıcılara offline erişim sağlamayı hedefler.

1.3 Referanslar ve Kullanılan Standartlar

Bu Yazılım Gereksinimleri Belgesi (SRS), yazılım mühendisliği alanında en yüksek kabul görmüş yöntemlerin ve standartların bir yansımasıdır. Doküman, geliştirme süreçlerinin şeffaflığını ve adaptasyonunu sağlayan Agile metodolojisine sıkı sıkıya bağlı kalmakta; yazılım kalitesini tanımlayan uluslararası kabul görmüş ISO/IEC 25010 standartlarını ve yazılım mühendisliğinin uygulama standartlarından olan IEEE'nin belirlediği normları referans alarak şekillendirilmiştir. Bu belge, yazılımın fonksiyonel ve işlevsel olmayan gereksinimlerini tanımlarken güncel yazılım mühendisliği teknikleri, metodolojiler ve en iyi uygulamaları benimsemiş; bunları sistem tasarımı ve gereksinim analizi süreçlerine entegre etmiştir. SRS, aynı zamanda sürekli değişen teknolojik trendleri ve pazar dinamiklerini göz önünde bulundurarak, esnek ve ölçeklenebilir bir yazılım mimarisi oluşturulmasını teşvik eden bir rehber olarak hizmet eder. Bu, yazılımın mevcut ve gelecekteki ihtiyaçları karşılayacak şekilde evrimleşebilmesi için kritik öneme sahiptir.

1.4 Hedef Kitle

Bu belge, yazılım geliştiricileri, sistem analistleri, proje yöneticileri ve kalite güvence ekipleri için hazırlanmıştır.

2. Genel Açıklamalar

Bu bölüm, not tutma sistemi olan Joplin'in genel çevresi, etkileşimde bulunduğu diğer sistemler, kullanıcı ihtiyaçlarına genel bakış ve sistemin kısıtlamalarını ve varsayımlarını ele alır.

2.1 Ürün Açıklaması

Joplin, Node.js, Electron ve React teknolojileriyle geliştirilmiştir. SQLite, uygulamanın temel veritabanı olarak kullanılır ve senkronizasyon için çeşitli bulut hizmetleri (Dropbox, OneDrive, Nextcloud vb.) desteklenir. Modern ve kullanıcı dostu bir arayüz sunarak not alma, düzenleme ve yönetim süreçlerini optimize eder. Uygulama, çeşitli platformlarda (Windows, MacOS, Linux, Android ve iOS) çalışarak çapraz platform desteği sunar.

2.2 Ürün Fonksiyonları

- **Not Yönetimi:**
  - Not oluşturma, düzenleme ve silme.
  - Markdown desteğiyle zengin metin düzenleme.
  - Notları birleştirme ve .MD, .pdf gibi formatlarda çıktılarını alabilme.

- **Yapılacaklar Listesi:**
  - Görev ekleme, düzenleme ve tamamlama.
  - Görev önceliklendirme.
  - Görev atama.

- **Eşitleme:**
  - Bulut hizmetleri (Dropbox, OneDrive, Google Drive vb.) ile not eşitleme.
  - Yerel ağa bağlanıp lokal cloud ile çalışabilme.
  - Notların sadece lokalde de tutulabilmesi.

- **Arama:**
  - Tam metin arama ve gelişmiş filtreleme.
  - Tarih, etiket veya içerik türüne göre hızlı arama.

- **Etiketleme ve Organize Etme:**
  - Notları etiketlerle gruplandırma.
  - Klasör ve alt klasör yapılarıyla organize etme.
  - Klasörleri etiketleyip sürükle bırakma yöntemiyle klasör içerisine item atabilme.

- **API Entegrasyonu:**
  - Uygulama, üçüncü taraf yazılımlarla entegrasyonu kolaylaştıran RESTful API sunar.
  - Notları oluşturma, okuma, güncelleme ve silme (CRUD işlemleri).
  - Kullanıcı yetkilendirmesi için OAuth 2.0 desteği.
  - JSON formatında veri iletimi.

    Örnek API Uç Noktaları:
    - *GET /api/notes* — Tüm notları getirir.
    - *POST /api/notes* — Yeni bir not oluşturur.
    - *PUT /api/notes/:id* — Belirli bir notu günceller.
    - *DELETE /api/notes/:id* — Belirli bir notu siler.
    - *GET /api/tags* — Etiketleri getirir.
    - *POST /api/tags* — Yeni bir etiket oluşturur.
    - *POST /api/notes/:id/tags* — Notlara etiket ekler.
    - API'ler, kullanıcıların kişisel projelerinde veya üçüncü taraf entegrasyonlarında Joplin'e erişim sağlamalarını kolaylaştırır.

- **Veri İçe/Dışa Aktarma**
  - Notların Markdown, JSON veya HTML formatlarında dışa aktarılması.
  - Üçüncü taraf not alma uygulamalarından veri içe aktarımı.

2.3 Kullanıcı İhtiyaçlarına Genel Bakış

Sistem, kullanıcılara aşağıdaki ihtiyaçları karşılayacak şekilde tasarlanmıştır:

- **Kolay Not Yönetimi:** Kullanıcıların notlarını hızlı ve kolay bir şekilde oluşturması, düzenlemesi ve organize etmesi.
- **Hassas Arama:** Notlar içinde tam metin arama ile içeriklere kolay erişim.
- **Esnek Veri Yönetimi:** Notların Markdown formatında düzenlenmesi, etiketlenmesi ve farklı cihazlar arasında eşitlenmesi.
- **Güvenilirlik ve Hata Yönetimi:** Sistem, kullanıcı hatalarına karşı dayanıklı olmalı ve anlamlı hata mesajları sağlamalıdır.

3. Spesifik Gereksinimler

3.1 Fonksiyonel Gereksinimleri

3.1.1 FR1: Not Yönetimi

- **FR1.1 Not Ekleme:** Kullanıcı, bir not oluşturabilmeli ve içeriğini Markdown formatında düzenleyebilmelidir.
- **FR1.2 Not Düzenleme:** Kullanıcı, mevcut bir notu düzenleyebilmelidir.
- **FR1.3 Not Silme:** Kullanıcı, bir notu silebilmelidir.
- **FR1.4 Not Görüntüleme:** Kullanıcı, kaydedilmiş notları liste halinde görüntüleyebilmelidir.
- **FR1.5 Notu Favorilere Ekleme:** Kullanıcı, notlarını favorilere ekleyebilmelidir.
- **FR1.6 Notu Favorilerden Çıkarma:** Kullanıcı, favorilerden not kaldırabilmelidir.
- **FR1.7 Not Şifreleme:** Kullanıcı, notlarını şifre korumalı hale getirebilmelidir.

3.1.2 FR2: Etiket Yönetimi

- **FR2.1 Etiket Ekleme:** Kullanıcı, notlara etiket ekleyebilmelidir.
- **FR2.2 Etiket Silme:** Kullanıcı, notlardan etiketleri kaldırabilmelidir.
- **FR2.3 Etiket Bazlı Arama:** Kullanıcı, etiketlere göre notları filtreleyebilmelidir.
- **FR2.4 Etiket Yönetimi:** Kullanıcı, birden fazla etiketle not gruplandırabilmelidir.

3.1.3 FR3: Eşitleme

- **FR3.1 Bulut Hizmetleri ile Eşitleme:** Kullanıcı, notlarını Dropbox, OneDrive veya WebDAV gibi bulut hizmetleriyle eşitleyebilmelidir.
- **FR3.2 Çevrimdışı Mod:** Kullanıcı, internet bağlantısı olmadığında notlarını düzenleyebilmeli ve bağlantı sağlandığında senkronizasyon gerçekleşmelidir.
- **FR3.3 Senkronizasyon Güvenliği:** Eşitleme işlemleri sırasında verilerin şifrelenmesi sağlanmalıdır.

3.1.4 FR4: API Entegrasyonu

- **FR4.1 Not CRUD İşlemleri:** Kullanıcı, API aracılığıyla not oluşturma, okuma, güncelleme ve silme işlemleri yapabilmelidir.
- **FR4.2 Yetkilendirme:** API, OAuth 2.0 standardını kullanarak güvenli bir kimlik doğrulama sağlamalıdır.
- **FR4.3 JSON Veri Desteği:** API üzerinden gönderilen ve alınan tüm veri JSON formatında olmalıdır.

3.1.5 FR5: Veri İçe/Dışa Aktarma

- **FR5.1 Veri Dışa Aktarma:** Kullanıcı, notlarını Markdown, JSON veya HTML formatında dışa aktarabilmelidir.
- **FR5.2 Veri İçe Aktarma:** Kullanıcı, farklı formatlardaki notları uygulamaya yükleyebilmelidir.
- **FR5.3 JSON Veri Uyumluluğu:** Tüm veri, JSON formatında sorunsuz şekilde serialize ve deserialize edilebilmelidir.

3.1.6 FR6: Arama

- **FR6.1 Anahtar Kelime ile Arama:** Kullanıcı, notlar içinde anahtar kelimeyle arama yapabilmelidir.
- **FR6.2 Gelişmiş Filtreleme:** Kullanıcı, tarih, etiket veya içerik türüne göre arama yapabilmelidir.
- **FR6.3 Sıralama:** Kullanıcı, notlarını oluşturulma veya düzenlenme tarihlerine göre sıralayabilmelidir.

3.1.7 FR7: Çöp Kutusu Yönetimi

- **FR7.1 Çöp Kutusundan Geri Yükleme:** Kullanıcı, çöp kutusundaki notları geri yükleyebilmelidir.
- **FR7.2 Çöp Kutusunu Temizleme:** Kullanıcı, çöp kutusundaki tüm öğeleri temizleyebilmelidir.

3.1.8 FR8: Conflict Yönetimi

- **FR8.1 Çakışma Uyarısı:** Kullanıcı, çakışma durumunda bilgilendirilmelidir.
- **FR8.2 Çakışan Notu Yönetme:** Çakışan notlar, yeni bir klasöre taşınabilmelidir.

3.2 Performans Gereksinimleri

- **PR1:** Uygulama, 10.000 not üzerinde performans kaybı yaşamadan çalışmalıdır.
- **PR2:** Arama sorguları, 1 saniyeden kısa sürede sonuç döndürmelidir.
- **PR3:** Uygulama başlangıç süresi, maksimum 3 saniye olmalıdır.
- **PR4:** Not kaydetme süresi 2 saniyeden kısa sürede tamamlanmalıdır.
- **PR5:** Mobil cihazlarda not kaydetme işlemi, 1 saniyeden kısa sürede sonuçlanmalıdır.
- **PR6:** Büyük boyutlu notlar (10.000 karakter üzerinde) 3 saniyeden kısa sürede yüklenmelidir.

- **PR7:** Çeşitli filtreleme seçenekleriyle yapılan aramalarda maksimum 2 saniye yanıt süresi sağlanmalıdır.
- **PR8:** Eşitleme işlemleri, maksimum 5 saniye içinde tamamlanmalıdır.
- **PR9:** Uygulamanın CPU kullanımı, maksimum %50'nin altında tutulmalıdır.
- **PR10:** Bellek kullanımı, 100 MB'ın altında olmalıdır.

3.3 Güvenilirlik Gereksinimleri

- **SR1:** Kullanıcı verileri şifrelenmiş bir şekilde saklanmalıdır.
- **SR2:** API erişimi yalnızca yetkili kullanıcılar tarafından yapılmalıdır.
- **SR3:** Zararlı veri girişlerinin (SQL enjeksiyonu, XSS vb.) engellenmesi sağlanmalıdır.
- **SR4:** Oturum süreleri ayarlanabilir olmalı ve süre dolduğunda oturum otomatik olarak sonlanmalıdır.
- **SR5:** Kullanıcı giriş hataları loglanmalı ve gerektiğinde analiz için kullanılabilmelidir.
- **SR6:** Senkronizasyon sırasında veri bütünlüğü korunmalıdır.

3.4 Kullanılabilirlik Gereksinimleri

- **UR1:** Kullanıcı arayüzü sezgisel ve kullanıcı dostu olmalıdır.
- **UR2:** Uygulama, farklı ekran boyutlarına ve cihazlara uyumlu şekilde tasarlanmalıdır.
- **UR3:** Tema değişikliği (örneğin, açık ve koyu tema) desteklenmelidir.
- **UR4:** Çoklu dil desteği sunulmalıdır.
- **UR5:** Mobil cihazlarda dokunmatik etkileşim sorunsuz çalışmalıdır.
- **UR6:** Yeni kullanıcılar için bir başlangıç rehberi veya ipucu özelliği sunulmalıdır.
- **UR7:** Ana menü ve alt menülerde gezinme kolay ve hızlı olmalıdır.
- **UR8:** Favorilere eklenen öğelere hızlı erişim sağlanmalıdır.
- **UR9:** Not düzenleme sırasında tüm işlemler gecikmesiz ve doğru çalışmalıdır.
- **UR10:** Klavye kısayolları tüm temel işlemler için desteklenmelidir.

4. Sistem Gereksinimleri

4.1 Sistem Kısıtlamaları ve Varsayımlar

- **Kısıtlamalar:**
  - Uygulama, bulut hizmetleri ile senkronizasyon için sürekli bir internet bağlantısına ihtiyaç duyar. Eğer offline kullanılacaksa bu ihtiyaç zorunlu değildir.
  - Büyük dosya boyutları için eşitleme süreleri uzayabilir.
  - Düşük bellekli cihazlarda performans sorunları yaşanabilir.
- **Varsayımlar:**
  - Kullanıcıların kullandığı bulut depolama hizmetleri (ör. Dropbox, OneDrive) hesaplarının aktif ve doğru yapılandırılmış olduğu varsayılır.
  - Kullanıcılar, minimum sistem gereksinimlerini karşılayan cihazlar kullanır.

4.2 Donanım Gereksinimleri

- **Masaüstü Bilgisayar:**
  - İşletim Sistemi: Windows 10, macOS 10.13 veya daha yenisi, Linux (Ubuntu 20.04 veya daha yenisi).
  - Bellek: Minimum 4 GB RAM.
  - Depolama: Minimum 500 MB boş disk alanı.
- **Mobil Cihaz:**
  - İşletim Sistemi: iOS 13.0 veya daha yenisi, Android 8.0 veya daha yenisi.
  - Depolama: Minimum 200 MB boş alan.

4.3 Yazılım Gereksinimleri

- Node.js: 14.0 veya üzeri.
- Bulut depolama için entegrasyon desteği (Dropbox, OneDrive, Google Drive).
- Devbox sanal ortamı.
- Yarn kütüphanesi.

5. Veri Modelleri ve Veritabanı Tasarımı

5.1 Veri Modelleri

Sistemde kullanılacak temel veri modelleri aşağıdaki gibidir:

- Not Modeli:
  - id (integer): Notun benzersiz kimliği.
  - title (string): Notun başlığı.
  - body (text): Notun içeriği.
  - tags (list): Notla ilişkili etiketlerin listesi.
  - created_at (timestamp): Notun oluşturulma zamanı.
  - updated_at (timestamp): Notun son güncellenme zamanı.
- Etiket Modeli:
  - id (integer): Etiketin benzersiz kimliği.
  - title (string): Etiketin adı.
  - notes (list): Bu etiketle ilişkilendirilmiş notların listesi.
- Senkronizasyon Modeli:
  - id (integer): Senkronizasyon işleminin kimliği.
  - status (string): İşlemin durumu (ör. başarılı, başarısız).
  - last_synced_at (timestamp): Son eşitleme zamanı.
- Kullanıcı Modeli:
  - id (integer): Kullanıcının benzersiz kimliği.
  - username (string): Kullanıcı adı.
  - password_hash (string): Şifre hash'i.
  - settings (json): Kullanıcıya özel ayarlar.

5.2 Veritabanı Şeması

Sistemde kullanılacak tablolar ve alanlar aşağıdaki gibidir:

- notes Tablosu:
    - id (INTEGER, PRIMARY KEY): Not kimliği.
    - title (TEXT): Notun başlığı.
    - body (TEXT): Notun içeriği.
    - created_at (TEXT): Notun oluşturulma zamanı.
    - updated_at (TEXT): Notun güncellenme zamanı.
- tags Tablosu:
    - id (INTEGER, PRIMARY KEY): Etiket kimliği.
    - title (TEXT): Etiketin adı.
- notes_tags Tablosu:
    - note_id (INTEGER, FOREIGN KEY): Not kimliği.
    - tag_id (INTEGER, FOREIGN KEY): Etiket kimliği.
- users Tablosu:
    - id (INTEGER, PRIMARY KEY): Kullanıcı kimliği.
    - username (TEXT): Kullanıcı adı.
    - password_hash (TEXT): Şifre hash'i.
- sync_logs Tablosu:
    - id (INTEGER, PRIMARY KEY): İşlem kimliği.
    - status (TEXT): İşlemin durumu.
    - last_synced_at (TEXT): Son eşitleme zamanı.

5.3 Veri Akış Diyagramı

Veri akış diyagramı (VAD), sistemdeki veri akışını görsel olarak gösterir:

- Kullanıcı, yeni bir not oluşturur ve kaydeder.
- Sistem, notu veritabanına kaydeder ve gerekirse bulut hizmetlerine eşitler.
- Kullanıcı, etiket ekleyerek notlarını organize eder.
- Kullanıcı, arama ve filtreleme işlemleriyle notlarına erişir.
- Sistem, yapılan tüm işlemleri loglar ve güvenliği sağlamak için kontrol mekanizmaları uygular.

5.4 Veri Tutarlılığı ve Bütünlük Kriterleri

- **Veri Doğrulama:** Kullanıcı girişleri doğru formatta olmalı ve ilgili sınırlar içinde olmalıdır.
- **Veri Bütünlüğü:** Veritabanındaki veriler ilişkisel bütünlük kurallarına uygun olmalıdır.
- **Veri Yedekleme:** Düzenli veri yedekleri alınarak veri kaybı önlenmelidir.

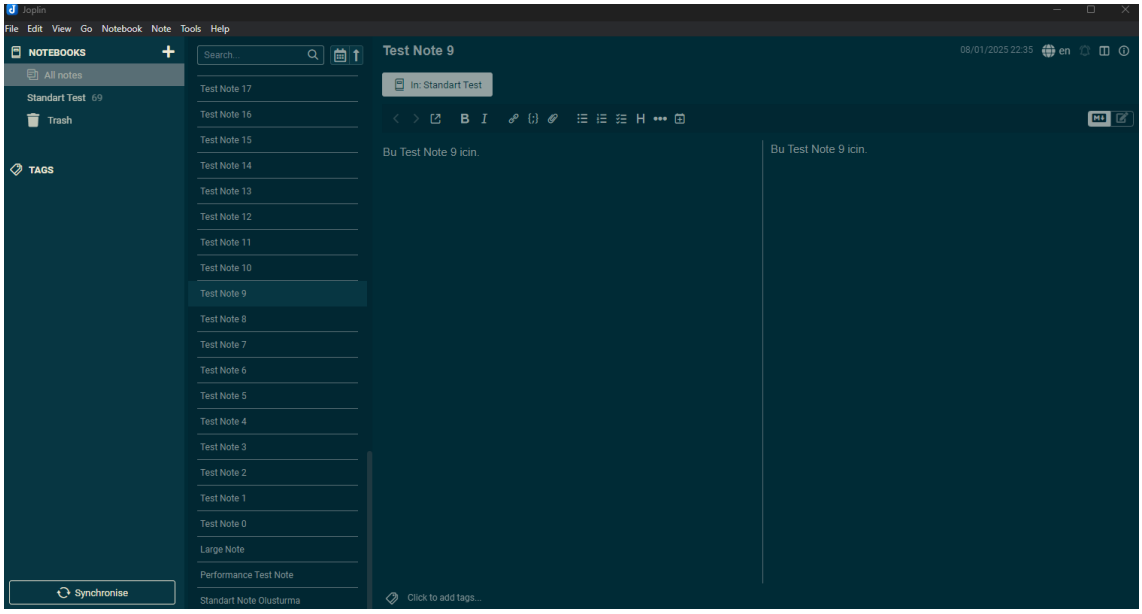
6. Kullanıcı Arayüzü Tasarımı

6.1 Arayüz Tasarımının Genel İlkeleri

Arayüz tasarımında aşağıdaki ilkeler gözetilecektir:

- Basitlik: Arayüz karmaşık olmamalı, kullanıcı kolaylıkla anlayıp kullanabilmelidir.
- Tutarlılık: Aynı işlevler, sistem genelinde tutarlı bir şekilde sunulmalıdır.
- Sezgisellik: Kullanıcılar arayüzü kolayca anlayıp tahmin edebilmelidir.
- Erişilebilirlik: Tüm kullanıcıların ihtiyaçları karşılanmalıdır.
- Verimlilik: Kullanıcılar, işlemlerini hızlı ve etkili bir şekilde tamamlayabilmelidir.

Arayüz örneği şekil 2'de görülebilir.



Şekil 1. Joplin ana ekranı

6.2 Ekran Düzenleri ve Kullanıcı Etkileşimleri İçin Prototipler

Sistem için belirlenen temel ekran düzenleri ve prototipler aşağıdaki gibidir:

- Not Yönetim Ekranı: Kullanıcı, not oluşturabilir, düzenleyebilir ve silebilir.
- Etiket Yönetim Ekranı: Etiketler eklenebilir, düzenlenebilir ve notlarla ilişkilendirilebilir.
- Senkronizasyon Durumu: Kullanıcı, senkronizasyon işlemlerinin durumunu görebilir.
- Arama ve Filtreleme Ekranı: Kullanıcı, notlar içinde anahtar kelime ve etiket filtrelemesi yapabilir.
- Veri Yönetimi Ekranı: Kaydedilen notlar ve eşitleme logları görüntülenebilir.,

7. Sistem Mimarisi Tasarımı

7.1 Sistem Mimarisi

Sistem, aşağıdaki katmanlı mimariyi takip eder:

- Sunum Katmanı (Presentation Layer): Kullanıcı arayüzü bileşenlerini içerir.
- Uygulama Katmanı (Application Layer): İş mantığını ve veri işleme süreçlerini içerir.
- Veri Katmanı (Data Layer): Veritabanı etkileşimlerini yönetir.

7.2 Sistem Bileşlenleri

- **Not Modülü:** Not oluşturma, düzenleme ve silme işlemlerini yönetir.
- **Etiket Modülü:** Etiket ekleme ve kaldırma işlemlerini yönetir.
- **Senkronizasyon Modülü:** Bulut hizmetleriyle veri eşitleme işlemlerini yönetir.
- **Arama Modülü:** Kullanıcının notlar içinde hızlıca arama yapmasını sağlar.

8. Kabul Kriterleri ve Test Planlaması

8.1 Kabul Kriterleri

- **Fonksiyonellik:** Tüm fonksiyonel gereksinimler belirtilen şekilde çalışmalıdır.
- **Performans:** Tüm işlemler belirlenen süre limitleri içinde tamamlanmalıdır.
- **Güvenilirlik:** Sistem, beklenmeyen durumlar karşısında hatasız çalışmalıdır.
- **Kullanılabilirlik:** Arayüz, kullanıcı dostu ve etkili olmalıdır.
- **Veri Doğruluğu:** Sistem verileri doğru şekilde işlemeli ve saklamalıdır.

8.2 Test Planları ve Stratejileri

Test planları, sistemin farklı yönlerini kapsamlı bir şekilde değerlendirmek için tasarlanmıştır.

- **Birim Testleri:**
  - Her bir modül veya fonksiyonun doğru çalıştığını doğrulamak için uygulanır.
  - Örneğin: Not oluşturma, düzenleme ve silme işlemleri için bağımsız testler.
- **Entegrasyon Testleri:**
  - Modüllerin birlikte uyumlu çalışmasını doğrular.
  - Örneğin: Not ve etiket modüllerinin birlikte çalışmasını test etmek.
- **Sistem Testleri:**
  - Sistemin genel fonksiyonel ve performans gereksinimlerini karşıladığını doğrular.
  - Örneğin: 10.000 not üzerinde arama performansının ölçülmesi.
- **Kabul Testleri:**
  - Son kullanıcıların gerçek senaryolarda sistemi test ederek gereksinimlerin karşılandığını doğrulaması.
  - Örneğin: Yeni kullanıcı rehberi, tema değişikliği gibi özelliklerin kullanıcı dostu olup olmadığının kontrolü.
- **Performans Testleri:**

    - Sistem performansını (hız, bellek kullanımı vb.) değerlendirir.
    - Örneğin: Büyük notların 3 saniye içinde yüklenip yüklenmediğinin kontrolü.
- **Güvenlik Testleri:**
    - Zararlı veri girişlerine ve yetkisiz erişimlere karşı sistemin koruma mekanizmalarını doğrular.
    - Örneğin: SQL enjeksiyonu ve XSS saldırılarının engellenmesi.
- **Kullanılabilirlik Testleri:**
    - Kullanıcı arayüzünün sezgisel ve kolay kullanılabilir olup olmadığını değerlendirir.
    - Örneğin: Menü navigasyonunun kolaylığı ve dokunmatik cihazlarda sorunsuz çalışması.
- **Yük Testleri:**
    - Sistem kaynaklarının yoğun yük altında nasıl çalıştığını test eder.
    - Örneğin: Aynı anda birden fazla eşitleme işlemi başlatıldığında sistem davranışı.

8.3 Test Çıktıları

- **Başarı Kriterleri:**
    - Fonksiyonel gereksinimlerin %100 karşılanması.
    - Performans gereksinimlerinin %95 başarı oranıyla sağlanması.
    - Güvenlik testlerinin %100 başarıyla tamamlanması.
- **Hata Yönetimi:**
    - Tespit edilen hatalar ilgili geliştirici ekiplerine bildirilecek ve çözüm süreci takip edilecektir.
    - Hataların çözüldüğü doğrulanana kadar ilgili testler tekrar uygulanacaktır.
- **Raporlama:**
    - Test sonuçları düzenli olarak proje yöneticisine ve diğer ilgili taraflara raporlanacaktır.