ANKARA ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BLM4531 - AĞ TABANLI TEKNOLOJİLER VE UYGULAMALARI PROJE RAPORU

UNINOTES

University Note Sharing Platform

Elem Yalçınkaya - 22290419

Proje Tanıtım Videosu:

https://youtu.be/29iwX1zrUTM

GitHub Linki:

https://github.com/elemyalcinkaya

10.01.2026

ÖZET

Bu rapor; Ağ Tabanlı Teknolojiler ve Uygulamaları dersi kapsamında geliştirilen UniNotes adlı web tabanlı yazılım sisteminin tasarım, geliştirme ve uygulama süreçlerini açıklamak amacıyla hazırlanmıştır. UniNotes, üniversite öğrencilerinin ders notlarını dijital ortamda yükleyebildiği, indirebildiği ve ders/sınıf/dönem bağlamında organize ederek paylaşabildiği bir akademik içerik platformudur. Sistem, notların dağınık kaynaklarda tutulması nedeniyle oluşan erişim zorluklarını azaltmayı ve öğrenciler arasında sürdürülebilir bir bilgi paylaşım ağı oluşturmayı hedeflemektedir.

Geliştirme süreci modern yazılım yaşam döngüsü ilkelerine uygun şekilde sürdürülmüş; kaynak kod yönetimi Git/GitHub üzerinden yürütülmüştür. Ön yüz (Frontend) bileşeni React ve TypeScript ile geliştirilmiş; Vite altyapısı sayesinde hızlı geliştirme ve optimize derleme süreçleri sağlanmıştır. Arka uç (Backend) tarafında {.NET 8 Web API kullanılarak RESTful servis mimarisi uygulanmış; veri erişimi Entity Framework Core aracılığıyla yönetilmiş, Visual Studio kullanılarak geliştirilmiştir. Verilerin kalıcı olarak saklanması için {Microsoft SQL Server tercih edilmiş ve veri bütünlüğü ilişkisel şema tasarımı ile garanti altına alınmıştır. Kullanıcı kimlik doğrulama ve yetkilendirme süreçlerinde {JWT tabanlı güvenli bir mekanizma uygulanmış, parolalar BCrypt ile hashlenmiştir.

UniNotes kapsamında sadece belirlenen güvenli dosya tiplerinin ({PDF, Word, JPEG) yüklenmesine izin verilmiş; ders seçimi ve not yükleme süreçleri veri tabanındaki ders kayıtlarıyla entegre edilerek kullanıcı hataları azaltılmıştır. Bu çalışma, modern web teknolojileriyle geliştirilen, ölçeklenebilir ve sürdürülebilir bir akademik paylaşım sistemine örnek teşkil etmektedir.

İçindekiler

1 GİRİŞ

Üniversite öğrencileri ders sürecinde farklı kaynaklardan notlar üretmekte ve bu notları çoğu zaman kişisel depolama alanlarında dağınık biçimde saklamaktadır. Bu durum, özellikle sınav dönemlerinde veya ders tekrarlarında içeriklere hızlı erişimi zorlaştırmakta; aynı ders için benzer notların tekrar tekrar hazırlanmasına yol açmaktadır. Bu bağlamda geliştirilen UniNotes, öğrencilerin ders notlarını merkezi bir platformda paylaşarak bilgiye erişimi kolaylaştıran ve akademik dayanışmayı artırmayı amaçlayan web tabanlı bir sistemdir.

UniNotes, öğrencilerin platform üzerinde hesap oluşturabildiği, ders seçimi yaparak not yükleyebildiği ve diğer öğrencilerin paylaştığı notları filtreleyerek indirebildiği bir yapıda tasarlanmıştır. Sistem yalnızca PDF, Word ve JPEG dosya türlerine izin vererek içerik standardizasyonu ve güvenlik açısından kontrollü bir yükleme mekanizması sunmaktadır. Ayrıca not içerikleri; ders kodu, sınıf seviyesi ve dönem bilgisi gibi akademik metaverilerle ilişkilendirilerek rastgele dosya paylaşımının ötesinde düzenli ve arama/filtrelemeye uygun bir içerik yönetimi sağlamaktadır.

1.1 UniNotes Nedir

UniNotes; öğrencilerin ders notlarını yükleyip paylaşabildiği, notları sınıf ve dönem bazında filtreleyerek görüntüleyebildiği ve indirebildiği bir akademik paylaşım platformudur. Uygulama, kullanıcıların sistemde tanımlı dersleri seçerek not yüklemesini sağlayarak içeriklerin yapılandırılmış bir veri modeline bağlı kalmasını hedefler. Böylece aranan notlara erişim hızlanır, içerikler daha düzenli bir arşiv mantığıyla yönetilebilir.

1.2 Amaç ve Kapsam

UniNotes projesinin temel amacı, ders notu paylaşımını güvenli ve organize bir yapı altında toplayarak öğrencilerin akademik kaynaklara erişimini kolaylaştırmak ve paylaşım kültürünü teşvik etmektir. Projenin kapsamı aşağıdaki bileşenleri içermektedir:

- **Not Yükleme ve İndirme:** Kullanıcılar PDF, Word ve JPEG formatındaki notlarını sisteme yükleyebilir ve diğer kullanıcıların paylaştığı notları indirebilir.

- **Ders Entegrasyonu:** Not ekleme süreçlerinde kullanıcıların veri tabanında mevcut olan dersler arasından seçim yapması sağlanır; sınıf ve dönem gibi bilgiler notlarla birlikte saklanır.
- **Kullanıcı Hesap Yönetimi:** Kayıt/giriş işlemleri JWT tabanlı kimlik doğrulama ile güvenli şekilde yönetilir.
- **Sayfa Modülleri:** About, Shared Notes, Add Notes, Profile ve Contact Us sayfaları üzerinden kullanıcı akışları gerçekleştirilir.
- **Güvenlik ve Doğrulama:** Dosya türü kısıtları, yetkilendirme kontrolleri, soft delete ve raporlama mekanizmaları ile sistem kontrol edilebilirliği artırılır.

1.3 Çalışmanın Önemi

UniNotes, akademik içeriklerin dağınık ortamlar yerine merkezi ve yönetilebilir bir platformda toplanmasını sağlayarak bilgiye erişimi hızlandırır. Ayrıca modern web geliştirme teknolojileriyle (React, TypeScript, .NET 8, EF Core, SQL Server) uygulanan ölçeklenebilir mimari sayesinde gerçek hayatta kullanılabilir bir sistem altyapısı sunar. Sistem, yalnızca dosya paylaşımı değil; ders bazlı sınıflandırma ve filtreleme ile yapılandırılmış içerik yönetimi yaklaşımını benimser.

1.4 Raporun Yapısı

Bu rapor beş ana bölümden oluşmaktadır. Birinci bölümde projenin amacı, kapsamı ve genel tanımı sunulmuştur. İkinci bölümde geliştirme ortamı, kullanılan teknolojiler ve sistemin teknik yapısı açıklanmıştır. Üçüncü bölümde ön yüz ve arka yüz bileşenleri, veri tabanı tasarımı ve temel modüller ayrıntılı olarak incelenmiştir. Dördüncü bölümde kullanılabilirlik ve teknik değerlendirme yapılmış, güçlü yönler ve kısıtlar ele alınmıştır. Beşinci bölümde ise sonuç ve gelecekteki geliştirme önerileri sunulmuştur.

2 PROJE ORTAMI VE KULLANILAN TEKNOLOJİLER

Bu bölümde UniNotes projesinde kullanılan geliştirme araçları, teknoloji yığını ve sistemin genel mimari yaklaşımı açıklanmaktadır.

2.1 Geliştirme Ortamı

Proje geliştirme sürecinde kaynak kod yönetimi Git/GitHub üzerinden yürütülmüş, ön yüz ve arka yüz bileşenleri bağımsız modüller halinde geliştirilmiştir. Geliştirme ve hata ayıklama süreçlerinde backend için Visual Studio ve frontend için Visual Studio Code gibi araçlar kullanılmış; API testleri için Postman benzeri araçlardan yararlanılmıştır. Veri tabanı yönetimi için SQL Server Management Studio (SSMS) tercih edilmiştir.

2.2 Projenin Genel Teknik Yapısı

UniNotes, istemci-sunucu (Client-Server) mimarisi ile tasarlanmış; istemci tarafı arayüz ile sunucu tarafı servisler HTTP üzerinden RESTful API yaklaşımıyla haberleştirilmiştir. Bu yapı sayesinde ön yüz ve arka yüz geliştirme süreçleri bağımsız yürütülmüş; modülerlik ve sürdürülebilirlik artırılmıştır.

2.3 Kullanılan Yazılım Teknolojileri

- **Frontend:** React, TypeScript, Vite, React Router, TailwindCSS
- **Backend:** .NET 8 Web API, Entity Framework Core, JWT Bearer, BCrypt
- **Veri Tabanı:** Microsoft SQL Server
- **Dokümantasyon:** Swagger/Postman

2.4 Sistem Mimarisi

Sistemde temel işlevler RESTful API uçları üzerinden sunulur. Kimlik doğrulama JWT ile sağlanır ve korumalı uç noktalara erişim token doğrulaması ile gerçekleştirilir. Veri erişim işlemleri EF Core üzerinden yürütülerek kod-veritabanı senkronizasyonu sağlanır.

3 YAZILIM GERÇEKLEME SÜRECİ

Bu bölümde UniNotes projesinin ön yüz, arka yüz ve veri tabanı bileşenleri ile temel modüller açıklanacaktır. Uygulama; not yükleme/indirme, ders seçimi ve kullanıcı yönetimi gibi süreçleri kapsayan modüler bir yapı üzerine inşa edilmiştir.

3.1 Geliştirilen Modüller

UniNotes sistemi aşağıdaki temel modüller etrafında geliştirilmiştir:

- **Kimlik Doğrulama Modülü:** Kayıt/giriş işlemleri JWT ile yönetilir; parolalar hashlenerek saklanır.
- **Not Yönetimi Modülü:** Not ekleme, listeleme, paylaşım durumu (shared), filtreleme ve not detaylarına erişim süreçleri yürütülür.
- **Dosya Yönetimi Modülü:** PDF/DOCX/JPEG dosyalarının yüklenmesi, doğrulanması ve indirilmesi süreçleri yönetilir.
- **Ders Entegrasyonu Modülü:** Ders kayıtları veri tabanından çekilir ve not oluşturma sürecinde seçim alanlarına yansıtılır.
- **İletişim Modülü:** Contact Us sayfasından gelen mesajlar kayıt altına alınır.

3.2 Ön Yüz Tasarımı

UniNotes projesinin ön yüz (Frontend) katmanı, kullanıcıların not paylaşım ve erişim süreçlerini hızlı, anlaşılır ve hataya dayanıklı şekilde gerçekleştirebilmesi amacıyla {bileşen tabanlı (component-based) bir yaklaşımla geliştirilmiştir. Uygulama, {React + TypeScript altyapısı üzerinde kurgulanmış; sayfa geçişleri istemci tarafında yönetilerek Tek Sayfa Uygulama (SPA) deneyimi hedeflenmiştir. Stil katmanında TailwindCSS ve proje genelindeki tutarlılığı sağlayan CSS dosyaları (örn. styles/index.css) kullanılmıştır. Ön yüzün temel hedefleri; notları düzenli ve filtrelenebilir bir yapıda sunmak, kullanıcıya kontrollü dosya yükleme deneyimi sağlamak, oturum ve yetkilendirmeyi güvenli biçimde yönetmek ve arayüz geri bildirimleriyle kullanıcı hatalarını azaltmaktır.

3.2.1 Ön Yüz Proje Yapısı ve Katmanlar

Ön yüz kod tabanı; sorumlulukların ayrıştırılması (Separation of Concerns) prensibine uygun olarak modüllere bölünmüştür. Paylaşılan dizin yapısı incelendiğinde ana bileşenlerin şu şekilde ayrıştığı görülmektedir:

- `src/components/`: Uygulamanın sayfa ve işlev bileşenleri bu dizinde konumlandırılmıştır. `About.tsx`, `SharedNotes.tsx`, `AddNotes.tsx`, `Profil.tsx`, `ContactUs.tsx` gibi sayfalar burada yönetilmektedir. Ayrıca güvenlik, erişim kontrolü için`ProtectedRoute.tsx` bu katmanda yer alır.
- `src/config/`: Ortam bağımlı konfigürasyonların merkezi olarak yönetildiği katmandır. `api.ts` dosyası, API temel adresi ve istek davranışlarının standartlaştırılması için kritik rol üstlenir.
- `src/services/`: Uygulamanın sunucu ile iletişimini soyutlayan servis katmanıdır. `apiService.ts` ve `authService.ts` gibi dosyalar, uç noktalara yapılan çağrıları merkezi hale getirerek bileşen içi karmaşıklığı azaltır.
- `src/styles/`: Global stiller ve temaya ilişkin ortak kurallar bu dizinde toplanır.
- `App.tsx`**,** `main.tsx`: Uygulamanın kök bileşeni, routing ve genel layout yapısı bu dosyalar üzerinden yönetilir.

Bu yapı sayesinde; UI bileşenleri ile API entegrasyonları birbirinden ayrılmış, test edilebilirlik ve bakım kolaylığı artırılmıştır. Özellikle `services/` katmanı, aynı endpoint'in birden fazla sayfada kullanılacağı senaryolarda tekrar eden kodu ortadan kaldırmıştır.

3.2.2 Geliştirilen Ön Yüz Modülleri ve Sayfa İşlevleri

UniNotes ön yüzü, kullanıcı senaryolarına göre modüler bir şekilde geliştirilmiş ve her modül birincil bir iş ihtiyacını karşılayacak şekilde yapılandırılmıştır.

3.2.2.1 Kimlik Doğrulama ve Oturum Modülü (Login/Register)

Bu modül; `Login.tsx` ve `Register.tsx` bileşenleri üzerinden kullanıcı kayıt ve giriş akışlarını yürütür. Kullanıcıdan alınan e-posta/parola gibi bilgiler, istemci tarafında temel doğrulamalardan geçirildikten sonra `authService.ts` aracılığıyla arka uca iletilir.

Başarılı giriş sonrası sistemin token tabanlı çalışması nedeniyle istemci tarafında kullanıcı oturumu bir JWT ile temsil edilir. Token; sonraki isteklerde yetkilendirme başlığına eklenmek üzere saklanır ve oturum sürekliliği sağlanır. Bu yaklaşım, sunucu tarafında session yönetimi ihtiyacını azaltarak ölçeklenebilir bir oturum mimarisi oluşturur.

3.2.2.2 Rota Koruma ve Yetkilendirme Modülü (ProtectedRoute)

Uygulama içerisinde belirli ekranlar yalnızca giriş yapmış kullanıcılar tarafından kullanılabilmektedir (örneğin not ekleme, profil görüntüleme, favoriler vb.). Bu gereksinim, `ProtectedRoute.tsx` bileşeniyle çözülmüştür. Bu bileşen, render edilmek istenen sayfa bileşeninden önce token varlığını ve/veya kullanıcı durumunu kontrol eder. Yetkisiz erişim denemelerinde kullanıcı uygun sayfaya yönlendirilir. Böylece istemci tarafında ilk savunma hattı oluşturulurken, asıl güvenlik arka uç tarafında `[Authorize]` kontrolleriyle tamamlanır.

3.2.2.3 Paylaşılan Notlar Modülü (SharedNotes)

`SharedNotes.tsx`, UniNotes'in en kritik ekranlarından biridir ve uygulamanın "keşfet/indir" kullanım senaryosunu karşılar. Bu sayfada kullanıcılar:

- sistemde paylaşılmış notları listeleyebilir,
- sınıf ve dönem gibi filtreler ile listeyi daraltabilir,
- not kartları üzerinden not başlığı, açıklama/özet
- indirme eylemi ile dosyaları cihazına alabilir,
- uygunsuz içerik için raporlama sürecini başlatabilir.

Filtreleme mekanizması, arayüzde iki dropdown (sınıf ve dönem) üzerinden yürütülerek kullanıcıya aradığı içeriğe hızlı erişim sağlar. Bu ekranda kullanılan kart tasarımı; yoğun içerik listelerinde okunabilirliği artırmak amacıyla bilgi hiyerarşisi gözetilerek tasarlanmıştır. İndirme işlemleri, ilgili dosya kimliği ile backend download endpoint'ine istek atılması şeklinde yürür ve kullanıcı davranışlarının izlenebilmesi için backend tarafında indirme geçmişinin tutulmasına da imkan verir.

3.2.2.4 Not Ekleme Modülü (AddNotes)

`AddNotes.tsx`, kullanıcıların sisteme yeni not eklediği ve dosya yüklediği modüldür. Bu modül, hem kullanıcı deneyimi hem de veri bütünlüğü açısından "kontrollü giriş" yaklaşımıyla tasarlanmıştır. Kullanıcı not eklerken:

- veri tabanında mevcut dersler arasından seçim yapar (ders kodu/ders adı),
- not başlığı ve açıklama gibi metinsel alanları doldurur,
- sınıf seviyesi ve dönem bilgilerini belirler,
- dosya yükleme alanı üzerinden dosyayı sisteme yükler.

Dosya yükleme tarafında sistem yalnızca PDF, Word (DOC/DOCX) ve JPEG uzantılarını kabul edecek şekilde kısıtlanmıştır. Bu kısıt istemci tarafında kullanıcıya anlık uyarı vermek için ön doğrulama ile başlar; ancak asıl doğrulama arka uçta tekrar edilerek güvenlik artırılır. İstemci tarafında dosya yükleme işlemleri genellikle multipart/form-data yapısıyla gerçekleştirildiğinden, `apiService.ts` katmanında bu isteklerin düzgün biçimlendirilmesi sağlanır.

3.2.2.5 Profil ve Kullanıcı İçerikleri Modülü (Profil)

`Profil.tsx` bileşeni, kullanıcının platform içindeki kişisel alanını temsil eder. Bu modül; kullanıcının kendi yüklediği notların listelenmesi, notların paylaşım durumlarının görüntülenmesi ve kullanıcıya ait temel bilgilerin sunulması gibi işlevleri kapsar. Profil ekranı aynı zamanda kullanıcı merkezli bir kontrol paneli görevi görerek, kullanıcıların "benim içeriklerim" perspektifinden sistemi yönetebilmesini sağlar.

3.2.2.6 Raporlama ve Moderasyon Modülü (ReportModal)

Sistemde paylaşılan içeriklerin denetlenebilirliğini artırmak ve topluluk güvenliğini sağlamak amacıyla `ReportModal.tsx` bileşeni geliştirilmiştir. Bu modül, paylaşılan bir notun raporlanması sürecini standartlaştırır. Kullanıcı raporlamayı başlattığında, sistem önceden tanımlı rapor nedenlerini sunar ve seçilen neden backend'e kayıt olarak iletilir. Bu sayede rapor verileri daha sonra yönetim paneli benzeri süreçlerde işlenebilir bir yapı kazanır.

3.2.2.7 Hakkında ve İletişim Modülü (About/ContactUs)

`About.tsx` sayfası sistemin amaç ve kapsamını kullanıcıya sunan bilgilendirici modüldür. `ContactUs.tsx` ise kullanıcıların geri bildirim gönderebilmesini sağlayan iletişim modülüdür. Contact akışında form alanları doğrulanır, gönderim sonucu kullanıcıya bildirilir ve mesajlar backend tarafında kayıt altına alınarak sürdürülebilir bir destek/geri bildirim kanalı oluşturulur.

3.2.3 API İletişimi, Hata Yönetimi ve Kullanıcı Geri Bildirimleri

Ön yüzün backend ile iletişimi `services/apiService.ts` ve konfigürasyon katmanındaki `config/api.ts` üzerinden standardize edilmiştir. Token gerektiren uç noktalarda, isteklerin `Authorization: Bearer <token>` başlığıyla gönderilmesi sağlanır. Ayrıca ağ hataları, doğrulama hataları ve yetkisiz erişim gibi senaryolarda kullanıcıya anlaşılır mesajlar verilecek şekilde hata yakalama (error handling) akışı kurgulanmıştır. Bu yaklaşım, hem kullanıcı deneyimini iyileştirir hem de hata ayıklama süreçlerini kolaylaştırır.

3.2.4 Arayüz Tasarımı ve Duyarlı Yapı

Arayüz; üst menü üzerinden sayfalar arası geçişi kolaylaştıran sade bir navigasyon yapısına sahiptir. Shared Notes ekranı gibi liste yoğun ekranlarda kart tabanlı tasarım ve filtre alanları kullanılmış; böylece kullanıcıların içeriklere hızlı erişmesi hedeflenmiştir. Uygulama genelinde responsive tasarım ilkeleri gözetilerek farklı ekran boyutlarında tutarlı bir deneyim sağlanmıştır.

3.3 Arka Yüz Tasarımı

UniNotes projesinin arka uç (Backend) katmanı, istemci tarafındaki işlemleri güvenli ve tutarlı bir veri modeli üzerinden yönetebilmek amacıyla .NET Web API yaklaşımıyla geliştirilmiştir. Sistem, temel olarak RESTful servis prensiplerine uygun endpoint'ler sağlayarak; kimlik doğrulama, not yönetimi, dosya yükleme/indirme, ders verilerinin sunulması ve raporlama gibi işlevleri merkezi bir API üzerinden sunar. Backend tasarımında; durumsuz (stateless) istek/yanıt modeli benimsenmiş, kimlik doğrulama JWT ile sağlanmıştır. Veri erişim katmanında Entity Framework Core kullanılarak Code First yaklaşımıyla ilişkisel şema yönetimi yürütülmüş; veritabanı tarafında SQL Server tercih edilmiştir.

3.3.1 Arka Yüz Proje Yapısı ve Katmanlaşma

Paylaşılan dizin yapısına göre backend bileşenleri şu ana modüller etrafında konumlandırılmıştır:

- `Controllers/Api/`: Dış dünyaya açılan API uç noktalarının bulunduğu katmandır. İstemciden gelen HTTP istekleri burada karşılanır, temel doğrulamalar yapılır ve iş mantığı servislerine yönlendirilir.
- `DTOs/`: İstemci ile sunucu arasında taşınan veri modellerinin bulunduğu katmandır. Entity modellerinden bağımsız, kontrollü veri sözleşmesi (contract) sağlayarak hem güvenliği hem de API cevaplarının stabilitesini artırır.
- `Models/`: Domain varlıkları (User, Notes, Files, Courses, Tags, Reports, ReportReasons, Favorites, Downloads vb.) burada yer alır. Bu modeller, EF Core üzerinden veritabanı tablolarına karşılık gelir.
- `Data/`: DbContext ve veri erişim yapılandırmalarını içerir. EF Core'un ilişkileri, kısıtları ve tablo eşlemeleri bu katmanda yönetilir.
- `Services/`: Uygulamanın iş mantığının yürütüldüğü katmandır. Örneğin dosya doğrulama, kullanıcı bazlı yetkilendirme kontrolleri, raporlama kuralları, not listeleme filtreleri gibi kurallar burada şekillenir.

- `Migrations/`: Code First yaklaşımıyla üretilen migrasyon dosyaları bu dizinde saklanır. Bu sayede veritabanı şeması kod ile senkron, izlenebilir ve versiyonlanabilir bir hale gelir.

Bu katmanlaşma ile controller katmanı yalnızca HTTP yönlendirme rolünü üstlenirken; veri yönetimi ve kurallar servis/EF katmanında ele alınarak bakım ve genişletme kolaylaştırılmıştır.

3.3.2 API Denetleyicileri (Controllers) ve Sorumlulukları

Backend tarafında API uç noktaları `Controllers/Api` altında toplanmıştır. Dizin yapısına göre sistemde en azından aşağıdaki controller'lar bulunmaktadır:

3.3.2.1 Auth / Kullanıcı İşlemleri (UserController ve/veya Auth Uçları)

Kullanıcı kaydı, giriş ve oturum yönetimi süreçleri bu modül üzerinden yönetilir. Kayıt esnasında parola veritabanına düz metin olarak yazılmaz; BCrypt gibi geri döndürülemez hashleme algoritmalarıyla saklanır. Giriş başarılı olduğunda sunucu, kullanıcı kimliğini temsil eden bir JWT üretir ve istemciye döndürür. Token içerisinde kullanıcıya ait temel claim'ler (Id, Email vb.) bulunur. Bu token, istemcinin sonraki isteklerinde kimlik doğrulama için kullanılır.

3.3.2.2 Notes Yönetimi (NotesController)

Notların eklenmesi, güncellenmesi, silinmesi (soft delete) ve listelenmesi gibi temel CRUD süreçleri NotesController üzerinden yürütülür. UniNotes senaryosunda not, yalnızca bir dosya nesnesi değil; aynı zamanda ders kodu, sınıf seviyesi, dönem, açıklama ve paylaşım durumu gibi metaverilerle tanımlanan bir içerik kaydıdır. Bu controller;

- Shared Notes ekranı için paylaşılmış notların filtrelenebilir biçimde listelenmesi,
- My Notes gibi kullanıcıya özel listeleme uçları ile kullanıcı içeriklerinin ayrıştırılması,
- güncelleme/silme gibi işlemlerde kullanıcı sahipliği kontrolü (yalnızca kendi notunu yönetebilme)

gibi kritik güvenlik ve iş kurallarını taşır.

3.3.2.3 Dosya İşlemleri (FilesController)

UniNotes'in temel değer önerisi not dosyalarının yönetilmesidir. FilesController, dosya yükleme ve indirme süreçlerini yönetir. Dosya yükleme sürecinde:

- istek multipart/form-data ile alınır,
- dosya uzantısı ve MIME tipi kontrol edilerek sadece PDF, Word, JPEG kabul edilir,
- dosya boyutu gibi kısıtlar uygulanarak servis kararlılığı korunur,
- dosya metaverileri (dosya adı, boyut, uzantı, ilgili noteId, userId) veritabanına kaydedilir,
- indirme uçlarında kullanıcıya doğru içerik tipi ile stream edilerek teslim edilir.

Bu yapı, dosya güvenliği açısından önemlidir çünkü istemci doğrulamasına ek olarak sunucu tarafında ikinci bir doğrulama katmanı sağlar.

3.3.2.4 Ders Verisi Uçları (CoursesController)

Kullanıcının not yüklerken ders seçebilmesi için gerekli ders verileri CoursesController üzerinden sunulur. Bu controller; sınıf seviyesi ve dönem gibi parametrelere göre filtrelenmiş ders listesini döndürebilir. Böylece not ekleme süreci "serbest metin" yerine "veri tabanındaki dersler" üzerinden gerçekleşir ve veri tutarlılığı yükselir.

3.3.2.5 Etiketleme ve Kategorilendirme (TagsController)

Notların tür bazında sınıflandırılması (örn. özet, final, vize, lab) gibi senaryolar TagsController ve ilişkili tablolar ile desteklenir. Etiketler seed edilebilir; böylece uygulama ilk kurulumda standart bir sınıflandırma setiyle başlar.

3.3.2.6 Raporlama ve Moderasyon (ReportsController / ReportReasons)

Topluluk tabanlı platformlarda içerik kalitesini korumak için raporlama kritik bir gereksinimdir. UniNotes'te raporlama; rapor nedenlerinin (`ReportReasons`) yönetilmesi ve kullanıcı raporlarının (`Reports`) kaydedilmesi üzerinden yürütülür. Kullanıcı bir notu raporladığında; notId, kullanıcıId ve seçilen rapor nedeni backend tarafında kaydedilir. Böylece ileride yönetici paneli veya moderasyon süreçlerine veri sağlanır.

3.3.2.7 Favoriler ve İndirme Geçmişi (Favorites / Downloads)

Kullanıcı etkileşimini artırmak ve kişiselleştirilmiş deneyim sağlamak için favori sistemi ve indirme geçmişi tutulabilir. Favorites uçları ile kullanıcılar notları favorilere ekleyip çıkarabilir. Downloads tabloları ile hangi kullanıcının hangi dosyayı ne zaman indirdiği izlenebilir; bu hem analitik hem de kullanım geçmişi açısından fayda sağlar.

3.3.2.8 İletişim Modülü (ContactController)

ContactUs ekranından gelen mesajlar ContactController üzerinden alınır ve `ContactMessage` benzeri bir modelle veritabanına yazılır. Bu modül, kullanıcıların sistemle ilgili geri bildirim sunmasına imkan vererek sürdürülebilir iyileştirme yaklaşımına katkı sağlar.

3.3.3 DTO Katmanı ve Veri Sözleşmesi Yönetimi

Backend'de `DTOs/` klasörü, API'nin istemciye doğrudan entity modellerini döndürmesini engelleyen kritik bir katmandır. Örneğin User entity'sinde parola hash alanı bulunabileceğinden, istemciye dönen User DTO yalnızca gerekli alanları içermelidir. Benzer şekilde Notes/Files gibi varlıklarda, ilişkisel alanlar (FK) ve görünmesi istenmeyen iç alanlar DTO ile kontrol edilir. Bu yaklaşım:

- Hassas verilerin istemciye sızmasını önler,
- API cevaplarını sadeleştirir ve stabil hale getirir,
- Sürüm geçişlerinde geriye dönük uyumluluğu artırır.

3.3.4 Veri Erişimi, DbContext ve Migration Yönetimi

Entity Framework Core ile Code First yaklaşımı kullanıldığı için veritabanı şeması doğrudan C# modellerinden üretilir ve `Migrations/` altında versiyonlanır. Paylaşılan migrasyon listeleri, projede tabloların adım adım eklendiğini ve kapsamın genişlediğini göstermektedir (Users, Notes, Files, Courses, Tags, Downloads, Favorites, Reports vb.). Migration yaklaşımı sayesinde:

- Şema değişiklikleri izlenebilir ve geri alınabilir,
- Farklı geliştirme ortamlarında aynı şema kolayca kurulabilir,
- CI/CD süreçlerinde otomatik veritabanı güncellemeleri mümkün hale gelir.

3.3.5 Kimlik Doğrulama, Yetkilendirme ve Güvenlik Yaklaşımı

UniNotes'te güvenlik iki katmanda ele alınır: istemci tarafında rota koruma (`ProtectedRoute`), sunucu tarafında `[Authorize]` ile endpoint koruması. JWT doğrulaması başarısız olduğunda sunucu `401 Unauthorized` döndürür. Ayrıca kullanıcı sahipliği gerektiren işlemlerde (not silme, dosya silme vb.) token içindeki kullanıcıId ile kaynak sahibi eşleştirilir; böylece kullanıcıların başkasına ait içeriği yönetmesi engellenir. Parolalar BCrypt ile hashlenir; düz metin olarak saklanmaz.

3.3.6 Dosya Güvenliği ve Doğrulama Stratejisi

Dosya yükleme işlemleri güvenlik açısından saldırı yüzeyi oluşturabileceğinden, backend tarafında çok katmanlı doğrulama uygulanır:

- Uzantı/MIME kontrolü (PDF, Word, JPEG dışında reddetme),
- Boyut limitleri ve null kontrolü,
- Dosya adlarının normalize edilmesi,
- İndirme uçlarında doğru content-type ile servis edilmesi.

Bu sayede sistemin hem stabilitesi hem de kötü niyetli içerik yüklemelerine karşı direnci artırılmış olur.

3.3.7 Swagger/OpenAPI ile Dokümantasyon

Backend API'sinin test edilebilirliğini ve anlaşılabilirliğini artırmak için Swagger/OpenAPI dokümantasyonu kullanılmıştır. Bu dokümantasyon; endpoint listeleri, request/response örnekleri ve yetkilendirme gereksinimlerini görünür hale getirerek geliştirme ve değerlendirme süreçlerini kolaylaştırır.

3.4 Veri Tabanı Tasarımı

UniNotes projesinde veri tabanı tasarımı; kullanıcı hesap yönetimi, ders-not eşleştirme, dosya saklama metaverileri, kullanıcı etkileşimleri (favori/indirme geçmişi) ve içerik denetimi (raporlama) gibi platformun tüm işlevsel gereksinimlerini karşılayacak şekilde

kurgulanmıştır. Sistem, ilişkisel veri modelinin sağladığı tutarlılık ve bütünlük avantajları nedeniyle Microsoft SQL Server üzerinde tasarlanmış ve Entity Framework Core aracılığıyla Code First yaklaşımı ile yönetilmiştir. Bu yaklaşım sayesinde tablo şemaları, ilişkiler ve kısıtlar C# entity modelleri üzerinden oluşturulmuş; şema değişiklikleri ise `__EFMigrationsHistory` tablosu ile izlenebilir hale getirilmiştir.

UniNotes veri tabanı şeması; birincil anahtarlar (Primary Key), yabancı anahtarlar (Foreign Key) ve bire-çok (One-to-Many) ilişkiler üzerinden; veri tutarlılığını garanti altına alan bir şekilde tasarlanmıştır. Diyagram incelendiğinde sistemin merkezinde Users ve Notes tablolarının yer aldığı, diğer tabloların ise bu çekirdek varlıkların etrafında konumlandığı görülmektedir. Bu tasarım, not paylaşım sisteminin doğal iş akışına uygundur: kullanıcı not üretir (`Notes`), not dosyalarla temsil edilir (`Files`), bu notlar indirilebilir (`Downloads`), favorilenebilir (`Favorites`) ve gerektiğinde raporlanabilir (`Reports` + `ReportReasons`). Ayrıca ders seçimi süreçlerinin standardize edilmesi için `Courses` tablosu ve notların tür bazlı sınıflandırılması için `Tags` tablosu eklenmiştir. Kullanıcı iletişimi ve geri bildirim yönetimi ise `ContactMessages` tablosu üzerinden yürütülmüştür.

3.4.1 Şema Yönetimi ve Migration Stratejisi

Veri tabanı şemasının yaşam döngüsü Entity Framework Core migration mekanizmasıyla yönetilmektedir. `__EFMigrationsHistory` tablosu, uygulanan migration kayıtlarını ve EF sürüm bilgisini tutarak şema değişimlerinin izlenebilirliğini sağlar. Bu yaklaşımın en önemli avantajı, farklı geliştirme ortamlarında (farklı bilgisayarlarda veya farklı ekip üyelerinde) aynı veri tabanı yapısının hızlı ve hatasız şekilde kurulabilmesidir. Böylece şema değişiklikleri manuel SQL yazımı yerine kontrollü, geri alınabilir ve sürümlenebilir bir biçimde uygulanır.

3.4.2 Temel Varlıklar ve Alanların Gerekçelendirilmesi

Bu bölümde diyagramda yer alan tablolar, alanlar ve platformdaki işlevsel karşılıkları detaylı olarak açıklanmaktadır.

3.4.2.1 Users (Kullanıcılar) Tablosu

`Users` tablosu sistemin kimlik ve sahiplik merkezidir. Her kullanıcıya ait benzersiz bir `Id` (Primary Key) bulunur. Kullanıcının `Name` ve `Email` bilgileri, arayüzde profil ve paylaşım

gösterimlerinde kullanılmaktadır. Password alanı uygulamada güvenlik amacıyla düz metin yerine hashlenmiş biçimde saklanacak şekilde tasarlanmıştır. Ayrıca CreatedAt ve UpdatedAt zaman damgaları, kayıtların yaşam döngüsünü izlemek ve log/denetim süreçlerini desteklemek amacıyla eklenmiştir. Kullanıcı tablosu, Notes, Files, Downloads, Favorites ve Reports tabloları ile ilişkili olup; veri tabanında kullanıcı bazlı sahiplik (ownership) modelinin kurulmasını sağlar.

3.4.2.2 Courses (Dersler) Tablosu

Courses tablosu, notların ders bağlamında standardize edilmesi için oluşturulmuştur. Ders kodu (CourseCode) ve ders adı (CourseName) temel tanımlayıcı alanlardır. ClassLevel ve Semester gibi alanlar, ön yüzde (Shared Notes ve Add Notes ekranlarında) filtreleme ve seçim süreçlerini doğrudan destekler. Ayrıca dersin seçmeli olup olmadığını belirtmek için IsElective gibi bir alan bulunması (diyagramda görüldüğü üzere) ders kataloğunu daha doğru modellemeyi sağlar. Bu yapı, notların "rastgele başlıklarla" değil; sistemde tanımlı derslerle eşleşmesini sağlayarak veri kalitesini ve arama/filtreleme doğruluğunu artırır.

3.4.2.3 Notes (Notlar) Tablosu

Notes tablosu UniNotes'in çekirdek içerik varlığıdır. Her not kaydı:

- Id ile benzersiz olarak tanımlanır,
- UserId ile notun sahibine (Users) bağlanır,
- Title ve Summary alanları ile notun açıklayıcı metin içeriğini taşır,
- CourseCode ile notun hangi derse ait olduğunu belirtir,
- CreatedAt ve UpdatedAt ile yaşam döngüsü izlenir,
- IsDeleted ile soft delete yaklaşımı uygulanır.

Soft delete yaklaşımı, notların fiziksel olarak silinmesi yerine mantıksal olarak silinmesini sağlar. Böylece yanlışlıkla silinen içeriklerin geri alınabilmesi ve raporlama/denetim izlerinin kaybolmaması mümkün olur. Ayrıca not ile dosya ilişkisi nedeniyle fiziksel silme, bağlı tablolar üzerinde zincirleme silme riskleri doğuracağından, soft delete veri bütünlüğü açısından güvenli bir çözümdür.

3.4.2.4 Files (Dosyalar) Tablosu

`Files` tablosu, notlara ait gerçek dosya içeriklerinin sistemde nasıl temsil edildiğini belirtir. Her dosya kaydı:

- `Id` ile benzersizleşir,
- `NoteId` ile ilgili nota (`Notes`) bağlanır,
- `UserId` ile dosyanın sahibini (`Users`) işaret eder,
- `Title` gibi kullanıcı dostu bir isim alanı içerir,
- `FileUrl` ile dosyanın fiziksel saklama konumuna referans verir,
- `CreatedAt` ile yükleme zamanı tutulur.

Dosya tablosunun `NoteId` ve `UserId` içermesi iki açıdan kritiktir: (i) dosyanın mutlaka bir not kaydına bağlı olması (yetim dosya oluşumunu engeller), (ii) dosyanın kullanıcı sahipliği kontrolü (başkasına ait dosyanın silinmesi/indirilmesi gibi yetkisiz işlemleri engelleme). UniNotes'in yalnızca PDF, Word ve JPEG formatlarını kabul etmesi, dosya yönetiminde güvenlik ve standardizasyon sağlar; ancak asıl güvenlik doğrulaması uygulama katmanında yapılırken veritabanı bu dosyaların izlenebilirliğini sağlar.

3.4.2.5 Downloads (İndirme Geçmişi) Tablosu

`Downloads` tablosu, kullanıcıların hangi not/dosyaları hangi bağlamda indirdiğini izlemek için tasarlanmıştır. Diyagramda görüldüğü üzere `UserId`, `NoteId` ve `FileId` alanları içerir. Bu tasarım, indirme olayını yalnızca "dosya indirildi" şeklinde değil, aynı zamanda hangi not bağlamında indirildiğini de ifade edecek şekilde kayıt altına alır. Böylece:

- kullanıcı tarafında indirme geçmişi gösterimi mümkün olur,
- sistem kullanım analitiği üretilebilir,
- olası kötüye kullanım (aşırı indirme vb.) senaryolarında denetim yapılabilir.

3.4.2.6 ReportReasons (Rapor Nedenleri) ve Reports (Raporlar) Tabloları

UniNotes'te topluluk kalitesi ve içerik denetimi için raporlama mekanizması tasarlanmıştır. Bu mekanizma iki tabloyla modellenmiştir:

- `ReportReasons`: Raporlama nedenleri sözlüğüdür. `Id` ve `ReasonText` alanlarını içerir. Nedenlerin ayrı tabloda tutulması, rapor girişlerinin standardize edilmesini sağlar (serbest metin yerine kontrollü seçim).
- `Reports`: Kullanıcının belirli bir notu raporladığı kayıt tablosudur. `NoteId` (raporlanan içerik), `UserId` (raporlayan kullanıcı) ve `ReportReasonId` (seçilen neden) ilişkilerini içerir.

Bu tasarım, raporların hem moderasyon için işlenebilir olmasını sağlar hem de rapor istatistiklerinin çıkarılmasına imkan tanır (hangi notlar daha çok raporlanıyor, hangi rapor nedenleri daha sık seçiliyor vb.). Ayrıca rapor kaydı tutulması, sistemin ileride bir yönetici paneli ile genişletilmesi halinde doğrudan kullanılabilir altyapı sunar.

3.4.2.7 ContactMessages (İletişim Mesajları) Tablosu

`ContactMessages` tablosu, kullanıcıların Contact Us sayfası üzerinden gönderdiği geri bildirimleri saklar. `Name`, `Email`, `Subject` ve `Message` alanları, mesajın içeriğini ve iletişim bilgilerini taşır. `IsRead` alanı ise mesajın okunma durumunu temsil ederek basit bir iş akışı yönetimi sağlar. Bu sayede sistem, yalnızca içerik paylaşım platformu değil, kullanıcı ile geliştirici/administrasyon arasında sürdürülebilir bir iletişim kanalı sunar.

3.4.3 Tablolar Arası İlişkiler ve Kardinalite Analizi

Veritabanı diyagramındaki ilişkiler incelendiğinde aşağıdaki temel bağlantılar öne çıkmaktadır:

- **Users → Notes (1-N):** Bir kullanıcı birden fazla not oluşturabilir; her not tek bir kullanıcıya aittir.
- **Notes → Files (1-N):** Bir nota bir veya birden fazla dosya bağlanabilir; her dosya tek bir not kaydına bağlıdır.

- **Users → Files (1-N):** Kullanıcı kendi notu kapsamında birden fazla dosya yükleyebilir.
- **Users → Favorites (1-N) ve Notes → Favorites (1-N):** Favoriler tablosu kullanıcı ve not arasında ilişki kuran ara tablo gibi çalışır.
- **Users → Downloads (1-N), Notes → Downloads (1-N), Files → Downloads (1-N):** İndirme geçmişi hem kullanıcı hem not hem dosya boyutunda izlenebilir.
- **Notes → Reports (1-N) ve Users → Reports (1-N):** Bir not birden fazla kez raporlanabilir; kullanıcı farklı notları raporlayabilir.
- **ReportReasons → Reports (1-N):** Her rapor tek bir rapor nedenine bağlıdır; bir rapor nedeni birçok raporda kullanılabilir.

Bu ilişkisel yapı; veri bütünlüğünü korurken sorgulama esnekliği sunar. Örneğin bir notun kaç kez indirildiği, kaç kişi tarafından favorilendiği veya raporlanma nedenleri gibi veriler doğrudan ilişkisel sorgularla elde edilebilir.

3.4.4 Veri Tutarlılığı, Kısıtlar ve Performans Perspektifi

UniNotes veri tabanı tasarımı, veri tutarlılığını sağlamak amacıyla yabancı anahtar kısıtlarına dayanır. Bu kısıtlar; yetim kayıt oluşumunu önler (örneğin bir `NoteId` olmadan `Files` kaydı oluşturulamaz) ve platform davranışlarını doğal akışa zorlar. Ayrıca sık kullanılan sorgu senaryoları dikkate alındığında (paylaşılan notların listelenmesi, kullanıcıya ait notların çekilmesi, indirme/favori sorguları) performans için `UserId`, `NoteId` ve `CourseCode` gibi alanlarda indeksleme yaklaşımı uygun olacaktır. Bu rapor kapsamında indekslerin fiziksel tanımı ayrıntılandırılmasa da, modelin ilişkisel tasarımı performans optimizasyonuna açık bir zemin sunmaktadır.

Sonuç olarak UniNotes veritabanı; içerik üretimi, içerik erişimi ve içerik denetimi gereksinimlerini kapsayan, genişlemeye uygun ve tutarlı bir ilişkisel model olarak tasarlanmıştır. Şema; sistemin mevcut modüllerini desteklediği gibi, ileride yönetici paneli, gelişmiş arama, etiket bazlı filtreleme ve içerik puanlama gibi ek özelliklere de uygun bir altyapı sunmaktadır.

4 DEĞERLENDİRME

Bu bölümde UniNotes projesi; kullanılabilirlik (UX), arayüz tasarımı (UI), performans, güvenlik ve sürdürülebilirlik kriterleri açısından değerlendirilmiştir. Değerlendirme sürecinde, uygulamanın hedeflediği kullanıcı akışları (not görüntüleme/indirme, not yükleme, ders seçimi, raporlama, favorileme ve iletişim) temel alınmış; istemci-sunucu mimarisinin sağladığı teknik avantajlar ve proje geliştirme sürecinde ortaya çıkan kısıtlar ele alınmıştır. Ayrıca, sistemin ileride genişletilebilmesine yönelik iyileştirme önerileri sunulmuştur.

4.1 Kullanılabilirlik (UX) Değerlendirmesi

UniNotes'in kullanıcı deneyimi, öğrencilerin platform üzerinde en sık gerçekleştireceği işlemlerin hızlı ve hatasız tamamlanmasına odaklanacak şekilde tasarlanmıştır.

- **Gezinim ve Bilgi Mimarisi:** Uygulama içerisinde `About`, `Shared Notes`, `Add Note`, `Profile` ve `Contact Us` sayfalarına üst menü üzerinden doğrudan erişim sağlanmaktadır. Bu sayede kullanıcılar, aradıkları işlevlere minimum tıklama ile ulaşabilmektedir.

- **Shared Notes Akışı:** Paylaşılan notlar ekranında, kullanıcıların notları hızlıca tarayabilmesi için kart tabanlı bir listeleme yaklaşımı benimsenmiştir. sınıf ve dönem filtreleri ile içerik listesi daraltılarak arama yükü azaltılmış, özellikle çok sayıda not bulunan senaryolarda erişim verimliliği artırılmıştır.

- **Add Notes Akışı ve Hata Önleme:** Not ekleme süreci, kullanıcıların dersleri veri tabanındaki kayıtlar üzerinden seçmesine imkan tanıyarak yazım hatalarını ve ders kodu/ders adı tutarsızlıklarını azaltmaktadır. Ayrıca yalnızca PDF, Word ve JPEG dosya tiplerine izin verilmesi, kullanıcıların yükleme sırasında yanlış formatla ilerlemesini engelleyerek sistem kararlılığı ve güvenliğine katkı sağlar.

- **Geri Bildirim Mekanizmaları:** Form gönderimi, giriş işlemleri ve dosya yükleme gibi kritik etkileşimlerde; başarı/hata durumlarının kullanıcıya net biçimde yansıtılması hedeflenmiştir. Bu yaklaşım, kullanıcıların sistem durumunu anlamasını kolaylaştırır ve belirsizliği azaltır.

- **Raporlama ve Topluluk Güvenliği:** Paylaşılan notların raporlanabilmesi, kullanıcıların içerik kalitesini korumaya katkı sağlamasına imkan verir. Bu özellik, platformun yalnızca içerik paylaşan değil aynı zamanda içerik denetimini destekleyen bir yapıya evrilmesini sağlar.

4.2 Arayüz (UI) ve Duyarlı Tasarım Değerlendirmesi

UniNotes arayüzü, modern bir görsel kimlik hedeflenerek tutarlı bir tema yapısı ve sade bir tipografi ile geliştirilmiştir. Sayfa düzenlerinde okunabilirlik ve bilgi hiyerarşisi önceliklendirilmiştir.

- **Tasarım Tutarlılığı:** Sayfalar arasında benzer bileşenlerin (butonlar, kartlar, form alanları) tutarlı şekilde kullanılması, kullanıcıların arayüze alışma süresini kısaltır.
- **Responsive Yaklaşım:** Arayüzün farklı ekran boyutlarına uyumlu olacak şekilde tasarlanması, uygulamanın masaüstü ve mobil kullanım senaryolarında işlev kaybı olmadan çalışmasını destekler.

4.3 Güvenlik Değerlendirmesi

UniNotes, kullanıcı doğrulama ve yetkilendirme süreçlerinde modern web standartlarına uygun mekanizmalar kullanacak şekilde tasarlanmıştır.

- **JWT Tabanlı Kimlik Doğrulama:** Korunan endpoint'lere erişim, token doğrulaması ile sağlanmaktadır. Bu yaklaşım, durumsuz (stateless) mimariyi destekleyerek ölçeklenebilirliği artırır.
- **Yetkilendirme ve Sahiplik Kontrolü:** Kullanıcıların yalnızca kendi notları/dosyaları üzerinde işlem yapabilmesi, veri bütünlüğü ve kullanıcı gizliliği açısından kritik bir gereksinimdir. Bu kontrolün hem ön yüz (ProtectedRoute) hem arka yüz ([Authorize]) tarafında uygulanması güvenliği güçlendirir.
- **Dosya Yükleme Kısıtları:** Dosya türü kısıtlaması ve sunucu tarafı doğrulamalar, kötü niyetli veya sistem dışı içeriklerin yüklenmesini azaltır.

4.4 Performans ve Ölçeklenebilirlik Değerlendirmesi

Uygulamanın istemci-sunucu ayrımı; ön yüz ve arka yüzün bağımsız ölçeklenebilmesini sağlar. Not listeleme, filtreleme ve dosya indirme akışları, gerçek kullanımda en yoğun çalışan senaryolar arasında olduğundan, bu alanlarda verimli sorgu ve veri transferi önemlidir. Bu kapsamda:

- listeleme endpoint'lerinde gereksiz veri taşınmaması (DTO kullanımı),
- dosya indirme süreçlerinde doğru içerik tipiyle stream yaklaşımı,
- sık sorgulanan alanlar (UserId, NoteId, CourseCode) için indekslenebilir bir şema yapısı

performans açısından olumlu bir temel sunmaktadır.

4.5 Kısıtlar ve İyileştirme Önerileri

Proje mevcut haliyle temel gereksinimleri karşılamakla birlikte, ileride geliştirilebilecek noktalar bulunmaktadır:

- **Gelişmiş Arama:** Ders kodu/ders adı dışında anahtar kelime bazlı (full-text) arama eklenebilir.
- **Dosya Önizleme:** PDF için tarayıcı içi önizleme, kullanıcı deneyimini artırabilir.
- **Yönetici Paneli:** Raporların, iletişim mesajlarının ve içeriklerin yönetilebileceği bir admin arayüzü eklenebilir.
- **E-posta Doğrulama / Şifre Sıfırlama:** Hesap güvenliğini artırmak için doğrulama ve kurtarma mekanizmaları eklenebilir.
- **Etiket Bazlı Filtreleme:** Tags tablosu ile notların tür bazlı filtrelenmesi genişletilebilir.

Sonuç olarak UniNotes; kullanıcı odaklı arayüz yaklaşımı, kontrollü içerik yükleme süreci, güvenli kimlik doğrulama altyapısı ve genişlemeye uygun veritabanı şeması ile akademik not paylaşımı senaryosu için işlevsel ve sürdürülebilir bir temel sunmaktadır.

5 SONUÇ

Bu çalışma kapsamında, öğrencilerin akademik içerikleri yapılandırılmış biçimde paylaşabilmesi amacıyla geliştirilen UniNotes platformu başarıyla tasarlanmış ve uygulanmıştır. Proje; React tabanlı modern bir kullanıcı arayüzü ile .NET 8 Web API temelli servis altyapısını bir araya getirerek güvenli ve ölçeklenebilir bir not paylaşım sistemi sunmuştur. SQL Server üzerinde kurulan ilişkisel veri tabanı modeli sayesinde ders, not ve dosya yapıları tutarlı biçimde yönetilmiş; JWT tabanlı kimlik doğrulama ile kullanıcı işlemlerinde güvenlik sağlanmıştır.