GGHub Backend Mimarisi ve Geliştirme Raporu (V1.0)

Tarih: 14 Ağustos 2025

Hazırlayan: Ahmet Demiroğlu

Sürüm Notu: Bu belgenin amacı, backend altyapısını kullanacak herhangi bir geliştirici için eksiksiz bir referans noktası oluşturmaktır.

1. Proje Vizyonu ve Hedefler

GGHub projesi, Türkiye'deki oyun severler için modern, sosyal ve etkileşimli bir platform oluşturma hedefiyle başladı. Başlangıçtaki temel hedef, kullanıcıların oyunları keşfetmesi, yorum yapması ve listeler oluşturmasıydı. Ancak geliştirme süreci boyunca, senin de yönlendirmelerinle, bu vizyon basit bir CRUD uygulamasının çok ötesine geçti. Proje, canlıya alınacak, kullanıcı verisini güvenle saklayacak, moderasyon ve yönetim altyapısına sahip, profesyonel bir sosyal platform olma hedefine evrildi.

2. Mimari Felsefesi ve Teknoloji Yığını

Projenin temelinde, bakımı kolay, genişletilebilir ve test edilebilir bir yapı kurma felsefesi yatmaktadır. Bu amaçla **Katmanlı Mimari (Layered Architecture)** prensipleri benimsenmiştir.

2.1. Katmanlı Mimari

Proje, sorumlulukları net bir şekilde ayıran dört ana katmandan oluşur:

- GGHub.Core **(Çekirdek/Kalp):** Projenin en iç katmanıdır. Sadece temel iş nesnelerini (Entity'ler) ve Enum gibi temel tanımlamaları içerir. Başka hiçbir katmana bağımlılığı yoktur.
- GGHub.Application **(Uygulama/Beyin):** İş mantığının tanımlandığı katmandır. Dış dünyaya "ne yapılacağını" söyleyen sözleşmeleri (Interface'ler) ve veri transferi için kullanılan modelleri (DTO'lar) barındırır.
- GGHub.Infrastructure **(Altyapı/Eller):** Dış dünya ile konuşan tüm "teknik" işlerin yapıldığı katmandır. Veritabanı bağlantısı (EF Core), harici API çağrıları (RAWG), e-posta gönderimi (MailKit) gibi operasyonel görevler burada yer alır.
- GGHub.WebAPI **(Sunum/Yüz):** Projenin dış dünyaya açılan kapısıdır. Gelen HTTP isteklerini karşılar, ilgili servisleri çağırır ve sonucu JSON formatında döndürür.

2.2. Teknoloji Yığını

- **Framework:** .NET 8
- **Veritabanı ORM:** Entity Framework Core 8
- **Veritabanı:** Geliştirme için **SQLite**, üretim hedefi **PostgreSQL**.

- **Kimlik Doğrulama:** JWT (JSON Web Tokens) - AccessToken & RefreshToken
- **Loglama:** Serilog (Yapısal loglama için)
- **E-posta Gönderimi:** MailKit (SMTP üzerinden)
- **Harici Veri Kaynağı:** RAWG Video Games Database API
- **API Dokümantasyonu & Test:** Swashbuckle (Swagger)

3. Veritabanı Mimarisi ve Modelleri (Entity**'ler)**

Projemizin kalıcı hafızası olan veritabanı, EF Core Code-First yaklaşımıyla aşağıdaki Entity'ler kullanılarak tasarlanmıştır.

- **User:** Kullanıcı bilgilerini, rollerini ve gizlilik ayarlarını tutar.
    - *Temel Alanlar:* Id, Username, Email, PasswordHash, PasswordSalt, Role ("User" veya "Admin").
    - *Profil Alanları:* FirstName, LastName, Bio, ProfileImageUrl, DateOfBirth, Status.
    - *Gizlilik & Güvenlik:* MessageSetting, ProfileVisibility, IsEmailVerified, IsDeleted.
- **Game:** RAWG'dan çekilen ve kendi sistemimizde önbelleklenen oyun verilerini tutar.
    - *Temel Alanlar:* Id, RawgId (RAWG'daki ID), Name, Slug, Description, CoverImage, BackgroundImage, Released.
- **Review:** Bir kullanıcının bir oyuna yaptığı yorum ve puanı saklar.
    - *İlişkiler:* UserId (User'a), GameId (Game'e).
    - *İçerik:* Content, Rating (1-10).
- **ReviewVote:** Yorumlara yapılan olumlu/olumsuz oyları tutar.
    - *İlişkiler:* UserId (User'a), ReviewId (Review'e).
    - *İçerik:* Value (+1 veya -1).
- **UserList:** Kullanıcıların oluşturduğu özel oyun listelerini ("Favorilerim" vb.) tanımlar.
    - *İlişkiler:* UserId (User'a).
    - *İçerik:* Name, Description, IsPublic.
- **UserListGame:** Hangi oyunun hangi listede olduğunu belirten ilişki tablosu.
    - *İlişkiler:* UserListId (UserList'e), GameId (Game'e).
- **Follow:** Kullanıcıların birbirini takip etme ilişkisini tutar.
    - *İlişkiler:* FollowerId (User'a), FolloweeId (User'a).

- **UserListFollow:** Kullanıcıların listeleri takip etme ilişkisini tutar.
    - *İlişkiler:* FollowerUserId (User'a), FollowedListId (UserList'e).
- **Message:** Kullanıcılar arası özel mesajları saklar.
    - *İlişkiler:* SenderId (User'a), RecipientId (User'a).
    - *İçerik:* Content, ReadAt, SentAt.
- **UserBlock:** Kullanıcıların birbirini engelleme ilişkisini tutar.
    - *İlişkiler:* BlockerId (User'a), BlockedId (User'a).
- **RefreshToken:** Kalıcı oturumlar için uzun ömürlü token'ları saklar.
- **Notification:** Kullanıcılara gönderilen bildirimleri tutar.
- **ContentReport:** Uygunsuz içerik raporlarını saklar.
- **AuditLog:** Sistemdeki önemli değişikliklerin denetim kaydını tutar.

4. Servis Katmanı ve Sorumluluklar

Application katmanında tanımlanan Interface'ler, sistemin yeteneklerini belirler. Infrastructure katmanı bu yetenekleri hayata geçirir.

- IAuthService**:** Kullanıcı kimlik doğrulama işlemlerini yönetir.
    - *Metotlar:* Register, Login, RefreshTokenAsync, VerifyEmailAsync.
- IGameService**:** Oyun verilerini RAWG'dan getirme ve kendi veritabanımızda yönetme.
    - *Metotlar:* GetGamesAsync (listeleme, filtreleme), GetGameBySlugOrIdAsync (tek oyun detayı), GetOrCreateGameByRawgIdAsync (yorumlama/listeleme için oyunu DB'ye ekleme).
- IProfileService**:** Kullanıcının kendi profil verilerini yönetir.
    - *Metotlar:* GetProfileAsync, UpdateProfileAsync, AnonymizeUserAsync, GetUserDataForExportAsync, gizlilik ayarlarını güncelleme metotları.
- IReviewService**:** Yorumlarla ilgili tüm CRUD ve oylama işlemlerini yönetir.
    - *Metotlar:* CreateReviewAsync, GetReviewsForGameAsync, UpdateReviewAsync, DeleteReviewAsync, VoteOnReviewAsync.
- IUserListService**:** Kullanıcı listeleriyle ilgili CRUD işlemlerini yönetir.
    - *Metotlar:* CreateListAsync, GetListsForUserAsync, GetListByIdAsync, AddGameToListAsync, RemoveGameFromListAsync.
- ISocialService**:** Kullanıcılar arası sosyal etkileşimleri yönetir.

    - *Metotlar:* Kullanıcı/liste takip etme/bırakma, takipçi/takip edilen listeleme, mesaj gönderme, konuşmaları listeleme, kullanıcı engelleme/engeli kaldırma.
- IAdminService**:** Sadece adminlerin yapabileceği işlemleri yönetir.
    - *Metotlar:* GetContentReportsAsync, UpdateReportStatusAsync.
- IReportService**:** İçerik raporlama işlemlerini yönetir.
- ISearchService**:** Platform genelinde arama yapma.
- INotificationService**:** Bildirim oluşturma.
- IAuditService**:** Denetim kaydı oluşturma.
- IEmailService**:** E-posta gönderme.

5. Tamamlanan Modüller ve Mevcut Yetenekler

Yol haritamızdaki 12 ana modülün tamamı için temel veya ileri düzey altyapı kurulmuştur.

1. **Gelişmiş Kullanıcı Profili:** ✔ Tamamlandı
2. **Liste Yönetimi:** ✔ Tamamlandı
3. **Gözlemlenebilirlik & Denetim:** ✔ Tamamlandı
4. **Gelişmiş Güvenlik:** ✔ Tamamlandı
5. **Oran Sınırlandırma:** ✔ Tamamlandı
6. **Gelişmiş Keşif:** ✔ Tamamlandı
7. **İçerik Moderasyonu:** ✔ Tamamlandı
8. **Yönetim Paneli API'si:** ✔ Tamamlandı
9. **Bildirim Altyapısı:** ✔ Tamamlandı
10. **Sosyal Etkileşim:** ✔ Tamamlandı
11. **Gelişmiş Arama:** ✔ Tamamlandı
12. **Gizlilik & Veri Yaşam Döngüsü:** ✔ Tamamlandı

6. API Sözlüğü (Frontend İçin Kılavuz)

Aşağıda, frontend uygulamasının backend ile konuşmak için kullanacağı temel API endpoint'lerinin bir özeti bulunmaktadır. Tüm rotalar küçük harflidir. Yetki gerektiren tüm endpoint'ler Authorization: Bearer {accessToken} başlığı bekler.

Auth Controller (/api/auth**)**

- POST /register: Yeni kullanıcı kaydı. (Public)
- POST /login: Kullanıcı girişi. accessToken ve refreshToken döndürür. (Public)
- POST /refresh: refreshToken kullanarak yeni bir accessToken alır. (Public)
- GET /verify-email: E-posta doğrulama linki. (Public)

Games Controller (/api/games**)**

- GET /: Oyunları listeler. page, pageSize, search, genres, ordering parametrelerini kabul eder. (Public)
- GET /{idOrSlug}: Tek bir oyunun detayını getirir. (Public)

Profile Controller (/api/profile**)**

- GET /me: Giriş yapmış kullanıcının kendi profilini getirir. (Authenticated)
- PUT /me: Giriş yapmış kullanıcının kendi profilini günceller. (Authenticated)
- DELETE /me: Giriş yapmış kullanıcının hesabını anonimleştirir. (Authenticated)
- GET /me/notifications: Kullanıcının bildirimlerini listeler. (Authenticated)
- PUT /me/message-setting: Mesaj gizlilik ayarını günceller. (Authenticated)
- PUT /me/visibility: Profil görünürlük ayarını günceller. (Authenticated)
- GET /me/export-data: Kullanıcının tüm verilerini JSON dosyası olarak indirir. (Authenticated)

Profiles Controller (/api/profiles**)**

- GET /{username}: Belirtilen kullanıcının herkese açık profilini getirir. (Public, gizlilik kurallarına tabidir)
- POST /{username}/follow: Bir kullanıcıyı takip eder. (Authenticated)
- DELETE /{username}/follow: Bir kullanıcıyı takipten çıkar. (Authenticated)
- GET /{username}/followers: Bir kullanıcının takipçilerini listeler. (Public)
- GET /{username}/following: Bir kullanıcının takip ettiklerini listeler. (Public)
- POST /{username}/block: Bir kullanıcıyı engeller. (Authenticated)
- DELETE /{username}/block: Bir kullanıcının engelini kaldırır. (Authenticated)

Reviews Controller (/api/reviews**)**

- POST /: Yeni bir yorum oluşturur. (Authenticated)

- GET /api/games/{rawgGameId}/reviews: Bir oyuna ait yorumları listeler. (Public)
- PUT /{reviewId}: Kullanıcının kendi yorumunu günceller. (Authenticated)
- DELETE /{reviewId}: Kullanıcının kendi yorumunu siler. (Authenticated)
- POST /{reviewId}/vote: Bir yoruma oy verir. (Authenticated)
- POST /{reviewId}/report: Bir yorumu raporlar. (Authenticated)

UserLists Controller (/api/user-lists**)**

- POST /: Yeni bir liste oluşturur. (Authenticated)
- GET /: Kullanıcının kendi listelerini getirir. (Authenticated)
- GET /{listId}: Belirli bir listenin detayını ve içindeki oyunları getirir. (Authenticated)
- POST /{listId}/games: Bir listeye oyun ekler. (Authenticated)
- DELETE /{listId}/games/{gameId}: Bir listeden oyun siler. (Authenticated)
- POST /{listId}/follow: Herkese açık bir listeyi takip eder. (Authenticated)
- DELETE /{listId}/follow: Bir listeyi takipten çıkar. (Authenticated)

Diğer Controller'lar

- AdminController **(**/api/admin**):** Raporları listeleme, durum güncelleme gibi sadece adminlerin erişebileceği endpoint'leri barındırır. (Admin Role Required)
- MessagesController **(**/api/messages**):** Mesaj gönderme, konuşmaları ve konuşma geçmişini listeleme. (Authenticated)
- SearchController **(**/api/search**):** Oyunlar ve kullanıcılar içinde genel arama yapma. (Public)

7. Sonuç ve Frontend'e Geçiş

Bu raporun gösterdiği gibi, elimizde son derece yetenekli, güvenli, yönetilebilir ve tam özellikli bir **GGHub Backend V1.0** bulunmaktadır. API, bir frontend uygulamasının ihtiyaç duyacağı tüm temel ve ileri düzey özellikleri sunmaya hazırdır.

Frontend geliştirme sürecine başlarken bu belge, API'nin nasıl çalıştığını, hangi verileri beklediğini ve ne tür yanıtlar döndürdüğünü anlamak için temel bir kaynak olacaktır.

Backend artık sırtımızı güvenle yaslayabileceğimiz sağlam bir duvardır. Şimdi bu duvarın önüne, kullanıcıların hayran kalacağı o güzel evi, yani frontend'i inşa etme zamanı.