|  |  |
| --- | --- |
|  | **İZMİR BAKIRÇAY ÜNİVERSİTESİ**  **MÜHENDİSLİK VE MİMARLIK FAKÜLTESİ**  **BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ** |

**<Allegro >**

# ****Sistem Gereksinimleri ve Özellikleri Raporu****

2021-2022 GÜZ

BİL 203 & BİL 209 Dönem Projesi

Sürüm 1.1

*Hazırlayan*

200601070 Caner Çakmak

200601015 Özmen Kahveci

İçindekiler

İçindekiler 2

Revizyon Geçmişi 2

1. Giriş 3

1.1 Amaç 3

1.2 Ürün Kapsamı 3

1.3 Kurallar, Tanımlar ve Kısaltmalar 3

1.4 Referanslar 3

1.5 Rapora Genel Bakış 3

2. Genel Tanım 4

2.1 Ürüne Bakış 4

2.2 Ürün İşlevleri 4

2.3 Kullanıcı Sınıfları ve Özellikleri 4

2.4 Çalışma Ortamı 4

2.5 Tasarım ve Uygulama Kısıtlamaları 5

2.6 Kullanıcı Belgeleri 5

2.7 Varsayımlar ve Bağımlılıklar 5

2.8 Riskler 5

3. Harici Arayüz Gereksinimleri 6

3.1 Kullanıcı Arayüzleri 6

3.2 Donanım Arayüzleri 6

3.3 Yazılım Arayüzleri 6

3.4 İletişim Arayüzleri 6

4. Sistem Özellikleri 7

4.1 <Sistem Özelliği 1> 7

4.2 <Sistem Özelliği 2> 7

5. Yazılım 8

6. Sistem Tasarımı 9

6.1 Varlık-İlişki Diyagramı 9

6.2 İlişkisel Cebir İfadeleri 9

6.3 Sınıf Diyagramı 9

6.4 *CRUD* Matrisi 9

6.5 View, Trigger ve Stored Procedure Kullanımı 9

Ek A: SQL İfadeleri 10

Ek B: Ekran Görüntüleri 11

Ek C: To Be Determined List (TBDL) 12

Revizyon Geçmişi

|  |  |  |  |
| --- | --- | --- | --- |
| **Adı** | **Tarihi** | **Değişiklik Nedeni** | **Sürüm** |
| Taslak | 20.11.2021 | Sistem gereksinimleri ve özellikleri raporu şablonunun oluşturulması ve açıklanması | 1.0 |
| Faz 1 | 30.11.2021 | Sistem gereksinimlerinin belirlenmesi | 1.1 |
| Faz 2 | 18.12.2021 | Belli bir seviyeye gelen sistemin açıklanması | 2.0 |
| Final | 02.02.2022 | Projenin tamamlanması ve baştan sona açıklanması | 4.0 |

## Giriş

Bu bölümde önerilen sistem için tanımlayıcı bilgilere yer verilir.

Projemiz web tabanlı bir müzik yönetim uygulamasıdır. Uygulamamızı kullanmak isteyen kişilerin sistemimize üye olup internet bağlantısına sahip olmaları beklenmektedir. Üye olan kullanıcılar uygulamamıza diledikleri zaman sisteme istedikleri müzikleri rahatlıkla yükleyebilmektedir. Üye kullanıcılarımız sisteme giriş yaptıkları takdirde sisteme daha önceden yükledikleri müziklere erişebilir, o müzikleri dinleyebilir veya indirebilirler.

### Amaç

Bu raporda gereksinimleri belirtilen ürünü tanımlayın. İlgili faydalar, amaçlar ve hedefler dahil olmak üzere belirtilen yazılımın ve amacının kısa bir tanımını ifade edin.

Geliştiriciler, proje yöneticileri, pazarlama personeli, kullanıcılar, testçiler ve rapor yazarları gibi raporun amaçlandığı hedef okuyucu türlerini tanımlayın.

Tasarladığımız bu projede günümüzde insanlar tarafından oldukça fazla ilgi gösterilen müzik piyasasında insanların beğendikleri, dinlemekten zevk aldıkları müzikleri sistemimize yükleyerek daha sonra da dinleme şanslarına sahip olmalarını amaçlar. Projemizde İnsanlar sistemimize üye olmak şartıyla beğendikleri müzikleri daha sonra dinlemek için sistemimize eklerler ve diledikleri zaman sistemimize giriş yaparak kaydettikleri müzikleri keyifle dinlerler. Ayrıca en çok dinledikleri müzikleri görebileceklerdir.

### Ürün Kapsamı

Özellikle bu rapor, tüm sistemin yalnızca bir bölümünü veya tek bir alt sistemini açıklıyorsa, bu rapora söz konusu olan ürünün kapsamını açıklayın.

Yazılımı kurumsal hedeflerle veya iş stratejileriyle ilişkilendirin.

Ayrı bir vizyon ve kapsam belgesi mevcutsa, onu referans edin.

Raporumuz bütün projeyi detaylı olarak anlatmakta ve açıklamaktadır. Bu rapordan ayrı herhangi bir rapor bulunmamaktadır.

### Kurallar, Tanımlar ve Kısaltmalar

Özel önemi olan yazı tipleri veya vurgulama gibi bu raporu yazarken izlenen standartları veya tipografik kuralları açıklayın. Örneğin, daha yüksek seviyeli gereksinimler için önceliklerin ayrıntılı gereksinimler tarafından miras alınıp alınmadığını veya her gereksinim ifadesinin kendi önceliğine sahip olup olmayacağını belirtin.

Raporu anlamaya yardımcı olacak ve alana özel terim ve kavramın tanımlarını verin ve rapor içinde kullanılan kısaltmaları alfabetik olarak listeleyin.

HTML = Hypertext Markup Language

CSS = Cascading Style Sheets

HTTP = Hyper Text Transfer Protocol

CRUD = Create Read Update Delete

### Referanslar

Bu raporun atıfta bulunduğu diğer belgeleri veya web adreslerini listeleyin. Bunlar, kullanıcı arayüzü stil kılavuzlarını, sözleşmeleri, standartları, sistem gereksinimleri spesifikasyonlarını, kullanım senaryosu belgelerini veya bir vizyon ve kapsam belgesini içerebilir. Okuyucunun başlık, yazar, sürüm numarası, tarih ve kaynak veya konum dahil olmak üzere her referansın bir kopyasına erişebilmesi için yeterli bilgiyi sağlayın.

Bu rapor şablonu için, “IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications” belgesi referans alınmıştır.

### Rapora Genel Bakış

Bu raporun geri kalanının neler içerdiğini ve nasıl bir bölüm düzeninde olduğunu açıklayın. Raporu okumak için genel bakış bölümlerinden (ikinci bölümden) başlayarak ve her bir okuyucu tipine en uygun bölümlerden geçerek bir sıra önerin.

2. Bölümde projemizin gereksinimlerine ve işlevlerine yer verdik.

3. Bölümde projemizin yazılımına ve arayüzüne yer verdik.

4. Bölümde projemizin özelliklerine yer verdik.

5. Bölümde projemizin yazılım mimarisine yer verdik.

6. Bölümde projemizin diyagramlarına, CRUD matrisine ve sql sorgularına yer verdik.

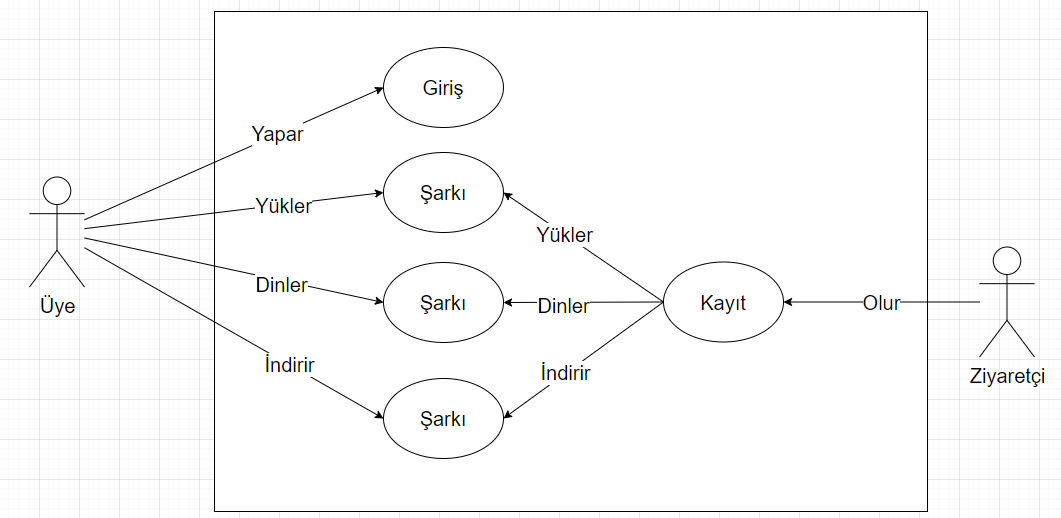
Raporun son sayfalarında projemizin detaylı ekran görüntülerine yer verdik.

## Genel Tanım

Sistemin içeriğini kısaca tanıtın.

Kullanım senaryolarını ve sistem işlemlerini oluşturun. Bunlar paydaşları belirlemenize yardımcı olacaktır. Paydaşların sistemdeki rolleri nelerdir? Onları listeleyin. Data flow, use case, activity, sequence gibi UML diyagramları çizilebilir.

Bu raporda yer alan sistem tasarımı tarafından desteklenen işlevselliğin anlaşılmasına yardımcı olacaksa, bu bölümde projenin arka planını tartışabilirsiniz.

Kullanıcı yapmış olduğumuz müzik yönetim sistemine üye olduktan sonra sistemimizi rahatlıkla kullanabilecektir. Üye olan kişi hesabına internet bağlantısına sahip olduğu her zaman giriş-çıkış yapabilir. Üye, giriş yaptığı takdirde sistemimize istediği şarkıyı yükleyebilir. Yükledikleri şarkıları sisteme giriş yaparak dilediği zaman dinleyebilir veya indirebilir. Ayrıca eklediği şarkıları ne kadar dinlediği bilgisine de sahip olabilirler.

### Ürüne Bakış

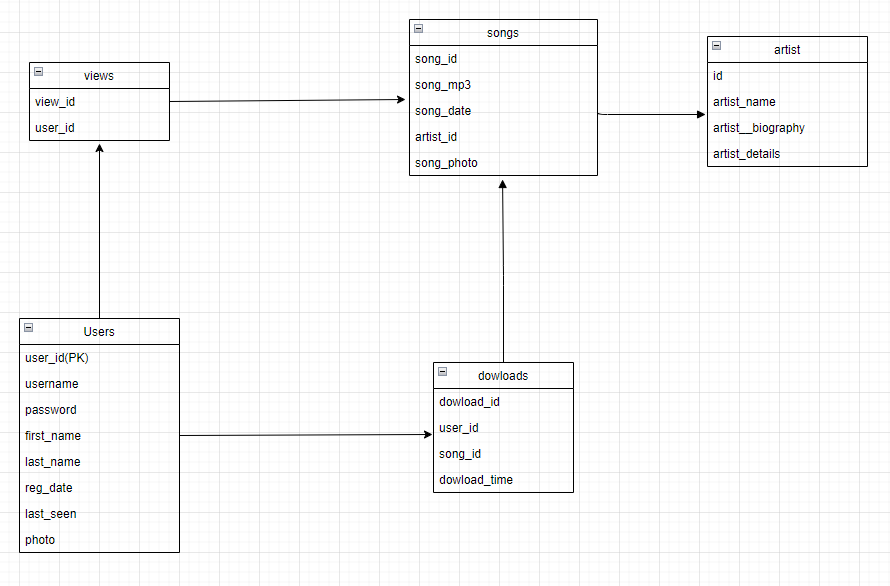
Bu raporda belirtilen ürünün bağlamını ve menşeini tanımlayın. Örneğin, bu ürünün bir ürün ailesinin devam üyesi mi, belirli mevcut sistemlerin yedeği mi yoksa yeni, bağımsız bir ürün mü olduğunu belirtin.

Rapor daha büyük bir sistemin bir bileşenini tanımlıyorsa, daha büyük sistemin gereksinimlerini bu yazılımın işlevselliği ile ilişkilendirin ve ikisi arasındaki arayüzleri tanımlayın. Genel sistemin ana bileşenlerini, alt sistem ara bağlantılarını ve harici arabirimleri gösteren basit bir blok diyagram ya da diyagramlar yardımcı olabilir.

Projemiz herhangi bir başka ürüne veya mevcut sisteme bağlı değildir, bağımsız bir projedir.

### Ürün İşlevleri

Ürünün gerçekleştirmesi gereken veya kullanıcının gerçekleştirmesine izin vermesi gereken ana işlevleri özetleyin. Ayrıntılar Bölüm 4'te verilecektir, bu nedenle burada yalnızca yüksek düzeyde bir özet (madde işareti listesi gibi) gereklidir. İşlevleri, raporun herhangi bir okuyucusu için anlaşılır kılmak için metinsel ya da grafiksel gösterimleri kullanarak düzenleyebilirsiniz. Bir üst düzey data flow diyagramı veya object class diyagramı gibi, ilgili gereksinimlerin ana gruplarının ve bunların nasıl ilişkili olduğunun bir resmi genellikle etkili olabilir. Söz konusu diyagramlar ürünün tasarımını göstermez, sadece işlevler arasındaki mantıksal ilişkiyi ifade eder.



### Kullanıcı Sınıfları ve Özellikleri

Bu ürünü kullanacağını tahmin ettiğiniz çeşitli kullanıcı sınıflarını tanımlayın. Kullanıcı sınıfları, kullanım sıklığına, kullanılan ürün işlevlerinin alt kümesine, teknik uzmanlığa, güvenlik veya ayrıcalık seviyelerine, eğitim düzeyine veya deneyime göre farklılaştırılabilir. Her kullanıcı sınıfının ilgili özelliklerini tanımlayın. Belirli gereksinimler yalnızca belirli kullanıcı sınıflarıyla ilgili olabilir. Bu ürün için en önemli kullanıcı sınıflarını, tatmin edilmesi daha az önemli olanlardan ayırt edin.

Projemizde herhangi bir ayrım gözetmeksizin tek bir kullanıcı sınıfı bulunmaktadır

### Çalışma Ortamı

Donanım platformu, işletim sistemi ve sürümleri ve uyum içinde bir arada var olması gereken diğer yazılım bileşenleri veya uygulamaları dahil olmak üzere yazılımın çalışacağı ortamı tanımlayın.

Projemizin çalışma ortamı için bir internet bağlantısı ve herhangi bir web tarayıcı yeterlidir.

### Tasarım ve Uygulama Kısıtlamaları

Geliştiricilerin kullanabileceği seçenekleri sınırlayacak öğeleri veya sorunları tanımlayın. Bunlar şunları içerebilir: kurumsal veya düzenleyici politikalar; donanım sınırlamaları (zamanlama gereksinimleri, birincil ve ikincil bellek gereksinimleri); diğer uygulamalara arayüzler; kullanılacak belirli teknolojiler, araçlar ve veri tabanları; yedekleme ve kurtarma kısıtları; paralel işlemler; dil gereksinimleri; iletişim protokolleri; Güvenlik Hususları; tasarım sözleşmeleri veya programlama standartları (örneğin, müşterinin kuruluşu teslim edilen yazılımın bakımından sorumlu olacaksa).

Projemiz front-end kısmında html, css ve js içermektedir. Veritabanı olarak MYSQL kullanılmaktadır. Front-end ve veritabanı bağlantısı içinse PHP kullandık. Projenin Geliştirilirken Kullanılan Minimum Sistem:

İşletim sistemi: Windows 10

Ram: 8 gb

İşlemci: i-7 4700 HQ

### Kullanıcı Belgeleri

Yazılımla birlikte teslim edilecek kullanıcı belgeleri bileşenlerini (kullanım kılavuzları, çevrimiçi yardım ve öğreticiler gibi) listeleyin. Bilinen tüm kullanıcı belgeleri teslim biçimlerini veya standartlarını tanımlayın.

Yazılımımızın yanında bir adet “ReadMe.txt” belgesi bulunacak ve bu belgenin içinde yazılım kullanımıyla alakalı bilgiler bulunacaktır.

### Varsayımlar ve Bağımlılıklar

Raporda belirtilen gereksinimleri etkileyebilecek varsayılan etkenleri (bilinen gerçeklerin aksine) listeleyin. Bunlar, kullanmayı planladığınız üçüncü şahıs veya ticari bileşenleri, geliştirme veya işletim ortamıyla ilgili sorunları veya kısıtlamaları içerebilir. Bu varsayımların yanlış olması, paylaşılmaması veya değişmesi durumunda proje etkilenebilir. Ayrıca, başka bir projeden yeniden kullanmayı düşündüğünüz yazılım bileşenleri gibi, başka bir yerde (örneğin, vizyon ve kapsam belgesinde veya proje planında) belgelenmemişlerse, projenin dış etkenlere olan bağımlılıklarını da belirleyin.

Projemizde bootstrap css kütüphanesi ve Jquery kullandığımız için projemiz bunlara bağımlıdır. Ayrıca Projemizde kullandığımız database sayesinde yüklenenler uploads klasörü içinde tutulmaktadır. O yüzden uploads klasörünün kaybolmaması gerekmekte.

### Riskler

*Sistem tasarımı ve önerilen stratejilerle ilgili tüm riskleri tanımlayın.*

* Veri tabanı kaydedilenleri uploads dosyasında tuttuğundan dolayı uploads dosyasında olan bir karışıklık projede çökmelere sebep olabilir.
* Projemizde güvenlik açığımız var.

## Harici Arayüz Gereksinimleri

### Kullanıcı Arayüzleri

Yazılım ürünü ve kullanıcılar arasındaki her bir arabirimin mantıksal özelliklerini tanımlayın. Bu, örnek ekran görüntüleri, herhangi bir GUI standartları veya izlenecek ürün ailesi stil kılavuzları, ekran düzeni kısıtlamaları, her ekranda görünecek standart düğmeler ve işlevler (örn. yardım), klavye kısayolları, hata mesajı görüntüleme standartları ve yakın zamanda. Kullanıcı arabiriminin gerekli olduğu yazılım bileşenlerini tanımlayın. Kullanıcı arayüzü tasarımının detayları ayrı bir kullanıcı arayüzü spesifikasyonunda belgelenmelidir.

Kullanıcılar ekranda en çok dinlenen 10 şarkıyı, en çok dinlenen sanatçıları ve ayrıca son eklenen şarkıları görebileceklerdir. Bu şarkı kapaklarına tıklanınca müzik çalma ekranı gelecektir.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

### Donanım Arayüzleri

Yazılım ürünü ile sistemin donanım bileşenleri arasındaki her arabirimin mantıksal ve fiziksel özelliklerini tanımlayın. Bu, desteklenen cihaz türlerini, verinin yapısını ve yazılım ile donanım arasındaki kontrol etkileşimlerini ve kullanılacak iletişim protokollerini içerebilir.

### Yazılım Arayüzleri

Veritabanları, işletim sistemleri, araçlar, kitaplıklar ve entegre ticari bileşenler dahil olmak üzere bu ürün ile diğer belirli yazılım bileşenleri (ad ve sürüm) arasındaki bağlantıları açıklayın. Sisteme giren ve çıkan veri öğelerini veya mesajları tanımlayın ve her birinin amacını açıklayın. İhtiyaç duyulan hizmetleri ve iletişimin doğasını açıklayın. Ayrıntılı uygulama programlama arabirimi protokollerini açıklayan belgelere bakın. Yazılım bileşenleri arasında paylaşılacak verileri belirleyin. Veri paylaşım mekanizmasının belirli bir şekilde uygulanması gerekiyorsa (örneğin, çok görevli bir işletim sisteminde global bir veri alanının kullanılması), bunu bir uygulama kısıtlaması olarak belirtin.

### İletişim Arayüzleri

E-posta, web tarayıcısı, ağ sunucusu iletişim protokolleri, elektronik formlar vb. dahil olmak üzere bu ürünün gerektirdiği tüm iletişim işlevleriyle ilgili gereksinimleri açıklayın. İlgili mesaj formatını tanımlayın. FTP veya HTTP gibi kullanılacak tüm iletişim standartlarını tanımlayın. Tüm iletişim güvenliği veya şifreleme sorunlarını, veri aktarım hızlarını ve senkronizasyon mekanizmalarını belirtin.

## Sistem Özellikleri

Bu şablon, ürün tarafından sağlanan başlıca hizmetler olan sistem özelliklerine göre ürün için işlevsel gereksinimlerin organize edilmesini gösterir. Bu bölümü, ürününüz için en mantıklı olanı, kullanım durumu, çalışma modu, kullanıcı sınıfı, nesne sınıfı, işlevsel hiyerarşi veya bunların kombinasyonlarına göre düzenlemeyi tercih edebilirsiniz.

### <Kayıt Olma>

Gerçekten “Sistem Özelliği 1” demeyin. Özellik adını birkaç kelimeyle belirtin.

#### Tanım ve Öncelik

Özelliğin kısa bir tanımını yapın ve Yüksek, Orta veya Düşük öncelikli olup olmadığını belirtin. Ayrıca fayda, ceza, maliyet ve risk gibi belirli öncelikli bileşen derecelendirmelerini de dahil edebilirsiniz (her biri en düşük 1 ile en yüksek 9 arasında göreceli bir ölçekte derecelendirilmiştir).

Sitemizde kayıt olmak yüksek önceliklidir. Kullanıcı siteye kayıt olmadan şarkı yükleyemez, indiremez, görüntüleyemez. Öncelikle kullanıcıdan ad, soyad , kullanıcı adı ve şifre alınmaktadır. Bu alınan bilgilerden şifre hariç her biri veri tabanına doğrudan yazılır. Alınan şifre değeri veri tabanına yazılmadan önce şifrelenir ve daha sonra yazılır. Eğer bu bilgilerden kullanıcı adı veri tabanında zaten bulunuyorsa sistem "Kullanıcı adı kullanılmakta." Şeklinde hata vermektedir. Bunun dışında herhangi bir bilgi boş bırakılırsa ayrı olarak sistem boş geçilemez uyarısı da vermektedir.

Fayda = 6

Maliyet = 4

Risk = 8

#### Uyaran/Yanıt Dizileri

Bu özellik için tanımlanan davranışı uyaran kullanıcı eylemlerinin ve sistem yanıtlarının sırasını listeleyin. Bunlar, kullanım durumları ile ilişkili diyalog öğelerine karşılık gelir.

Eğer kullanıcı kayıt olurken herhangi site herhangi bir problem ile karşılaşırsa "Hesabınız oluşturmaya çalışırken bir hata oldu. Lütfen tekrar deneyiniz." Uyarısı vermektedir. Kullanıcı kayıt olurken eksiksiz ve doğru şekilde bilgilerini doldurur ve bir problem olmazsa sitemizin ana sayfasına giriş yapar ve "Hesabınız başarıyla oluşturuldu" uyarısı alır.

#### İşlevsel Gereklilikler

Bu özellikle ilişkili ayrıntılı işlevsel gereksinimleri maddeleyin. Bunlar, kullanıcının özellik tarafından sağlanan hizmetleri yerine getirmesi veya kullanım senaryosunu yürütmesi için mevcut olması gereken yazılım yetenekleridir. Ürünün beklenen hata koşullarına veya geçersiz girdilere nasıl yanıt vermesi gerektiğini dahil edin. Gereksinimler özlü, eksiksiz, açık, doğrulanabilir ve gerekli olmalıdır. Gerekli bilgilerin henüz mevcut olmadığını belirtmek için “TBD”yi (to-be-determined) kullanın. Her gereksinim, bir sıra numarası veya bir tür anlamlı etiketle (REQ-1, REQ-2, …) benzersiz bir şekilde tanımlanmalıdır.

REQ-1 = Ad (boş geçilemez)

REQ-2 = Soyad (boş geçilemez)

REQ-3 = Kullanıcı Adı (Boş geçilemez , veri tabanında aynı iki tane kullanıcı adı olamaz)

REQ-4 = Şifre (Boş geçilemez)

### <Giriş Yapma>

#### Tanım ve Öncelik

Sitemizde giriş yapmak yüksek önceliklidir. Kullanıcı yüklenen müzikleri dinlemek için kendi hesabına girmelidir.

Fayda = 6

Maliyet = 4

Risk = 8

#### Uyaran/Yanıt Dizileri

Kullanıcı giriş yaparken doğru kullanıcı adı ve şifreyi girerse "Başarıyla hesabınıza giriş yapıldı." uyarısı , kulanıcı adını yanlış girerse "Böyle bir kullanıcı bulunamadı." Uyarısı , şifreyi yanlış girerse "Yanlış şifreyi kullandınız." uyarısı alır.

#### İşlevsel Gereklilikler

REQ-1 = Kullanıcı Adı (Boş geçilemez)

REQ-2 = Şifre (Boş geçilemez)

### <Şarkı Ekleme , Silme , ve Düzenleme>

#### Tanım ve Öncelik

Sitemizde şarkı eklemek yüksek önceliklidir. Çünkü zaten sitemiz müzik dinlemek için kurulmuştur.

Şarkı silmek ve düzenlemenin bir önceliği yoktur. Kullanıcının tamemen isteğine ve keyfine kalmaktadır.

Fayda = 9

Maliyet = 6

Risk = 4

#### Uyaran/Yanıt Dizileri

Eğer kullanıcı sitemize şarkıyı başarıyla yüklerse "Yeni şarkı başarıyla eklendi." eğer başarısız olursa "Yeni şarkı eklenirken bir hata oluştu." uyarısı alır.

Kullanıcı şarkıyı düzenlerken başarılı olursa "Şarkı başarılı bir şekilde güncellenmiştir." uyarısı , başarısız olursa "Şarkı güncellenirken bir şeyler ters gitti." uyarısı alır.

Kullanıcı yüklenen şarkıyı silmekte başarılı olursa "Şarkı başarıyla silinmiştir." uyarısı alırken başarısız olursa "Şarkı silinirken bir hata oluştu." uyarısı alır.

#### İşlevsel Gereklilikler

REQ-1 = Şarkı Adı (Boş geçilemez)

REQ-2 = Sanatçı Adı (Boş Geçilemez)

REQ -3 = Şarkı Kapağı

REQ-4 = Şarkı mp3 Dosyası

### < Sanatçı Ekleme >

#### Tanım ve Öncelik

Sitemizde sanatçı eklemek normal önceliklidir. Sanatçı olmadan şarkı eklenebilir. Ama sitenin düzenli ve güzel olması için sanatçı eklemesi tavsiye edilir.

#### Uyaran/Yanıt Dizileri

Eğer kullanıcı sitemize sanatçıyı başarıyla yüklerse "Yeni sanatçı başarıyla eklendi." eğer başarısız olursa "Yeni sanatçı eklenirken bir hata oluştu." uyarısı alır.

#### İşlevsel Gereklilikler

REQ-1 = Şanatçı Adı (Boş geçilemez)

REQ-1 = Sanatçı Hakkında

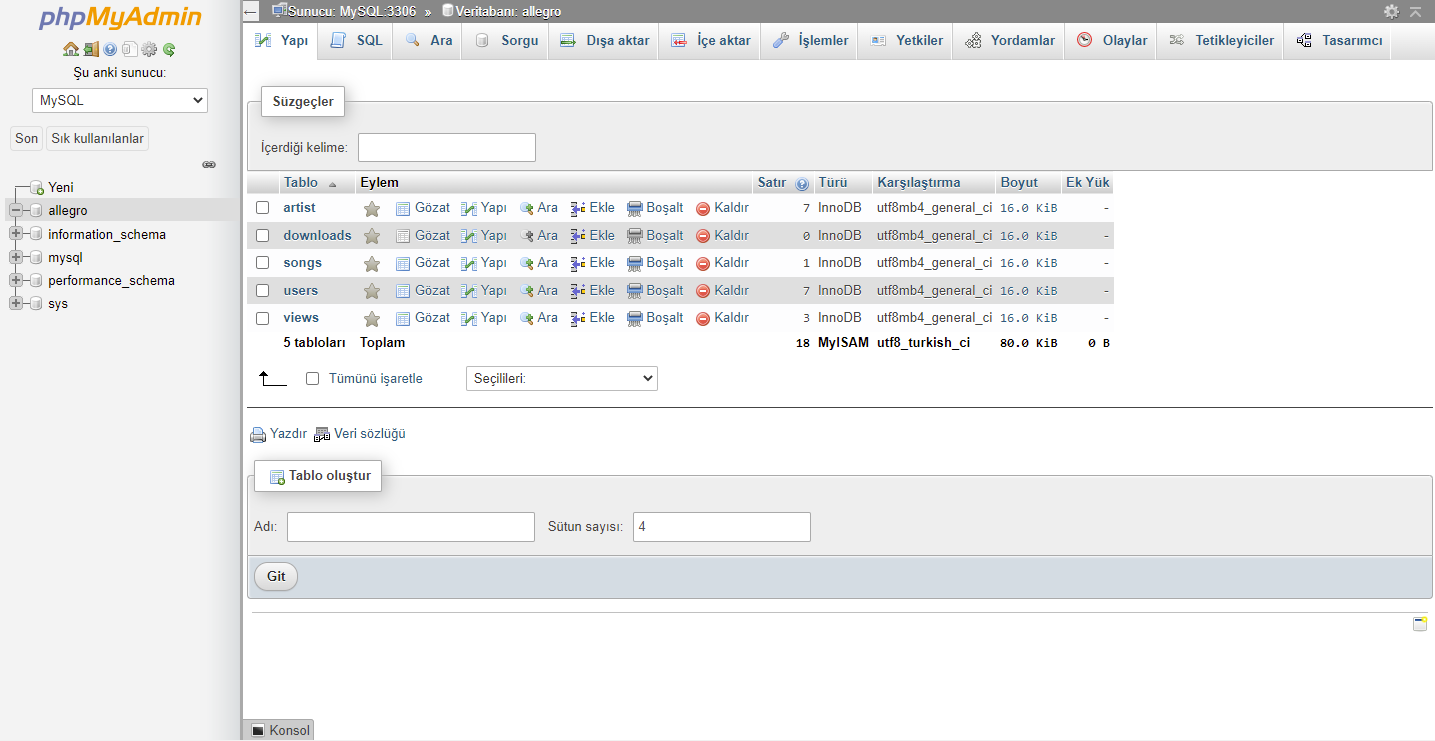
REQ-1 = Sanatçı Fotoğrafı

## Yazılım

Yazılım mimarisiyle ilgili ayrıntıları gösterin. n-tier mimari kullanıyor musunuz? Projenizde nasıl tasarladınız? Yazılım mimarisi diyagramı çizebilirsiniz (Tipik bir yazılım sisteminin kullanıcıları, dış sistemleri, veri kaynakları ve hizmetleri ile nasıl etkileşime girebileceğini gösterir.). Veritabanı ile ön uç (front-end) arasındaki bağlantıyı nasıl sağladınız? Yazılım mimariniz hakkında ayrıntılı bilgi verin.

Projemizi herhangi bir yazılım mimarisi kullanmadan tasarladık.

Projemizin back-end kısmında kullandığımız MySql veritabanını projemizin front-end kısmına bağlamak için php dilini kullandık. Projemizin front-end kısmında ise sayfalar ve uygulamalar için yapı profilleri, bağlantılar, blok alıntılar, paragraflar ve başlıklar oluşturmak için HTML işaretleme dilini, sayfaların görsel denetimi için CSS ve sistemimizin daha kullanışlı olabilmesi için de JS web yazılım dilini Kullandık.

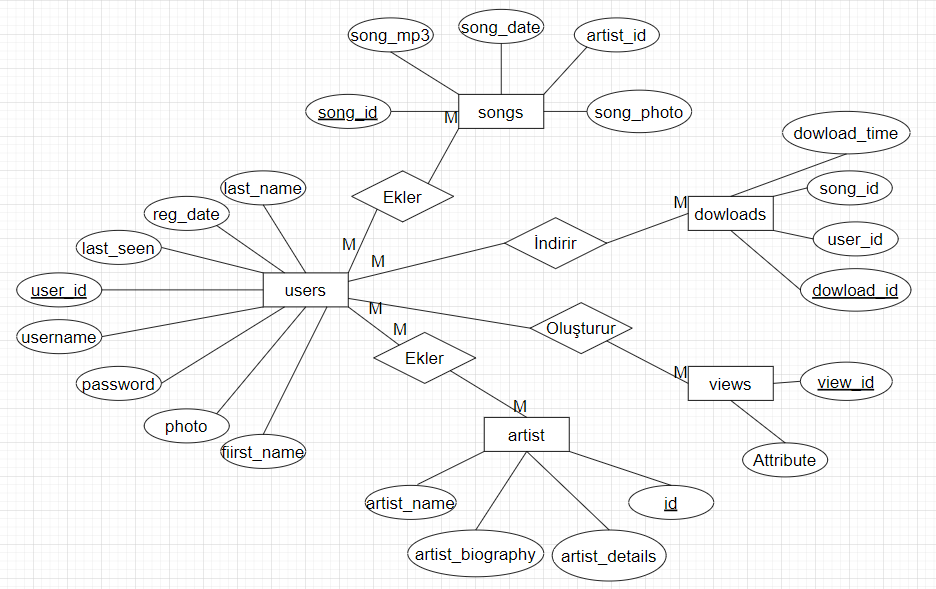
metin, ekran görüntüsü, ekran içeren bir resim

Açıklama otomatik olarak oluşturulduProjemizi xampp server aracılığla Google Chrome tarayısında çalışmaktadır.

## Sistem Tasarımı

### Varlık-İlişki Diyagramı

*Tüm kuralları (tablolar, özellikler, ilişkiler, kardinaliteler, vb.) uygulayarak Varlık İlişki Diyagramını çizin.*

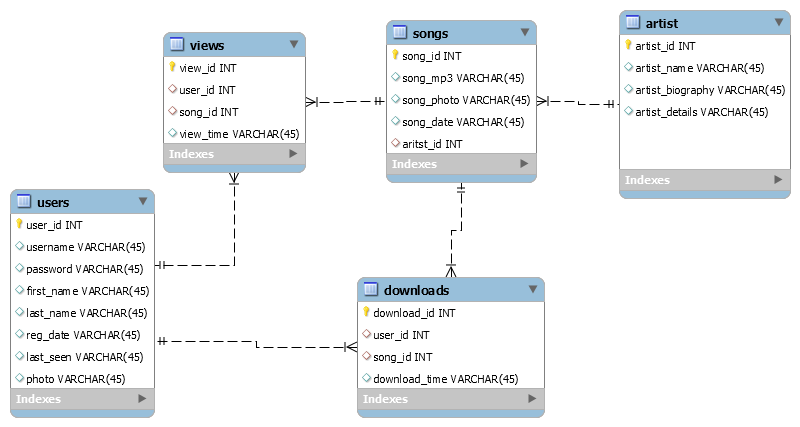


### İlişkisel Cebir İfadeleri

*İlişkisel cebir ifadeleri ve onların SQL cümleleri*

TBD (To Be Determined)

### Sınıf Diyagramı

*UML Class diyagramınızı çizin.*

### *CRUD* Matrisi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tablo-Operasyon** | **CREATE** | **READ** | **UPDATE** | **DELETE** |
| users | X | X |  | X |
| views |  | X | X |  |
| songs | X | X | X | X |
| artist | X | X | X | X |
| dowloads |  | X | X |  |

Aşağıdaki tablodaki gibi tüm ilişkiler için verilerin nasıl korunacağını ve verilere nasıl erişileceğini gösteren CRUD Matrisi (Create, Read, Update, Delete) oluşturun. Tüm CRUD işlemlerini tüm tablolara uyguladınız mı?

### View, Trigger ve Stored Procedure Kullanımı

*Kullandığınız view, trigger ve stored procedure kullanımlarını açıklayınız.*

// view:Top 10 şarkı getirme

function get\_top\_songs($conn){

$songs = get\_all\_songs($conn);

$\_songs = array( );

foreach ($songs as $key => $song) {

$song['view\_count'] = get\_song\_views($conn,$song['song\_id']);

$song['download\_count'] = get\_song\_downloads($conn,$song['song\_id']);

array\_push($\_songs, $song);

}

$i = 0;

$j = 0;

for($j = 0; $j < (count($\_songs) - 1);$j++) {

for($i = 0; $i < (count($\_songs) - 1);$i++) {

if($\_songs[$i]['view\_count'] < $\_songs[$i+1]['view\_count']){

$temp = $\_songs[$i];

$\_songs[$i] = $\_songs[$i+1];

$\_songs[$i+1] = $temp;

}

}

}

return $\_songs;

}

//view: şarkı indirilme sayısı

function get\_song\_downloads($conn,$song\_id){

$sql = "SELECT count(download\_id) AS download FROM downloads WHERE song\_id = {$song\_id}";

$res = $conn->query($sql);

$views = array();

$data = $res->fetch\_assoc();

return $data['download']; }

//view:şarkı görüntülenme sayısı

function get\_song\_views($conn,$song\_id){

$sql = "SELECT count(view\_id) AS view\_count FROM views WHERE song\_id = {$song\_id}";

$res = $conn->query($sql);

$views = array();

$data = $res->fetch\_assoc();

return $data['view\_count']; }

// view :Tüm şarkıları çağırma

function get\_all\_songs($conn){

$sql = "SELECT \* FROM artist,songs

WHERE

songs.aritst\_id = artist.artist\_id

ORDER BY artist\_name ASC";

$res = $conn->query($sql);

$songs = array();

while ($data = $res->fetch\_assoc()) {

array\_push($songs, $data);

}

return $songs; }

//view : tüma sanatçıları çağırma

function get\_all\_artists($conn){

$sql = "SELECT \* FROM artist ORDER BY artist\_name ASC";

$res = $conn->query($sql);

$artists = array();

while ($data = $res->fetch\_assoc()) {

array\_push($artists, $data);

}

return $artists;

}

//trigger: Görüntülenme sayısı arttırma

function record\_view($conn,$song\_id,$user\_id){

$view\_time = time();

$sql = "INSERT INTO `views`

(song\_id,user\_id,view\_time)

VALUES

(

{$song\_id},{$user\_id},'{$view\_time}'

) ";

//trigger: indirme sayısı arttırma

function record\_dowload($conn,$song\_id,$user\_id){

$download\_time = time();

$sql = "INSERT INTO `downloads`

(song\_id,user\_id,download\_time)

VALUES

(

{$song\_id},{$user\_id},'{$download\_time}'

) ";

# Ek A: SQL İfadeleri

Tüm SQL ifadelerine yer verin (komut dosyaları oluşturun ve ekleyin, veritabanındaki temel işlemler, view’ler, karmaşık sorgular, stored procedure’ler, trigger’lar)

*-- phpMyAdmin SQL Dump*

*-- version 5.0.1*

*-- https://www.phpmyadmin.net/*

*--*

*-- Host: 127.0.0.1*

*-- Generation Time: Jul 10, 2020 at 05:47 AM*

*-- Server version: 10.4.11-MariaDB*

*-- PHP Version: 7.2.28*

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

SET AUTOCOMMIT = 0;

START TRANSACTION;

SET time\_zone = "+00:00";

*/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/*;

*/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/*;

*/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/*;

*/\*!40101 SET NAMES utf8mb4 \*/*;

*--*

*-- Database: `allegro`*

*--*

*-- --------------------------------------------------------*

*--*

*-- Table structure for table `artist`*

*--*

CREATE TABLE `artist` (

  `artist\_id` int(11) NOT NULL,

  `artist\_name` varchar(45) DEFAULT NULL,

  `artist\_biography` text DEFAULT NULL,

  `artist\_details` varchar(45) DEFAULT NULL,

  `artist\_photo` text NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

*-- --------------------------------------------------------*

*--*

*-- Table structure for table `downloads`*

*--*

CREATE TABLE `downloads` (

  `download\_id` int(11) NOT NULL,

  `user\_id` int(11) DEFAULT NULL,

  `song\_id` int(11) DEFAULT NULL,

  `download\_time` varchar(45) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

*-- --------------------------------------------------------*

*--*

*-- Table structure for table `songs`*

*--*

CREATE TABLE `songs` (

  `song\_id` int(11) NOT NULL,

  `song\_mp3` text DEFAULT NULL,

  `song\_photo` text DEFAULT NULL,

  `song\_date` text DEFAULT NULL,

  `aritst\_id` varchar(35) DEFAULT NULL,

  `song\_name` varchar(225) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

*--*

*-- --------------------------------------------------------*

*--*

*-- Table structure for table `users`*

*--*

CREATE TABLE `users` (

  `user\_id` int(11) NOT NULL,

  `username` varchar(45) DEFAULT NULL,

  `password` text DEFAULT NULL,

  `first\_name` varchar(45) DEFAULT NULL,

  `last\_name` varchar(45) DEFAULT NULL,

  `reg\_date` varchar(45) DEFAULT NULL,

  `last\_seen` varchar(45) DEFAULT NULL,

  `photo` varchar(45) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

*--*

*-- --------------------------------------------------------*

*--*

*-- Table structure for table `views`*

*--*

CREATE TABLE `views` (

  `view\_id` int(11) NOT NULL,

  `user\_id` int(11) DEFAULT NULL,

  `song\_id` int(11) DEFAULT NULL,

  `view\_time` varchar(45) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

*--*

*--*

*-- Indexes for dumped tables*

*--*

*--*

*-- Indexes for table `artist`*

*--*

ALTER TABLE `artist`

  ADD PRIMARY KEY (`artist\_id`);

*--*

*-- Indexes for table `downloads`*

*--*

ALTER TABLE `downloads`

  ADD PRIMARY KEY (`download\_id`);

*--*

*-- Indexes for table `songs`*

*--*

ALTER TABLE `songs`

  ADD PRIMARY KEY (`song\_id`);

*--*

*-- Indexes for table `users`*

*--*

ALTER TABLE `users`

  ADD PRIMARY KEY (`user\_id`);

*--*

*-- Indexes for table `views`*

*--*

ALTER TABLE `views`

  ADD PRIMARY KEY (`view\_id`);

*--*

*-- AUTO\_INCREMENT for dumped tables*

*--*

*--*

*-- AUTO\_INCREMENT for table `artist`*

*--*

ALTER TABLE `artist`

  MODIFY `artist\_id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=9;

*--*

*-- AUTO\_INCREMENT for table `downloads`*

*--*

ALTER TABLE `downloads`

  MODIFY `download\_id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=11;

*--*

*-- AUTO\_INCREMENT for table `songs`*

*--*

ALTER TABLE `songs`

  MODIFY `song\_id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=13;

*--*

*-- AUTO\_INCREMENT for table `users`*

*--*

ALTER TABLE `users`

  MODIFY `user\_id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=8;

*--*

*-- AUTO\_INCREMENT for table `views`*

*--*

ALTER TABLE `views`

  MODIFY `view\_id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=40;

COMMIT;

*/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/*;

*/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/*;

*/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/*;

# Ek B: Ekran Görüntüleri

*Ana işlemler için kısa açıklamalı arayüzlerinizin ekran görüntüleri*

Giriş ve Kayıt Ekranı

metin içeren bir resim

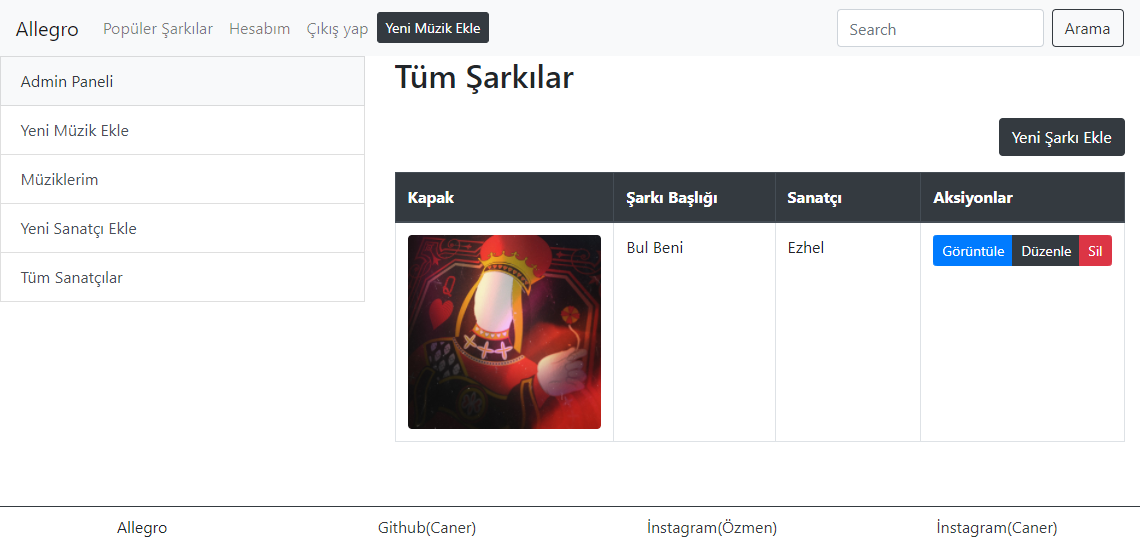
Açıklama otomatik olarak oluşturuldu

Şarkı Ekleme Ekranı

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Tüm Şarkılar Ekranı



Yeni Sanatçı Ekleme

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Tüm Sanatçılar

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

# Ek C: To Be Determined List (TBDL)