PROGRAMLAMA LABORATUVARI 2

3. PROJE

Muhammet Ali BAKINÇ
Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi

Özet

Bu doküman Programlama Laboratuvarı 2 dersi 3. Projesi için çözümümü açıklamaya yönelik oluşturulmuştur. Dokümanda projenin tanımı, çözüme yönelik yapılan araştırmalar, kullanılan yöntemler, proje hazırlanırken kullanılan geliştirme ortamı ve kod bilgisi gibi programın oluşumunu açıklayan başlıklara yer verilmiştir. Doküman sonunda projemi hazırlarken kullandığım kaynaklar ve proje derlenirken dikkat edilmesi gereken hususlar bulunmaktadır.

1. Proje Tanımı

Projede bizden istenen, kullanıcıların şarkı dinleyeceği, Premium kullanıcıların çalma listelerinin takip edilebileceği ve tüm kullanıcıların kendi çalma listelerini oluşturabileceği bir müzik uygulaması için veri tabanı ve masaüstü uygulaması gerçekleştirmektir.

Tasarlayacağımız programda kullanıcıların kayıt olabilmesi ve giriş yapabilmesi gerekmektedir. Her kullanıcı için oluşturulacak üç çalma listesine kullanıcılar şarkı ekleme – çıkarma işlemleri yapabilecektir. Normal kullanıcılar Premium kullanıcıları takip edebilecek ve listelerini görebileceklerdir.

Program işleyişi aşağıdaki gibi olmalıdır:

- Program giriş ekranıyla başlar. Kullanıcı giriş yap veya kayıt ol alanlarını kullanarak bilgilerini girmesi beklenmektedir. Veriler veri tabanından kontrol edilerek ana sayfaya geçilir.
- Ana sayfada türlerine göre tüm şarkılar listelenmiştir. Ayrıca en çok dinlenenler listeleri de bulunmaktadır. Sol menüden kullanıcının çalma listeleri açılabilir ve varsa takip ettiği kullanıcılar görüntülenebilir.
- Her şarkının altında bulunan ekleme butonları ile şarkılar uygun listelere eklenir.
- Liste ekranlarında takip edilen şarkılar görüntülenir. Ekleme – çıkarma yapılabilir. Takip edilen kullanıcıların listelerinden kendi listelerine toplu şarkı eklemesi ve çıkarması yapılabilir.
- Arama çubuğuyla kayıtlı kullanıcılar bulunabilir ve uygun olanlar takip edilebilir.
- Admin giriş yapması halinde tamamen farklı bir ara yüze giriş yapar. Buradan şarkı, albüm ve sanatçı verilerini düzenleyebilir.

2. Araştırmalar ve Yöntem

Projeye, en önemli kısım olan ara yüz tasarımı ile başladım.

Projede nasıl yapılması gerektiğini düşündüren en önemli nokta ara yüz tasarımını çözmekti. Bunun için çeşitli kaynaklardan kodlar inceledim ve videolar izledim. Araştırmalarım sonucu en uygun kütüphanenin JavaFX olduğuna karar verdim. *SceneBuilder* isimli tasarım aracıyla ekranları tasarlamaya başladım. Ekranda görünen komponentleri isimlendirerek kod için erişebilmeyi sağladım. Tasarımı özelleştirebilmek için çeşitli kaynaklardaki CSS kodlarından faydalandım. Tasarım

esnasında zaman zaman kullanımında zorlandığım bazı araçlar için yine internette araştırmalar yaptım. Diğer bir kısım da veri tabanını tasarlamaktı. Bunun için verileri en uygun ve anlaşılır tablolarda tutabilmek için normalizasyon çeşitlerini inceledim. Veri tabanında *MySQL* kullandım. Kodlarla veri tabanını haberleştirebilmek için konektör kullanımını da araştırdım.

3. Geliştirme Ortamı

Projemi Windows sistemde, *Intellij Idea* üzerinde geliştirip derledim. *Java* dilini, *MySQL* veri tabanını ve *JavaFX* kütüphanesini kullandım.

4. Kod Bilgisi

4.1. ER Diyagramı ve Akış Şeması

Son sayfadadır.

4.2. Algoritma

Bu kısımda projenin genel algoritmasına açıklık getireceğim.

4.2.1. Oluşturulan Sınıflar

Aşağıda belirtilen tüm model sınıflarda ortak olarak yapıcı (*constructor*) metotları, sınıflardaki tüm özellikler için *get, set* metotları tanımlanmıştır. Tüm kontrol sınıflarında ekranın üst kısmındaki arama, çıkış yapma, kapatma ve küçültme butonları için fonksiyonlar ortak olarak bulunmaktadır.

Admin Kontrol sınıfı

Adminin giriş yaptığı ara yüzü kontrol eder ve işlemleri gerçekleştirecek fonksiyonları içerir. Ekran oluşturulduğunda öncelikle *initialize* metodu çalışır. Metotta sırasıyla ekrandaki üç tabloya veri tabanından verileri getiren fonksiyonlar çağrılır. Örneğin şarkı tablosu için *insertTableSarki* metodu veri tabanından tüm şarkıları alır ve tabloya ekler. Tablodaki belirli sütunlar düzenlenebilir hale getirilir. Şarkı, albüm ve sanatçı verileri model sınıflarından türetilir ve her satırda güncelle ve sil butonları bulunur. Ekranın alt kısımlarında yeni veri eklemek için alanlar bulunur. Yeni veri eklendikten sonra *refreshTableSarki* ile tablo yenilenir. Yeni veri ekleme sırasında daha önce bulunup bulunmadığı da kontrol edilir.

Arama Kontrol sınıfı

Her ekranın üst kısmında bulunan arama çubuğu ile yapılan kullanıcı arama sonuçlarını ekrana getirir. Arama yapıldığında öncelikle *sonucGetir* fonksiyonu çalışır. Burada alınan metin değerini veri tabanındaki kullanıcılar tablosundan sorgulayarak sonuçları listeye alır. Bu liste *setData* fonksiyonuna gönderilir. Burada liste elemanları sırasıyla Arama Liste Kontrol sınıfına gönderilir ve değerler ilgili alanlara yerleştirildikten sonra ekrana eklenir.

Arama Liste Kontrol sınıfı

Arama sonuçlarındaki kullanıcı listelerindeki her bir satır buradan türetilir. Kullanıcı bilgisini alan setData fonksiyonu verileri satıra yerleştirir. Kullanıcıların abonelik durumlarına göre takip butonunu günceller. Bulunan kullanıcının mevcut kullanıcı tarafından takip edilme durumunu *listeSorgula* fonksiyonu ile veri tabanından sorgular ve *takipEt* veya *listedenCikar* metotlarından uygun olanı butona atar. Kullanıcın takip edilmesi halinde *takipMenuEkle* ile ekranın sol sütunundaki takip edilenler bölümüne kullanıcı eklenir.

Giriş Kontrol sınıfı

Giriş yapılan ana sayfanın kontrolünü yapar. Çalışmaya *initialize* metodu ile başlar. İlk olarak kullanıcının abonelik durumuna göre sponsor reklam görseli düzenlenir. Daha sonra sırasıyla Pop, Jazz ve Klasik türleri için *getList* metodu çağrılır ve veri tabanından alınan şarkılar türlerine göre listelere atanır. Aynı işlem top 10 listeleri için de *getListT10* metodu çağrılır. Son olarak da *getListT10TR* ile Türkiye'de en çok dinlenen listesi sorgulanır. Listelerdeki şarkılar tek tek *SarkiController* sınıfı ile oluşturulur ve ekrana yerleştirilir.

Home Kontrol sınıfı

Ekranın temel iskeletinin kontrolünü sağlar. Sol sütunda bulunan Giriş sekmesi ile *anaSayfa* metodu çağrılır ve giriş sayfası yüklenir. Pop, Jazz ve Klasik listelerine

tıklandığı zaman *listeCagir* metodu ile çalma listesinin olduğu ekran oluşturulur. Sınıf *initialize* ile çalışmaya başlar. Burada *takipListe* metodu ile öncelikle kullanıcının takip ettiği kişiler sorgulanır. Varsa takip ettikleri için takip edilenler alanına listeler eklenir. Ana sayfa olan giriş ekranı da yüklenir.

Liste Kontrol sınıfı

Sınıf *setData* fonksiyonu ile çalışır. Listenin türünü ve kullanıcı bilgilerini parametre alır. Öncelikle *listeGetir* ile çalma listesi veri tabanından alınır. Daha sonra liste gezilerek *SarkiListeController* ile liste elemanları oluşturulur ve ekrana eklenir. Liste adı, sahibi ve kapak resmi gibi bilgiler ekranda güncellenir. Tümünü ekleme ve çıkarma butonu listenin sahibine göre gösterilir. Takip edilen kullanıcılardaki tümünü ekle butonu ile *tumunuEkle* çalışır ve listedeki tüm şarkılar aynı türdeki kendi listesine eklenir. Var olan şarkılar eklenmez. Aynı şekilde *listedenCikar* ile de tümü çıkarılır. Şarkılar *listeSorgula* ile mevcut olma durumları kontrol edilir. Bu işlemler sonucu *listeCagir* fonksiyonu ile sayfa güncellenir.

Şarkı Liste Kontrol sınıfı

Çalma listesindeki şarkı satırlarını oluşturur. Öncelikle setData ile çalışır. Şarkıların kullanıcın kendi listesinde olup olmadığını *listeSorgula* ile kontrol eder. Buna göre takip butonunu günceller. Gerekli *listeyeEkle* ve *listedenCikar* fonksiyonlarını atar. Şarkı bilgilerini ilgili alanlara ekler. Şarkı üzerine tıklanması halinde cal fonksiyonu ile şarkının dinlenme sayısını arttırır ve ekranın alt kısmındaki player bölümüne şarkıyı ekler.

Login Kontrol sınıfı

Programın başlangıç ekranı olan giriş yapma ekranını kontrol eder. İlk olarak *initialize* ile başlar. Burada bazı komponentler düzenlenir. Ekrandaki giriş yap ve kayıt ol alanları için fonksiyonlar bulunur. Giriş yap butonuna tıklandığında *girisButon* çalışır. Öncelikle *login* fonksiyonunu çağırır ve E-posta ve şifre alanlarından alınan veriler veri tabanında sorgulanır. Sonuç true ise *girisGorev* çağrılır. Burada geçilecek ekranın yüklenmesi görevi vardır. Bu görev tamamlanana kadar yükleme çubuğu ekranda belirir. Kayıt ol butonuna tıklandığında aynı şekilde *kayitOl* fonksiyonu çalışır. Öncelikle *kayit* metodunu çağırır. Burada kullanıcının mevcut olup olmadığı kontrol edilir. Yoksa alınan bilgiler ile yeni kayıt oluşturulur. Yeni kullanıcı için çalma listeleri de veri tabanına eklenir. Kullanıcı bilgisi sınıfta daha sonra kullanılmak üzere *static* olarak saklanır.

Şarkı Kontrol sınıfı

Ana sayfada oluşturulan şarkılar bu sınıftan türer. Daha önce Şarkı Liste Kontrol sınıfında bahsedilen fonksiyonlar aynen bulunur.

Takip Liste Kontrol sınıfı

Sol sütundaki takipçi listesi elemanlarını kontrol eder. Açılır menü başlığına kullanıcı adını ekler. Listelere kullanıcı bilgilerini gönderir.

DbHelper sınıfı

Veri tabanı bağlantısının oluşturulduğu sınıftır.

Albüm sınıfı

Veri tabanından alınan sanatçı bilgileri için türetilen model sınıftır. *ID, albumAdi, sanatciID, tarih, tur, silButon ve guncelleButon* değişkenleri bulunur. Admin panelindeki butonlara bağlı *kayitGuncelle* ve *kayitSil* metotları ilgili albümü veri tabanında günceller veya siler.

Sanatçı sınıfı

Veri tabanından alınan sanatçı bilgileri için türetilen model sınıftır. *ID, sanatciAdi, ulke, silButon ve guncelleButon* değişkenleri bulunur. Güncelleme ve silme metotları aynen mevcuttur.

Şarkı sınıfı

Veri tabanından alınan sanatçı bilgileri için türetilen model sınıftır. *ID, sarkiAdi, sanatciID, albumID, sure, tarih, tur, dinlenme, silButon* ve *guncelleButon* değişkenleri bulunur. Güncelleme ve silme metotları aynen mevcuttur.

User sınıfı

Veri tabanından alınan sanatçı bilgileri için türetilen model sınıftır. *ID, kullaniciAdi, email, sifre, abonelik, ulke, adminlik* ve *takipci* değişkenleri bulunur.

4.3. İstatistik

Program kodu boşluklar ve tasarım kodları dahil yaklaşık 4750 satırdan oluşmaktadır. Kullandığım kütüphaneler ve ne için kullandığımız kabaca aşağıdaki gibidir:
javafx
Tasarım elemanlarını kontrol etmek için kullanıldı.
java.sql
Veri tabanı sorguları için kullanıldı.
java.io
Hata yönetimi için kullanıldı.
java.util
Listeler gibi temel Java değişkenleri için kullanıldı.

4.4. Programın Derlenmesi

Programın kaynak kodu 11 kontrol sınıfı, 4 model sınıfı, 1 veri tabanı sınıfı, 1 *CSS* dosyası, 11 *FXML* kodu ve 27 görselden oluşmaktadır. Bu projeyi *Intellij Idea* veya farklı bir Java IDE'si kullanarak derleyebilirsiniz. Derlerken dikkat edilmesi gereken mesele ise programa JavaFX kütüphanesinin eklenmiş olması gereklidir.

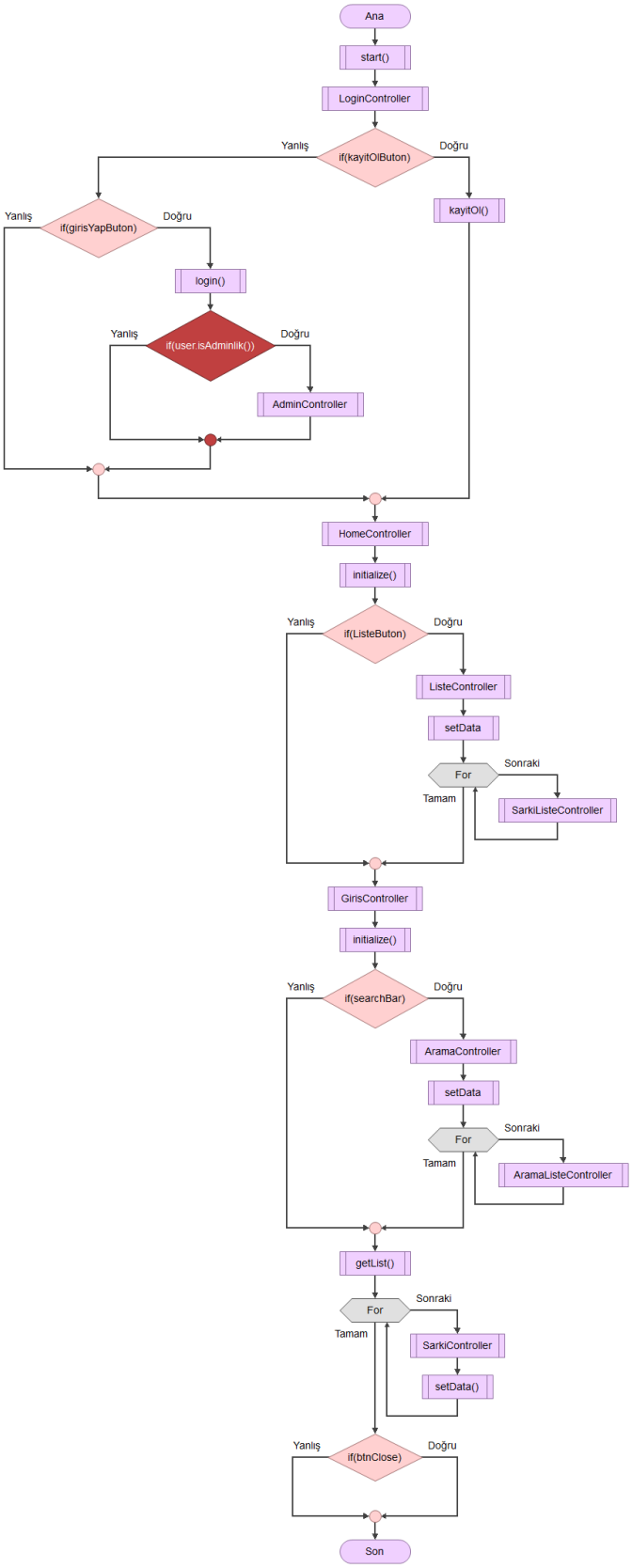
Kaynakça

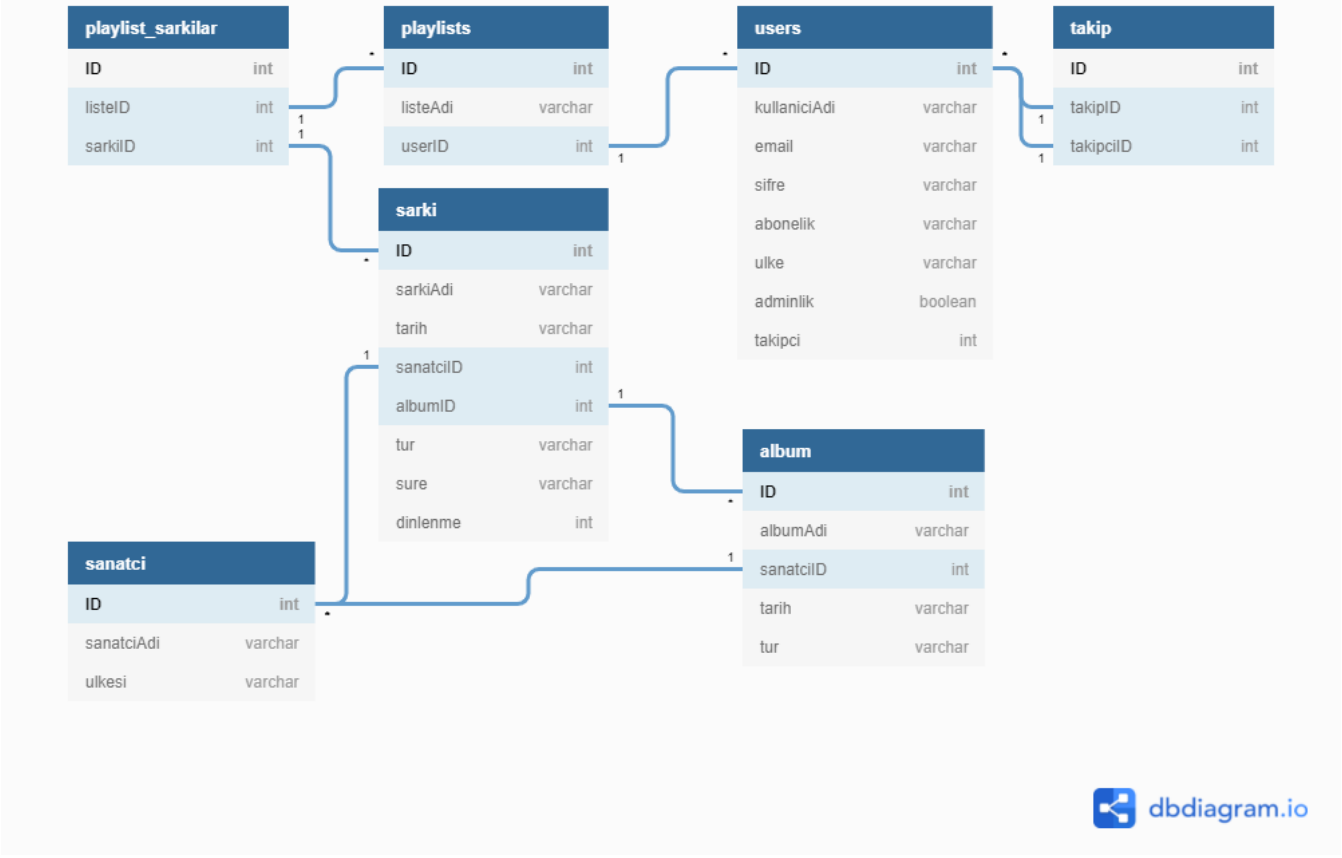
1. JavaFX kütüphanesi kullanım rehberi, https://www.javatpoint.com/javafx-tutorial

2. MySQL kullanım rehberi, https://www.tutorialspoint.com/mysql/index.htm

3. Çeşitli sorunları çözmek için, https://stackoverflow.com/

Akış Diyagramı




playlist_sarkilar
ID int
listeID int
sarkiID int
playlists
ID int
listeAdi varchar
userID int
users
ID int
kullaniciAdi varchar
email varchar
sifre varchar
abonelik varchar
ulke varchar
adminlik boolean
takipci int
takip
ID int
takipID int
takipciID int
sarki
ID int
sarkiAdi varchar
tarih varchar
sanatciID int
albumID int
tur varchar
sure varchar
dinlenme int
album
ID int
albumAdi varchar
sanatciID int
tarih varchar
tur varchar
sanatci
ID int
sanatciAdi varchar
ulkesi varchar
dbdiagram.io