**Bölüm:** Marmara Üniversitesi – Bilgisayar Mühendisliği

**Proje Adı:** Sosyal Medya Arkadaş Takibi

**Projeyi Yapan:** Nuri Can Birdemir - 171421013

**Proje Doküman**

*Anasayfa.java*

Anasayfa’ya Application sınıfından kalıtım alındıktan sonra Start() metodu ile SceneBuilder ile hazırlanmış anasayfa.fxml dosyası sahnede gösterilmektedir. Buradaki “icon” adlı resim açılan penceredeki icon kısmına istenilen resmi eklemektedir.

Bu kısımda birde çıkış işlemi için yapılan bir işlem daha vardır. Pencere kapatılmak istendiğinde emin olup olmadığını kontrol ettiğimiz setOnCloseRequest() metodu çağrılmaktadır. Uyarı mesajı için cikis() adlı metot çağrılmaktadır.

Anasayfanın Contoller’ı ScaneController.java isimli classdır. Anasayfadaki butonların çalışmasını burada sağlanmaktadır.



Figüre . Uygulama Anasayfa Görünümü

*SceneController.java*

Bu class genel olarak anasayfadaki Giriş, Kayıt ve Çıkış işlemlerinin kontrolünü sağlamaktadır.

Classda ilk olarak Giriş ve Kayıt sayfalarındaki TextFieldların tanımı yapılmaktadır.

**loginCode() Metodu**: Bu metot genel olarak kod tutucu görevindedir. Giriş işlemi için yapılacak işlemleri tutmaktadır. Önce Giriş sayfasında olan Kullanıcı adı ve Şifrenin girilen değerlerini alınmaktadır. Alınan değerlerin Kontrolü için GirisController classının içindeki metotlara ulaşmak için bu classın kullanıldığı KullaniciGiris.fxml dosyasına ulaşır ve buradaki Controller olan GirisController.java classına ulaşım sağlanmış olmaktadır. Bu classın içinde olan KullaniciGiris() metoduna ulaşarak buraya girilen kullanıcı adını ve şifresini göndermiş olur ve kontoller yapılarak eğer hata varsa hata mesajlarını bir uyarı penceresinde verilmektedir.

**txtFieldKeyRelesed() Metodu:** Bu metod loginCode() metodunu çağırarak KeyEvent kontrolü ile eğer ENTER tuşuna basılmışsa loginCode() metodunu çalıştırılmaktadır.

**loginIslem() Metodu:** Bu metot da loginCode() metodunu çağrılmaktadır. ActionEvent kontrolu ile butona basıldığında loginCode() metodu çalıştırılmaktadır.

Bu bölümde kontrol edilen hatalar şu şekildedir:

*KullaniciYok:* Girilen kullanıcı adı database de bulunmuyorsa verilen hatadır.

*SifreYanlis:* Girilen kullanıcı adına ait olan şifre ile girilen şifre uyuşmuyorsa verilen hatadır.

**Kayitİslem() Metodu:** Bu metot genel olarak Kayıt butonuna basıldığında yapılacak işlemleri içermektedir.

İlk olarak bir if else ile TextFieldlarda boş girilmeye çalışılırsa getText().trim().isEmpty() ile hata verme kodu çalışmaktadır. Eğer bütün bilgiler alınmışsa program girilen verilerin doğruluğunu kontrol edilmektedir. Sonra Giriş işlemindeki gibi KayitConroller classımıza ulaşıp hatalar ve kayıt işlemi için gerekli metotlara ulaşmak için KullaniciKayit.fxml dosyasına ulaşıp buradaki tüm metotlar alınarak KayitController() metoduna ulaşılacaktır. Burada KayitController metodu boş gönderilmeye çalışılırsa hata mesajı uyarısı şeklindeki boş constructor çalışacaktır (**Overload**). Bu işlemlerden sonra başarılı bir kayıt yapılmışsa uyarı penceresi ile bilgi verip ‘tamam’ denilince anasayfa.fxml ‘e dönülecektir.

Bu bölümde kontrol edilen hatalar şu şekildedir:

*RakamBulundu:* Bu hata mesajı ad ve soyad kısmında rakam girilmeye çalıştığında verilen hatadır.

*KucukYasHatasi:* Yaş kısmına 18’den küçük değer girilmeye çalışılırsa verilen hatadır.

*FazlaDegerGirildi:* Kullanıcı adı kısmına 14 karakterden fazla değer girilmeye çalışırsa verilen hatadır.

*AyniKullaniciAdi:* Eğer database de girilen kullanıcı adı ile aynı kullanıcı adına sahip bir kayıt varsa verilen hatadır.

**Cikis() Metot:** Girilen hesaptan çıkış yapılmak istendiğinde verilen uyarı ve işlemdir.

Bu classın en sonunda da sayfa geçiş kodları bulunmaktadır.



Figure 2. Giriş Sayfası Görünümü

**

Figure 3. Kayıt Sayfası Görünümü

*GirisController.java*

Bu class genel olarak Giriş sayfasının kontrollerini içermektedir. KullaniciGiris.fxml dosyasının kontrol işlemleridir.

İlk olarak KullaniciGiris.fxml deki Nodelar tanımlanmaktadır. Daha sonra Giriş işleminde alınacak değerlerin ataması yapılacak değişkenler tanımlanmaktadır. Bu değişkenler Encapsulation’na uygun tanımlanmaktadır.

**KullaniciGiris() Metodu:** Giriş işlemi kontrolünü yapan genel metottur. Giriş butonuna basılınca aktifleşmektedir.

**GirisBilgi() Metodu:** Girilen kullanıcıyı karşılama bölümüdür. Label’da kullanıcının ismi ile karşılama yapılır.

**Girisİslem() Metodu:** Parametre olarak aldığı kullanıcı adı ve şifre verildikten sonra kullanıcıları tuttuğumuz dosya yolu açılmaktadır ve girilen kullanıcı adında bir dosya var mı diye kontrol edilmektedir. Eğer yoksa *KullaniciYok* hatası dönmektedir. Eğer kullanıcı varsa kullanıcının dosyasını açıp ad, soyad, yaş, kullanıcı adı ve şifresini oluşturduğumuz değişkenlere uygun şekilde atamalar yapılmaktadır. Daha sonra şifrenin doğruluğunu kontrol etmek için girilen şifre ile sistemdeki şifre karşılaştırılmaktadır. Eğer aynı değilseler *SifreYanlis* hatası dönmektedir. Eğer aynı ise Mesaj.java Interfaceden implements ettiğimiz mesaj() metodunu (override) çalışıyor.

Daha sonra KullaniciGiris.fxml de bulunan Arkadaş Ekle, Rastgele Arkadaş, Arkadaş Listele, Arkadaş Sil sayfalarının ilgili butona basılınca gitmesini sağlayan kodlar yazılmıştır. Çıkış işleminde SceneController sınıfından **polymorphism** ile oluşturulan parametreli ‘mesaj’ nesnesi **override** edilerek GirisController’daki mesaj() metodu çağrılması yapılmaktadır.

Figure 4. Kullanıcı Anasayfası Görünümü

*KayitController.java*

Bu class genel olarak Kayıt sayfasının kontrollerini içermektedir.

Classda ilk olarak değişkenler **Encapsulation’a** uygun tanımlanmaktadır. Parametreli ve parametresiz Constracterlar olan KayitController oluşturulmaktadır.

**KayitController() Metodu:** Parametresiz metotta boş bir kayıt oluşturmaya çalışılmıştır hatası dönerken parametreli metotda gelen uygun değerler ad, soyad, yaş, kullanıcı adı ve şifre için kontroledilmektedir. SceneController’da bahsettiğimiz hatalar oluşursa bu hataların uyarısı atılmaktadır.

**KullaniciText() Metodu:** Bütün bilgiler doğru girilmiş ise oluşturulacak kullanıcı için dosya yolu verilerek oluşturulacak kullanıcının Kullanıcı Adı ile bir “.txt” dosyası oluşturularak ve kullanıcının bilgileri oluşturulan dosyaya yazılarak kullanıcı kaydı tamamlanmış olmaktadır.

Class sonunda eğer kayıt yapılmadan anasayfaya dönmek için kullanılan Anasayfa butonu için sayfa geçişi yapılmaktadır.

*SifreUnutController.java*

Bu class genel olarak şifre güncelleme işlemleridir. Kullanıcı adı bilgisinin doğru girilmesi ile şifre güncelleme işlemi yapılmaktadır.

Bu işlemler için kullanılan fxml dosyası SifreUnuttum1.fxml ve SifreUnuttum2.fxml dir. SifreUnuttum1.fxml de kullanıcı kontrolu yapılırken SifreUnuttum2.fxml de şifre güncelleme işlemi yapılmaktadır.

Class başında gerekli Node kodları ve değişkenler verilmektedir.

**DegerAl() Metodu:** Bu metot da eğer kullanıcı adı kontrolü doğru ise şifre güncelleme için hangi kullanıcının dosyasını açacağımızı belirlemek için gönderilen Kullanıcı Adı bilgisini içermektedir.

**KullaniciKontrol() Metodu:** İlk olarak TextField boş mu diye kontrol edilmektedir. Eğer boş ise hata mesajı verilmektedir. Boş değilse girilen kullanıcı adında bir hesap var mı onun kontrolü yapılmaktadır. Eğer girilen kullanıcı adında bir dosya yok ise *KullaniciYok* hatası dönmektedir. Eğer varsa Şifre güncelleme sayfasına gerekli bilgiler (DegerAl()) ile SifreUnuttum2.fxml sayfasına geçiş yapılmaktadır.

**SifreDegistir() Metodu:** ilk olarak şifre TextFieldlar boş mu onun kontrolü yapılmaktadır. Eğer boş ise hata mesajı verilmektedir. Boş değilse önce girilen şifre ve şifre tekrar birbirinin aynısı mı diye kontrol edilmektedir. Kontrol sonrası ilgili kullanıcının dosyası açılıp önceki şifre alınmadan eski bilgiler tutularak girilen yeni şifre dosyaya yazdırılmaktadır. Doğru yapılan işlem sonrası bilgi ekranı ile gösterilmekte ve tekrardan giriş ekranına yönlendirme yapılmaktadır.



Figure 5. Şifre Unuttum Sayfası Kullanıcı Kontrol Görünümü

**

Figure 6. Şifre Unuttum Sayfası Şifre Güncelleme Görünümü

*Mesajlar.java*

Bu class bir Interfacedir. Sadece mesaj() adlı bir gövdesiz soyut metot içermektedir.

*ArkadasEkleController.java*

Bu class genel olarak arkadaş ekleme sayfasının kontrollerini gerçekleştirmektedir.

İlk olarak ArkadasEkle.fxml de kullandığımız Node’lar tanımlanmaktadır. ListView ögesine liste gönderip ListView de yazmasını sağlamak için “list1” adlı özel liste oluşturulmaktadır. Gerekli değişkenler ve ArrayListler class başlangıcında tanımlanmaktadır.

**KayitListesi() Metodu:** Metodumuzda dataBase klasöründe kayıtlarını tuttuğumuz tüm kullanıcıları alma işlemi yapılmaktadır. Dosya yoluna gidip bu klasördeki tüm ögeleri “dizi” adlı List değişkenine alınmaktadır. Daha sonra bu “dizi” adlı listede sadece dosya olan ögeleri almak için tüm listeyi “isFile()” ile kontrolden geçirip “tumKullanicilar” ArrayListine ekleme işlemi yapılmaktadır.

**DegerAl() Metodu:** Bu metot ArkadasEkle sayfasına geçerken girilen kullanıcının hesap bilgilerini Arkadaş eklemek için kullanılmak üzere alma işlemini yapmaktadır.

**metotKullaniciListesi() Metodu:** Metot return olarak bir ArrayList dönmektedir. Burada yapılan işlemler öncelikle listelenecek kullanıcılar içinde kullanıcının kendi hesabının olmaması ve eklediği arkadaşlarının olmamasını sağlama işini yapmaktadır.

Eklediği arkadaşları listelememek için kullanıcının arkadaşlarının bulunduğu dosya açılarak geçici bir ArrayListe tutulur. Daha sonra eklemek istediği kullanıcıyı eklerken önce ilk eklediği kişiler eklenir daha sonra dosya üzerine yeni eklediği kişi eklenerek arkadaş ekleme sağlanmaktadır.

**bulunanListele() Metodu:** Bu metot aranmak üzere girilen metin için benzer kullanıcıları getirmek için kullanılan listelemenin en son kısmı olmaktadır.

İlk olarak aranacak metni ilgili TextFielddan alır daha sonra “metotKullaniciListesi()” metodu çağrılarak listelenebilecek kişiler “liste” adlı bir ArrayListe tutulmaktadır. Bu ArrayListe aranacak metin ile listedeki eklenebilecek kullanıcıların her karakterine bakılarak 2’den büyük benzerlik varsa listelenecek işlemler yapılmaktadır. **‘**Ara’ butonuna her basıldığında aynı kullanıcıyı listeye eklememesi için bulunan kullanıcı sayısını geçmeye çalıştığında bizim listemizdeki eklenmek istenen aynı kişiyi kaldırarak her butona basıldığında doğru listeleme yapılması sağlanmaktadır. Yeni arama yapmak istendiğinde de kullandığımız ArrayListlerin temizlenmesi gerektiği için en sonda bu listeler temizlenmektedir.

Initialize() interface implements edilerek ListView’e istenilen kullanıcılar yazdırılır.

**ArkadasEkle() Metodu:** Doğru verilerin alınmasını sağladıktan sonra bu metotda her “Ekle” butonuna basıldığında tekrar doğru bir listeleme yapmak için “bulunanListele()” metodu çağrılmaktadır.

Daha sonra ListView de seçilen kullanıcının kullanıcı adı “toString()” ile alınarak “bulunan” listemizde bu kullanıcı hangi index de olduğu bulunur ve bulunan kullanıcıyı eklemek için ilgili hesabın dosyası açılmaktadır. Burada eski verileri silmemek için önce geçici bir ArrayListte tutulmaktadır. Daha sonra dosya silinerek tekrardan açılır ve eski veriler yazdırılmaktadır. Daha sonra seçilen kullanıcı en sona eklenme işlemi yapılır ve bilgi penceresi ile mesaj verilir ve dosya kapatılmaktadır. Burada ListView de bir kullanıcı seçilmeden işlem yapılmak istendiğinde hata mesajı verilmektedir.

En sonda güncel listeyi yazdırmak için her ekleme işleminden sonra listeler ve ListView temizlenmektedir.

**KullaniciDetayYazdir() Metodu:** Bir buton ile seçilen kullanıcının yan tarafta ad, soyad ve yaş bilgileri bir kartvizit gibi yazdırılmaktadır.

Bu işlemlerde ilk olarak seçilen kullanıcının adını alarak bu kullanıcın kişisel bilgi dosyası açılır ve ilgili bilgiler “Bilgiler” adlı ArrayListe tutulur. Daha sonra KullaniciBilgiDeger classının **nesnesini tutan kullaniciBilgi arrayListi** oluşturulmuştur. ArratListden seçilen kullanıcının bilgisi kartvizitteki ilgili yerlere yazdırılmaktadır.

Classın en sonunda gerekli sayfa geçişleri ve mesaj işlemi Override edilerek kullanılmaktadır.

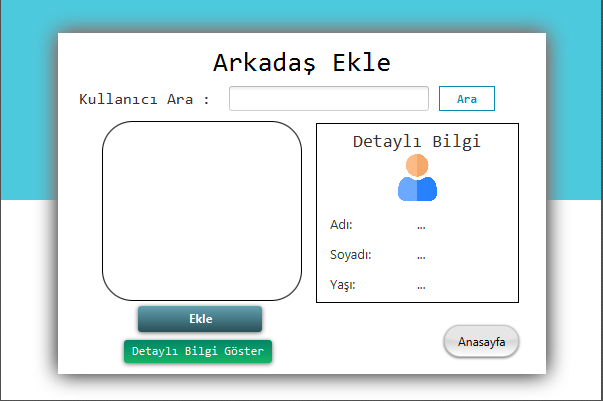


Figure 7. Arkadaş Ekleme Sayfası Görünümü

*ArkadasListeleController.java*

Bu class genel olarak eklediğimiz arkadaşlarımızı görüntülemek ve detaylı bilgilerine bakma işlemlerini içermektedir.

İlk olarak ArkadasListele.fxml de kullandığımız Node’ları ve gerekli değişkenler tanımlanmaktadır.

**DegerAl() Metodu:** ArkadasEkleController.java daki metotla tamamen aynı işlemi yapmaktadır.

**ArkadasListele() Metodu:** Bu metot girilen hesabın arkadaşlarını ListView de yazdırılmasını sağlamaktadır.

İlk olarak ListView de girilen hesabın kullanıcılarını bir resim ile yazdırma işlemi yapılmak istendiği için ilgili resim Image nesnesi oluşturulmaktadır. Daha sonra ilgili dosya açılmış ve dosya içindeki eklenmiş arkadaşlar ListView de tek tek ilgili resim ile yazdırma yapılmaktadır.

“icon” nesnesi resmin yazdırıldığı ImageView nesnesidir. Her arkadaş için oluşturulmaktadır.

updateItem() override metodu ile “icon” nesnesine verilecek resmin boyutları ayarlanmaktadır.

**KullaniciDetayYazdirma() Metodu:** ArkadasEkleController.java daki metotla aynı işi yapmaktadır.

Bu classın en sonunda gerekli sayfa geçişleri ve mesaj işlemi Override edilerek kullanılmaktadır.

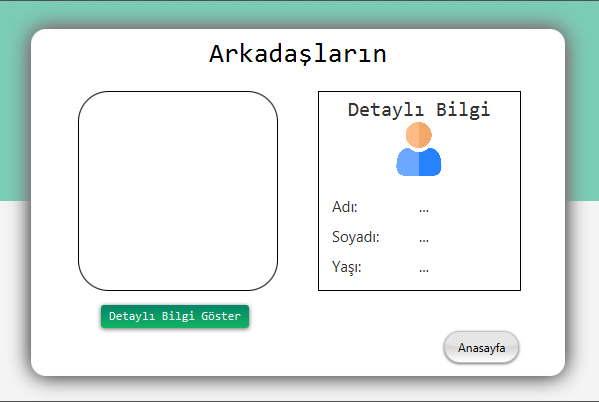


Figure 8. Arkadaş Listele Sayfası Görünümü

*RastgeleArkadasEkleContoller.java*

Bu class genel olarak arkadaşınız olmayan tüm kullanıcıları size öneri olarak sunarak ekleme işlemi yapmayı sağlamaktadır.

İlk olarak RastgeleArkadas.fxml de kullandığımız Node’lar tanımlanmaktadır. ListView ögesine liste gönderip ListView de yazmasını sağlamak için “list1” adlı özel liste oluşturulmaktadır. Gerekli değişkenler ve ArrayListleri de ilk kısımda tanımlanmaktadır.

**tumKullaniciAl() Metodu:** ArkadasEkleController.java daki **KayitListesi()** metodu ile tamamen aynı mantık ile çalışmaktadır.

**DegerAl() Metodu:** ArkadasEkleController.java daki metotla tamamen aynı işlemi yapmaktadır.

**KullaniciListesi() Metodu:** Bu metot kendimizi ve ekli olan arkadaşları listelenecek olan “list1” deki ArrayList den silme işlemini yapmaktadır.

Initialize() interface implements edilerek ListView’e istenilen kullanıcılar yazdırılmaktadır.

**ArkadasEkle() Metodu:** ListView’e listelenmiş veriden seçilen hesabın adı alınarak girilen hesabın arkadaşlarının olduğu dosya açılır, orada olan bilgiler geçici bir ArrayListe atılarak tutulur ve dosya silinerek tekrardan açılır ve ilk olarak eski bilgiler yazdırılır daha sonra seçilen kullanıcının bilgisi yazdırılır ve bilgi penceresi ile bilgi verilerek dosya kapatılmaktadır. Sonra güncel verileri ListView de listelemek için ArrayListlerimiz ve ListView deki verileri temizleyip **KullaniciListesi()** metodu tekrar çalıştırılarak güncel liste ListView’e yazdırılmaktadır.

**KullaniciDetayYazdirma() Metodu:** ArkadasEkleController.java daki metotla aynı işi yapmaktadır.

Classın en sonunda gerekli sayfa geçişleri ve mesaj işlemi Override edilerek kullanılmaktadır.

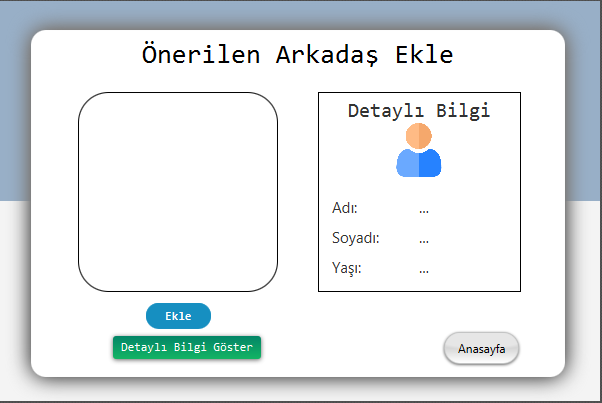


Figure 9. Önerilen Arkadaş Sayfası Görünümü

*ArkadasSilController.java*

Bu class genel olarak eklenen arkadaşları kaldırma işlemi yapılmasını sağlamaktadır.

İlk olarak ArkadasSil.fxml dosyasında oluşturduğumuz Node’ları tanımlanmaktadır. Özel listemiz olan “list1” adlı liste tanımlanmaktadır.

**DegerAl() Metodu:** ArkadasEkleController.java daki metotla tamamen aynı işlemi yapmaktadır.

**ArkadasListele() Metodu:** Girilen hesaptaki tüm arkadaşları “list1” listesine yazdırma işlemi yapılmaktadır.

Initialize() interface implements edilerek ListView’e “list1” listesindeki kullanıcılar yazdırılmaktadır.

**ArkadasSil() Metodu:** ListView’de silinecek olarak seçilen kullanıcının adı alınarak “list1” deki index’i bulunmaktadır. Girilen hesabın arkadaşlarının bulunduğu dosya açılır ve bütün bilgiler bir ArrayListe yazdırılmaktadır. Daha sonra bu dosya silinir ve güncel olarak yazdırılacak kullanıcılar için tekrar ilgili hesabın arkadaşlarının güncel şekilde yazdırılacağı şekilde açılmaktadır. İlk olarak eski verileri yazdırırken silinecek olan index’e gelindiğinde bu değer güncel arkadaşlara yazdırılmaz ve böylece arkadaş silme işlemi tamamlanmış olmaktadır. Daha sonra sayfamızdaki ListView’i güncel tutmak için “list1” temizlenir ve **ArkadasListele()** metodu tekrar çalıştırılmaktadır.

Bu classın en sonunda gerekli sayfa geçişleri ve mesaj işlemi Override edilerek kullanılmaktadır.



Figure 10. Arkadaş Sil Sayfası Görünümü

*SayfaGecis.java*

Bir abstract classdır. Kullanıcı hesabını girdiğinde ekleme, sıralama, silme vb. işlemler yaptıktan sonra hesaptan çıkmadan anasayfaya dönmesini sağlayan metot burada abstract olarak oluşturulmuş ve diğer sayfalardan extends edilmektedir.

*Hatalar Folder*

Bu klasörde Classları açıklarken bahsettiğimiz hatalar bulunmaktadır. Bu hatalar exception class olarak bulunmaktadır.

*dataBase Folder*

Bu klasörde kayıt yapıldıktan sonra oluşturulan ve kişisel bilgileri içeren “.txt” dosyaları bulunmaktadır.

Yine bu klasörün içinde bulunan diğer “Arkadas” klasöründe her kayıt yapan kullanıcının arkadaşlarını tutan “.txt” dosyaları bulunmaktadır.