Java swing kütüphanesi ile geliştirilmiş GUI müzik çalma uygulaması. Temelde 2 kısımdan oluşan uygulama artistler ve dinleyiciler için 2 farklı arayüz sunuyor. Artistler ve dinleyiciler uygulamaya kayıt olurken bilgileri kaydedilir ve uygulama tekrar açıldığında tekrardan kayıt olmanıza gerek kalmaz. Artistler giriş yaptıktan sonra istediği ".wav" uzantılı dosyayı projeye ekleyebilir ve bu dosya kaydedilir. Dinleyici girişi yapıldıktan sonra kaydedilmiş bütün müzik dosyalarını çalabilir durdurabilir ve uygulama hangi müzik türlerini ne kadar dinlediğini ve en çok hangi müzik türünün dinlendiğini raporlar.

Sınıflar, tasarım ve açıklamaları :

Model.User: Abstarct class olrak oluşturulmuştur ve username, password değerlerini özellik olarak tutar. Bu değerler private belirlenmiştir ve getter ve setterleri vardır.

Model.Artist: User sınıfından extend edilir ve ekstra olarak mahlas özelliğini private olarak saklar ve constructor username ve password bekler ve bunları set eder.

Model.Dinleyici: User sınıfından extend edilir ve listenTimes adında HashMap bir özelliği saklar bu HashMap'de dinleyicinin hangi müzik türünü ne kadar süre dinlediği saklanır Key değeri String ve Value değerleri Long tanımlanmıştır.

Helper.Helper: Bu class temelde JOptionPane kullanır ve kullanıcıya yardımcı olması amacıyla tasarlanmıştır. ShowMsg adında static bir metodu bulunur ve String bir değer bekler. Bu metod programcıya kolaylık olması adına bir switch case saklar gelen Stringe göre otomatik mesaj patlatır.
Örneğin "fill" değeri gönderildiğinde otomatik olarak Tüm alanları doldurun mesajı kullanıcıya gösterilir. Ayrıca wrongInput adında farklı bir case'i daha vardır. Switch case'de bulunmayan bir değer gönderilirse gönderilen string değer direkt mesaja yazdırılır. Böylelikle kullanıcı eğer ki bir hata yaparsa kullanıcı bilgilendirilir.

Helper.LogService: Interface olarak tanımlanmıştır printAction adında ve String değer bekleyen metodu bulunur gövdesi yoktur ve void döndürür.

Helper.Logger (implements LogService) : String olan değeri konsol ekranına printAction metodu aracılığılıyla konsola yazdırır uygulama bütününde yapılan olaylar buraya gönderilir ve konsoldan okunması sağlanılır.

Datas.AddArtistPass: Bu sınıfta bir adet "ekleDosyaya" adında static bir metod bulunur. 2 adet String değer bekler bunlar username ve passworddur. Filewriter classı ile bu gelen değerler "artistLoginPass.csv" dosyasına kaydedilir böylelikle datalar kaybolmaz.

Datas.AddDinleyiciPass: Bu sınıfta bir adet "ekleDosyaya" adında static bir metod bulunur. 2 adet String değer bekler bunlar username ve passworddur. Filewriter classı ile bu gelen değerler "DinleyiciLoginPass.csv" dosyasına kaydedilir böylelikle datalar kaybolmaz. Try catch ile hatalar ele alınılır.

Datas.ReadUserPass: Bu sınıfta ReadPass adında HashMap değer geri döndüren static bir method bulunur ve bu method String bir değer bekler ve bu değer artist veya dinleyici olamlıdır

bu metod gelen değere göre gerekli olan kayıt edilmiş csv dosyasını açar sırasıyla okur bir HashMap'e kaydeder ve bu Hashmapi return eder. Exceptionlar ele alınmıştır.

Datas.ReadMusikis : listFilesForFolder adında void döndüren static bir methodu bulunur bu method File nesnesi bekler. Gönderilen pathde bulunan müzikleri listeler ve konsola yazdırır.

Datas.UploadMusic: Müzikleri proje içerisindeki musikiler klasörüne taşır "upload" adında static methodu dosya yolu, sanatçı adı, müziiğin adı ve muzik türünü bekler. Gelen değerleri işler eğer ki gönderilen path doğruysa dosyayı kopyalar ve verilen değerler arasına kısa çizgi koyarak .wav uzantılı bir şekilde musikiler klasörüne kopyalar böylelikle bir kere projeye yüklenen dosya kaybolmaz. Artist nesnesi bu methodu sık kullanmaktadır. (Örnek: Lapsekili Tayfur-Geceler-Sevda)

ClipService.GetMusicList: Dinleyici uygulamayı açtığında Musikiler klasöründeki bütün şarkılar listelenir ve JList kullanılır dolayısıyla bu classdaki getList static methodu DefaultListModel return eder ve Musikiler klasöründeki bütün şarkıları alır dosya uzantılarını kullancıya göstermemek için siler ve listeyi geri gönderir.

ClipService.GetSummaryList: getList adında static bir methodu bulunur DefaultTableModel return eder aynı zamanda Dinleyici sınıfından bir nesne bekler çünkü bu classın amacı spesifik bir dinleyicinin dinleme özetini table haline getirip return etmektir bunu yparken dinleyicinin listenTimes adındaki HashMap özelliğini kullanır. Kullanıcı özet istediğinde bu method çağırlacaktır. Sütunlarında tür ve süre bulunur.

ClipService.LogMusics : kaydet adında static bir methodu , Dinleyici sınıfından bir nesne clip sınıfıdan bir nesne ve müzik türünü bekler. Dinleyicinin listenTimes adındaki HashMapi açılır ve gelen değerler listenTimes mapine eklenir. Böylelikle oluşturulmuş dinleyici nesnensinin özelliğinde her zaman bilgiler tutuluyor olur.

ClipService.MakeClipReady: gonder adında static bir methodu bulunur bu method Clip nesnesi geriye döndürürken String bir müzik ismi bekler. Gelen müzik adıyla müziğin çalınabilmesi için clip nesnesine dönüştürülür ve geri döndürür.

ClipService.MostLikedGenre: Ençok dinlenen türü bulması amacıyla oluşturulmuş bu classda FinlaGenre adında static bir method bulunur. Dileyici nesnesi bekler ve bu nesne üzerinden listenTimes HashMap'ine erişir en çok dinleninlen türü bulur ve bunu String yapıp geri döndürür. Dinleyici Özetinde kullanılan parçalardan biridir.

ClipService.PlaySong: Basit bir class'dır. Gelen Clip nesnesini başlatır return değeri yoktur.

ClipService.StopSong: Basit bir class'dır. Gelen Clip nesnesini durdurur return değeri yoktur.

Main.ArtistKayitGUI : 3 adet Field'ı bulunur ve kayıt ol butonunu saklar. Kullanıcı adı, şifre ve şifreyi tekrar bekler. Alanlar doldurulmadıysa Helper.showMsg çağırılır kullanıcıya alanları doldurması gerektiği söylenir. Her şey doğru girilidiyse ve passwordlar birbirine eşitse kayıt olundu mesajı basılır ve ekran kapanır. Artistimiz giriş yapmaya hazırdır.

Main.DinleyiciKayitGUI : 3 adet Field'ı bulunur ve kayıt ol butonunu saklar. Kullanıcı adı, şifre ve şifreyi tekrar bekler. Alanlar doldurulmadıysa Helper.showMsg çağırılır kullanıcıya alanları doldurması gerektiği söylenir. Her şey doğru girilidiyse ve passwordlar birbirine eşitse kayıt olundu mesajı basılır ve ekran kapanır. Dinleyicimiz giriş yapmaya hazırdır.

Main.Main: Yapımcı imzasını console ekranına yazdırır ve LoginGUI nesnesini çağırır.

LoginGUI : Uygulama çalıştırıldığında bizi ilk karşılayan JFrame'den türemiş ekrandır. Uygulama logosu basılır. 2 adet TabbedPane saklar biri dinleyiciler için diğeri ise artistler içindir.

Her iki TabbedPane de kullanıcı adı ve şifre bekler ayrıca 2 adet butonları bulunur. Giriş butonlarında ise kullancı adı ve şifre csv dosyaları aracılığıyla kontrol edilir doğru olursa giriş yapılır. Doğru değilse Helper.showMsg wrongInput mesajı gönderir. Artist girişi yapıldıysa username password artist nesnesine set edilir ve bu nesneyle ArtistGUI nesnesi çağırılır, dinleyici için de aynı şartlar geçerlidir.

ArtistGUI : Jframeden türer ve constructor'ı çalıştığında artist nesnesiyle çalışır dolayısıyla bu ekrana geçtiğimizde dahi artist nesnesinin özelliklerine erişebiliriz. Örnek olarak ArtistGUI açıldığında karşılama yapılır ve Artist ismi yazdırılır. Artist arayüzünün genel amacı yalnızca müzikleri musikiler klasörüne çekmektir bunun için birkaç bilgiye ihtiyacımız var . Bunlardan ilki sanatçımızın adı yani artistimiz bu anlamda artist nesneiyle çağırdığımız için default olarak username sanatçı ismi olarak geliyor. Şarkının adını girebileceğimiz bir field ve müziğimizin türünü girebileceğimiz bir field daha bizi karşılıyor. Daha sonra 2 adet buton görüyoruz bunlar dosya seç ve şarkıyı pakete olacaktır. Buradaki dosya seçme butonumuz FileChooser nesnesidir ve basıldığında kullanıcı ekranın istediği dosyayı seçebileceği bir ekran karşılar .wav uzantılı dosyasını seçtikten sonra open a basılır ve seçilen dosya yolu ekranın en altında yazdırılır. Dosya yolunu yani pathi String olarak alıyoruz Bunun sebebi ise File.move methodunun String değer beklemesinden dolayıdır kullanıcı bütün bunları yaptıktan sonra şarkıyı pakete ekle dediğinde dosyanın bir kopyası oluşturulup ismi formatlanıp src/Musikiler klasörüne eklenir. Artistlerin müzik ekleme işlemi bu şekilde gerçeklenir. Oldukça basit bir arayüz tasarımna sahip olan ArtistGUI bütün ihtiyaçlarınızı karşılayacaktır.

DinleyiciGUI: LoginGUI de giriş yapıldıktan sonra oluşan dinleyici nesnesi bu JFrame'e parametre olarak gönderilir ve ekranımız açılır. Burada bizi bir Liste karşılar bu listede src/Musikilerde bulunan bütün müzikleri görebiliyor olacağız program açılsa kapansa bile bu dosyalar projemizde kayıtlı olucağından dolayı. Buradaki listeyi getirirken Jlistten faydalanıyoruz ve bu Jlist model olarak ReadMusikis sınıfını kullanmaktadır. Arayüzün altında Oynatma Durdurma ve Özet Butonlarını görüyor olacağız. Jlistten eleman seçtikten sonra oynat tuşuna bastığımızda program seçilen şarkıyı oynatmaya başlayacktır. Eğer aynı şarkı sçeiliyken oynat tuşuna basılırsa şarkı başa saracaktır. Farklı bir şarkı seçildiğinde eskisi durdurulup yenisi başlakta Durdurma butonu yalnızca şarkıyı durdurur. Özet butonu ise dinleyicinin hangi müzik türlerini ne kadar süre dinlediğinin raporlarını verir detaylar için SummaryGUI. DinleyiciGUI ile müzikler dinlenebilir ve raporları alınabilir.

SummaryGUI: DinleyiciGUI üzerinden özet tuşuna basıldığında dinleyici nesnesini parametre alıp açılacaktır. Burada dinleyicinin listenTimes özelliğine erişilip hangi müzik türünün ne kadar dinlenildiği HashMap'e çekilir bu yapılrıken GetSummaryList classı kullanılır tabiki bunu görüntülemek adına Jtable kullandık dolayısıyla GetSummaryList classının getList methodu bize DefaultTableModel geri döndürüyor olacak. Bu DefaultTableModeli SummaryGUI ekranında kullandığımız Jtablea model olarak gönderiyoruz ve Hangi müizğin ne kadar süre çalındığı ekranımıza bastırılıyor . SummaryGUI nin en alt kısmında ise en çok dinlenen müzik türü yazdırılmakta burda ise MostLikedGenre class'ını kullanıyoruz ve sonuç olarak bu ekranımızda hangi müzik türlerinin en kadar süre dinlendiği ve en çok dinlenen müizk türü yazdırılmakta . Dinlenen süre hesaplanırken basit bir hesap yerine program gerçekten gerçek hayatta ne kadar dinlenildiyse o kadarını göstermektedir. Bunun sonucunda isabetli sonuçlar alınılır.