DigiLibrary Proje Raporu

İçindekiler

1. Özet

DigiLibrary, kullanıcıların kitaplarını dijital ortamda organize etmelerini sağlayan bir masaüstü uygulamasıdır. Java programlama dili ve Swing kütüphanesi ile geliştirilen bu uygulama, kullanıcı kaydı ve girişi, kitap ekleme-düzenleme, alıntılar kaydetme, okuma hedefleri belirleme ve ilerlemeleri izleme gibi işlevler sunar. Veriler JSON dosyalarında tutulmakta ve uygulama, modüler yapıdaki sınıflar üzerinden yönetilmektedir. Kullanıcı dostu arayüzü sayesinde, kullanıcılar dijital okuma alışkanlıklarını yönetebilmektedir.

2. Giriş

Günümüzde dijital okuma alışkanlıkları artarken, kişisel kütüphanelerin yönetimi için kullanıcı dostu uygulamalara olan ihtiyaç da artmıştır. Bu projede, kitap koleksiyonlarının, alıntıların ve okuma hedeflerinin dijital ortamda takip edilmesini sağlayan bir sistem geliştirilmiştir. DigiLibrary uygulaması, geleneksel defter veya mobil uygulamalara alternatif olarak masaüstü ortamında çalışacak şekilde tasarlanmıştır. Bu sayede hem kolay erişim sağlanmakta hem de verilerin kontrolü tamamen kullanıcıya bırakılmaktadır.

3. Yöntem ve Tasarım

Bu proje Java ile geliştirilmiş olup Swing kütüphanesi ile grafik kullanıcı arayüzü (GUI) tasarlanmıştır. Veriler, kalıcı olarak .json dosyalarında saklanmakta ve bu dosyalar repository katmanındaki sınıflar tarafından okunup yazılmaktadır. Projede MVC (Model-View-Controller) tasarım deseni tercih edilmiştir.

- Model Katmanı: User, Book, ReadingGoal, Quote, ReadingProgress gibi veri sınıflarını içerir.
- Controller Katmanı: LibraryController, UI ve servis katmanı arasındaki etkileşimi yönetir.

- Service Katmanı: İş mantığını kapsayan BookService, UserService gibi sınıflar içerir.
- Repository Katmanı: JSON dosyaları ile veri alışverişi yapan JsonUserRepository vb. sınıflar bulunur.
- UI Katmanı: Kullanıcı arayüzünü yöneten LoginPanel, MainPanel, QuotesPanel gibi paneller yer alır.

Bu yapı sayesinde her katman görevine odaklanır ve sistem sürdürülebilir hale gelir.

4. Kullanılan Teknolojiler ve Yapılar

Yazılım Dili ve Ortam:

- Java (JDK 17)
- Java Swing (Grafik Arayüz)
- Maven (Bağımlılık yönetimi)
- JSON (Veri saklama)

Mimari Yapı:

- MVC (Model-View-Controller) mimarisi
- OOP (Nesne Tabanlı Programlama)
- Repository ve Service katmanları

Veri Yapıları ve Dosyalar:

- Kitaplar `books.json` dosyasında
- Alıntılar `quotes.json` dosyasında
- Okuma hedefleri `goals.json` içinde
- Kullanıcılar `users.json` içinde saklanır

5. Uygulama Özellikleri

5.1. Kullanıcı Yönetimi

- Kullanıcı kaydı ve giriş işlemleri

- Kullanıcı rolleri (`User`, `Admin`) desteği

5.2. Kitap Yönetimi

- Kitap ekleme, silme, düzenleme
- Kitap listesi görüntüleme

5.3. Alıntılar Paneli

- Kitaplardan alınan alıntılar oluşturulabilir
- Alıntı listesi kullanıcı bazlı filtrelenebilir

5.4. Okuma Hedefleri ve İlerlemesi

- Günlük, haftalık, aylık hedefler belirlenebilir
- Kitap bazlı okuma ilerlemesi kaydedilebilir
- Grafiksel veya sayısal istatistikler görüntülenebilir

5.5. Grafik Arayüz (GUI)

- Java Swing ile sekmeli yapı
- Login, Register, Main Panel gibi bölümler
- Panel sınıfları: `LoginPanel`, `MainPanel`, `QuotesPanel`, `ReadingGoalsPanel`, `ReadingProgressPanel`, `StatsPanel`

6. Yazılım Mimarisi

6.1. Model Katmanı

- `Book`, `User`, `Quote`, `ReadingGoal`, `ReadingProgress` gibi sınıflardan oluşur.

6.2. Repository Katmanı

- Verilerin JSON dosyalarından okunmasını ve kaydedilmesini sağlar.
- Örnek: `JsonBookRepository`, `JsonUserRepository`

6.3. Service Katmanı

- İş mantığı içerir. Controller'dan gelen istekleri işler.
- Örnek: `BookService`, `UserService`

6.4. Controller Katmanı

- GUI ile veri katmanı arasındaki iletişimi sağlar.
- Örnek: `LibraryController`

6.5. Arayüz (UI) Katmanı

- Kullanıcının etkileşimde bulunduğu Swing tabanlı panel yapıları

7. Uygulama Aşamaları

Proje aşağıdaki adımlarla geliştirilmiştir:

1. Temel Yapının Kurulması: pom.xml ile Maven yapılandırması yapıldı, proje klasörleri oluşturuldu.
2. Model Sınıflarının Geliştirilmesi: Kullanıcı, kitap, alıntı, hedef gibi sınıflar tanımlandı.
3. Veri Yönetimi: JSON üzerinden veri okuma/yazma işlemleri JsonDataManager aracılığıyla yönetildi.
4. Kullanıcı Arayüzü: Swing ile login ekranı, kayıt ekranı, kitap listesi ve hedef ekranları geliştirildi.
5. İş Mantığı: Kullanıcı işlemleri, kitap ekleme ve okuma ilerleme hesaplamaları servis katmanında kodlandı.
6. GUI Entegrasyonu: Arayüzle servisler entegre edilerek etkileşimli hale getirildi.

8. Test ve Sonuçlar

Uygulama manuel olarak test edilmiş, aşağıdaki senaryolar çalıştırılmıştır:

• Kullanıcı Kaydı ve Girişi → Başarılı
• Kitap Ekleme ve Listeleme → Başarılı

• Alıntı Oluşturma → Başarılı
• Okuma Hedefi Belirleme ve Takibi → Başarılı
• JSON dosyalarında veri kaydı → Başarılı

Kullanıcı arayüzü ile yapılan tüm işlemler doğru şekilde veritabanına (JSON dosyalarına) kaydedilmiş ve yeniden yüklenmiştir.

9. Sonuç

DigiLibrary, bireylerin okuma alışkanlıklarını dijital ortamda izlemelerine olanak tanıyan, kullanıcı dostu ve işlevsel bir masaüstü uygulama olarak başarıyla geliştirilmiştir. Uygulama, kullanıcıların kitaplarını kolaylıkla yönetebilmesine, anlamlı alıntılar biriktirmesine, kişisel okuma hedefleri belirlemesine ve bu hedefler doğrultusunda ilerlemelerini düzenli olarak takip edebilmesine imkân tanımaktadır.

Proje, yazılım geliştirme sürecinde modern yazılım mimarileri ve iyi tasarım ilkeleri dikkate alınarak planlanmış ve gerçekleştirilmiştir. MVC yapısı ve katmanlı mimari sayesinde sistemin sürdürülebilirliği, esnekliği ve genişletilebilirliği ön planda tutulmuştur. JSON tabanlı veri saklama yaklaşımı, basit ama etkili bir yöntem olarak tercih edilmiş; veri bütünlüğü ve uygulama performansı test senaryolarında başarıyla gözlemlenmiştir.

Kullanıcı arayüzü, Java Swing bileşenleri kullanılarak sezgisel ve erişilebilir bir şekilde tasarlanmıştır. Login ekranından kitap yönetim paneline, alıntı ve hedef izleme ekranlarına kadar her bir bileşen, kullanıcı deneyimini artırmaya yönelik olarak yapılandırılmıştır. Farklı kullanıcı türleri (örneğin yeni başlayanlar ve kitap koleksiyoncuları) için özelleştirilebilir bir yapı oluşturulması da göz önünde bulundurulmuştur.

Ayrıca proje, açık kaynaklı bir yapıya sahip olduğundan dolayı akademik, bireysel veya kurumsal düzeyde özelleştirme ve geliştirme için oldukça elverişlidir. Bu sayede üniversitelerde öğrenci projeleri, bireysel dijital kütüphane ihtiyaçları veya küçük yayınevlerinin kitap takibi gibi çeşitli alanlarda uygulanabilir hale gelmiştir.

Sonuç olarak DigiLibrary, sadece bir yazılım uygulaması değil; aynı zamanda bireylerin okuma alışkanlıklarını bilinçli bir şekilde yönetmelerine katkı sağlayan dijital bir asistan niteliği taşımaktadır. Geliştirilmeye açık yapısı sayesinde, gelecekte çok daha kapsamlı bir dijital okuma platformuna dönüştürülme potansiyeli taşımaktadır.