YAZILIM LABORATUVARI 1

1. PROJE

Muhammet Ali BAKINÇ
Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi

Özet

Bu doküman Yazılım Laboratuvarı 1 dersi 1. 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 akıllı kargo dağıtım sistemi yapan bir masaüstü uygulaması geliştirmemiz beklenmektedir. Uygulamada iki farklı GUI olması, birinci ekranda kargo firmasına ait işlemler, ikinci ekranda ise bir harita olması istenmektedir.

Tasarlayacağımız programda kullanıcıların kayıt olabilmesi ve giriş yapabilmesi gerekmektedir.

Program işleyişi aşağıdaki gibi olmalıdır:

- Program giriş ekranıyla başlar. Kullanıcı giriş yap veya kayıt ol alanlarını kullanarak bilgilerini girmesi beklenmektedir. Veriler bulut veri tabanından kontrol edilerek ana sayfaya geçilir.
- Ana sayfada öncelikle ilk dağıtım bilgileri buton ile veri tabanından alınır.
- En kısa yol algoritması ile öncelik sırasına göre ekrana listelenir.
- Harita açma butonu ile ikinci ekran başlatılır ve konumlar ve gidilecek en kısa yol ekranda gösterilir.
- İlk ekrandaki açılır menüden yeni konum eklenebilir. Konum eklendiğinde liste ve harita yeniden güncellenir.
- Listedeki herhangi bir konum sil butonuyla kaldırılabilir. Ekranlar aynı şekilde güncellenir.

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. Tasarım esnasında zaman zaman kullanımında zorlandığım bazı araçlar için yine internette araştırmalar yaptım. Diğer bir kısım da bulut veri tabanı ve harita API'sini kullanmaktı. Bulut veri tabanında *Firebase* kullandım. Harita için de *Google Maps API*'yi kullandım.

3. Geliştirme Ortamı

Projemi Windows sistemde, *Intellij Idea* üzerinde geliştirip derledim. *Java* dilini, *Firebase* bulut veri tabanını, *Spring Boot* ve *JavaFX* kütüphanelerini kullandım.

4. Kod Bilgisi

4.1. 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.

Home Kontrol sınıfı

Ekranın ana sayfasının iskeletinin kontrolünü sağlar. En üst sütunda bulunan Harita Aç butonu ile *haritaAc* metodu çağrılır ve ikinci GUI yüklenir. Bir alt sütunda açılır pencereden seçilen konum hemen yanında bulunan Konum Ekle butonu ile *konumEkle* metoduna gönderilir. Sınıf *initialize* ile çalışmaya başlar. Burada veri tabanından tüm kayıtlı konumların listesi alınır ve açılır pencereye yüklenir. İlk açılışta aktif olan tek buton Liste Getir *listeGuncelleEkle* metodunu çağırır. Burada veri tabanında bulunan dağıtıma hazır kargoların konum bilgileri alınır. En kısa yol algoritması ile gerekli sıralama yapılır. Ekrana *listeEkle* metodu ile listelenir. İkinci ekranda çizdirilmesi için hesaplanan yol MapController sınıfına gönderilir.

List Item Kontrol sınıfı

Ekrana listelenen konum bilgilerini içeren liste elemanlarını oluşturan sınıftır. Gelen bilgileri *setData* metodu ile tasarıma ekler. Sil butonu için oluşturulan *adresSil* metodu ile silinecek konum önce veri tabanından silinir. Ardından HomeController sınıfındaki *listeGuncelleSil* metodu çağrılarak liste ve harita güncellenir.

Login Kontrol sınıfı

Programın başlangıç ekranı olan giriş yapma ekranını kontrol eder. İlk olarak *initialize* ile başlar. Burada veri tabanından kullanıcı bilgileri alınır. 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 tabanından gelen listeden 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. Aynı şekilde ana sayfaya yönlendirilir.

Map Kontrol sınıfı

İlk ekrandan gelen yol bilgisini setData metodu ile alır. Oluşturduğu Browser nesnesine parametre olarak gönderir.

FirebaseCRUD sınıfı

Tüm bulut veri tabanı işlemlerinin yönetildiği sınıftır. Kullanıcı oluşturma, alma, konumları alma, yeni kargo ekleme, silme gibi tüm işlemlerin *static* metotlar ile kullanımını sağlar

Map Service sınıfı

Google Maps API'si ile iletişim sağlayan metotlar bulunduğu sınıftır. Gelen konum bilgileri *createMap* ile oluşturulan özel link ekranda gösterilmek üzere döndürülür. Aynı zamanda iki konum arasındaki mesafeyi hesaplayan *getDistance* metodu da API aracılığıyla işlemleri tamamlar.

ShortestPath sınıfı

En kısa yolu hesaplayan algoritmaları kullanan metotları içerir. İlk olarak çağrılan *tabloOlustur* parametre olarak gelen konum listesini her ikili arasındaki mesafeyi hesaplamak için Map Service sınıfındaki metoda gönderir. Aldığı değerleri iki boyutlu bir matriste saklar. Sonrasında çağırdığı *permute* metodu ile konum sayısı uzunluğundaki aritmetik sayı dizisinin tüm sıralanmış olasılıklarını oluşturur. İlk elemanı başlangıç noktası olan listeler filtrelenir ve her biri için matristeki değerler kullanılarak yol uzunlukları hesaplanır. En

kısa mesafe ile gezilebilecek yol sırası geri döndürülür.

Browser sınıfı

İkinci ekranda gösterilen harita ile ilgili güncelleme işlerinin yapıldığı sınıftır. Tasarım değişkenlerinin yanı sıra *createMap* metodu ile Map Service'ten gelen özel linki tarayıcı ile ekranda oluşturur. Aldığı konum listesinin *yolCiz* metoduna da göndererek en kısa yolun ekrana çizdirilmesini sağlar.

Location sınıfı

Veri tabanından alınan konum bilgileri için türetilen model sınıftır. *ID, konumAdi, koordinat, lineX ve lineY* değişkenleri bulunur. Buluttan gelen veriyi nesneye dönüştüren *toLocation* metodu ve nesneyi veri tabanına yazmaya uygun hale getiren *toMap* metotları bulunur.

User sınıfı

Veri tabanından alınan kullanıcı bilgileri için türetilen model sınıftır. *Email* ve *password* değişkenleri bulunur. Buluttan gelen veriyi nesneye dönüştüren *toUser* metodu ve nesneyi veri tabanına yazmaya uygun hale getiren *toMap* metotları bulunur.

4.3. İstatistik

Program kodu boşluklar ve tasarım kodları dahil yaklaşık 1175 satırdan oluşmaktadır. Kullandığım kütüphaneler ve ne için kullandığımız kabaca aşağıdaki gibidir:
javafx
Tasarım elemanlarını kontrol etmek için kullanıldı.
com.google.firebase
Veri tabanı işlemleri için kullanıldı.
org.springframework.boot
Uygulamanın internet ile iletişim kurabilmesini ve gerekli paketlerin eklenebilmesine olanak sağlamak için kullanıldı.
java.io
Hata yönetimi için kullanıldı.
java.util
Listeler gibi temel Java değişkenleri için kullanıldı.
com.google.maps
Harita işlemleri için kullanıldı.

4.4. Programın Derlenmesi

Programın kaynak kodu 4 kontrol sınıfı, 2 model sınıfı, 4 veri tabanı ve servis sınıfı, 4 *FXML* kodundan oluşmaktadır. Bu projeyi *Intellij Idea* veya farklı bir Java IDE'si kullanarak derleyebilirsiniz. Derlerken dikkat edilmesi gereken mesele ise programa JavaFX, Spring Boot ve Firebase kütüphanelerinin eklenmiş olması gereklidir.

Kaynakça

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

2. Firebase kullanım rehberi, https://firebase.google.com/docs/firestore

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

Akış Diyagramı