Kargo Dağıtım Sistemi

Kaan Kalaycı, Alperen Ünlü

Bilgisayar Mühendisliği Bölümü

Kocaeli Üniversitesi

ahmet.alper96@gmail.com.tr, kaan.klyc1@hotmail.com

Özet:

Yazılım Laboratuvarı I projesi olarak bizden bir akıllı kargo dağıtım sistemi istenmiştir. Bu sistem 2 GUI ekranından oluşmaktadır. 1.GUI login ekranı, kargoların sisteme eklendiği teslimat adres ekranı ve teslimatları listeleyen teslimat durum ekranından oluşmaktadır. Login ekranında kullanıcı, kullanıcı adı ve şifresi ile giriş yapabilmeli, şifre değiştirilebilmeli ve kayıt olabilmeli. Teslimat adres ekranında teslim edilecek kargoların konumları harita üzerine tıklanarak ve ayrıca manuel olarak da girilebilecek şekilde olmalıdır. Ayrıca bu bilgiler masaüstü uygulaması kullanılarak girilecek olup bulut veri tabanında tutulması gerekmektedir. Teslimat durum ekranında kargoların teslim durumları liste olarak görüntülenecek. Kargo ekleme silme ve teslimat bilgisi girme işlemleri yapılacak. Kullanıcı, lokasyon bilgileri alınarak kargoları en kısa mesafeleri kullanarak teslim etmeye çalışacaktır. Geliştirilen uygulamada teslimat adresleri harita üzerinde gösterilecektir. 2.GUI teslimatları ve yollarını gösteren harita ekranıdır. Teslimat sırasını bulabilmek için en kısa yol algoritmalarından birinin (Dijkstra, A* Prim, Kruskal vb.) kullanılması zorunludur. 1. Ekrandaki teslimat durum ekranından kargo teslim edildiği zaman harita ekranından işaretli bölge kaldırılacak ve kargo teslim edildi bilgisi teslimat durum ekranında teslimat durumu güncellenecektir. Kargo firması teslimat adres ekranına yeni bir kargo girişi yaptığı anda, haritanın bulunduğu 2. ekranda yer işareti belirlenecek ve en kısa yol güncellenerek tekrardan çizdirilecektir. Bu GUIler threadler vb. araçlarla iletişim içinde olmalıdır.

Giriş:

Bu projede kaynak fazlalığından dolayı isterleri daha rahat gerçekleştirebileceğimiz C# dilini kullandık. IDE olarak Visual Studio kullandık. Bulut veritabanı için ücretsiz kullanım süresi daha uzun olan Google Firebase Realtime veri tabanını kullandık. Veri tabanına erişim için Visual Studio'muza FireSharp paketini ekledik. Harita işlemlerimizi Google Maps üzerinden gerçekleştireceğimiz için GMap.Net.Windows paketini ekledik. Ara yüzümüzün tasarımı için Visual Studio'nun sağladığı Windows Forms'u kullandık.

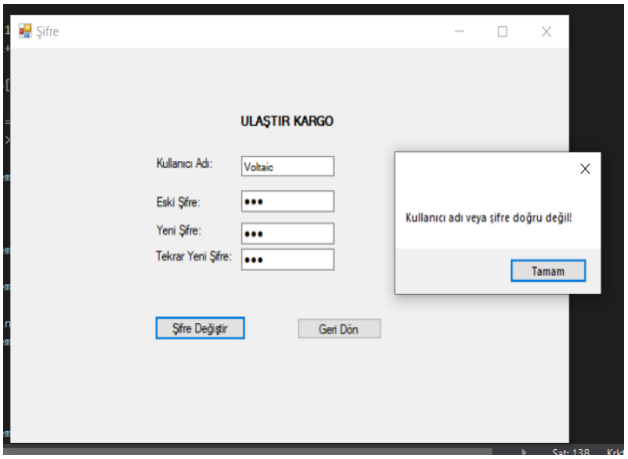
Yöntem:

İlk olarak veri tabanıyla ilgili işlemlerimizi kolaylaştırması için Bağlantı adında bir sınıf oluşturduk. Burada veri tabanımıza bağlandık ve kullanmaya hazır komut nesneleri oluşturduk. Ayrıca veri tabanı nesnelerimizi oluşturmak için Kullanıcı ve Sipariş isminde 2 sınıf oluşturduk. Kullanıcı sınıfının kullanıcıAdı ve şifre özellikleri var. Sipariş sınıfının telNo, müşteriAd, müşteriEnlem, müşteriBoylam ve teslimDurum özellikleri var. telNo özelliği Firebase veri tabanında otomatik artan bir primary key özelliği bulunmadığı için projenin isterlerinde geçen kargoID özelliği yerine geçiyor. müşteriEnlem ve müşteriBoylam haritada teslimat adreslerini oluşturmak için var. teslimDurum özelliği ilk defa sipariş oluşturulurken mutlaka "Teslim Edilmedi" durumunda bulunuyor. Bunu teslimat ekranı

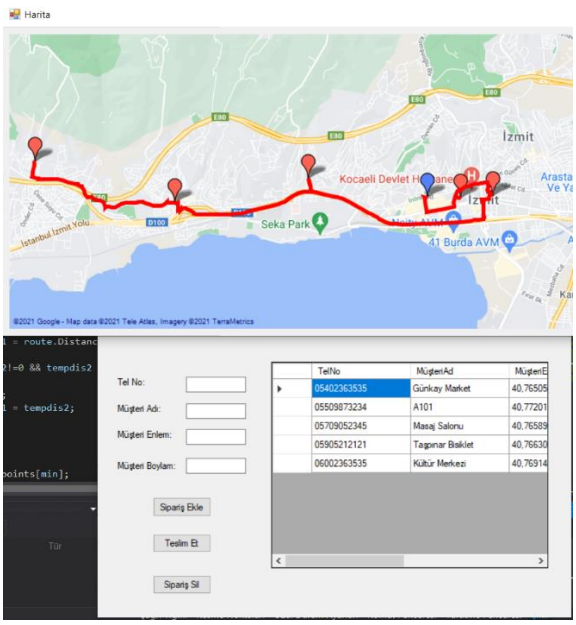
ara yüzümüzde Teslim Et tuşumuzu kullanarak değiştirdiğimizde sipariş haritadan siliniyor. Proje ilk açıldığında login ekranı geliyor. Burada giriş yapmanın yanı sıra kayıt olma ve şifre değiştirme özellikleri bulunuyor. Bu özellikler kullanıcının girdiği kullanıcı adına bakarak veri tabanında güncelleme yapıyor. Ayrıca şifre değiştirme ekranında güvenlik amacıyla kullanıcı adı ve eski şifrenin eşleşmesi kontrol ediliyor, böylece isteyen biri sadece kullanıcı adını bilerek değişiklik yapamıyor. Aynı yöntemi kullanarak giriş yapmayı sağlıyoruz. Giriş yaptıktan sonra Arayüz sınıfı ve ona bağlı olan Form1(harita) sınıfı açılıyor. Arayüzde bulunan "güncelle" adlı bir metot sayesinde veri tabanındaki siparişler listesini bir datagridView tablosuna aktarıyoruz. Aynı metot içinde yapılan diğer işlemler için haritanın güncellenmesi gerekiyor, bu yüzden haritayı kapatıp tekrardan açıyoruz. CellClick metotu sayesinde bir datagridView tablosundaki siparişlere tıkladığımızda onların özelliklerini textboxlara çekiyoruz bu da daha kolay işlem yapmamızı sağlıyor. Sipariş Ekle butonuyla textboxlara girdiğimiz özelliklerde bir sipariş oluşturuyoruz ve veri tabanına yolluyoruz. Ayrıca güncelle metodunu çağırarak tablomuzu ve haritamızı yeniliyoruz. Sil butonuna tıkladığımızda siparişin telNosuna bakarak o siparişi siliyor ve aynı şekilde tablo ile haritayı güncelliyor. Teslim et butonu ise siparişin telNosuna bakarak o siparişin teslimDurum özelliğini "Teslim Edildi" olarak değiştiriyor ve aynı şekilde tablo ile haritayı güncelliyor. Son olarak Form1 yani harita sınıfımızda Google haritasını çağırmamızı sağlayan API anahtarı bulunuyor. Haritamız açıldığında başlangıç noktasını İzmit Cumhuriyet Parkı olarak ayarladık. Haritada sağ tıkladığımızda yanda bulunan enlem ve boylam kutucuklarına tıkladığımız noktanın enlem ve boylamını veriyor böylelikle rahatça yeni bir sipariş oluşturabiliyoruz. Burada veri tabanıyla bağlantı kuran "veriAl" metodu bulunuyor. Bu metot ilk olarak veri tabanındaki siparişlere bakarak henüz teslim edilmemiş siparişlerin adreslerini bir listeye atıyor. Sonra başlangıç noktası mavi, teslim noktaları kırmızı olacak şekilde haritaya markerlar ekleniyor. Sonra Selection Sort kullanılarak listeye attığımız adreslerin birbiri arasındaki mesafelere bakarak birbirine en yakın adresler listede tekrardan düzenleniyor. Sonra olarak bu düzenlenmiş noktalardan yararlanılarak harita üstünde yollar çiziliyor. Bu "veriAl" metodu harita açılırken çalıştığı için teslimat ekranında haritayı kapatıp açmamız haritanın güncellenmesini sağlıyor.

Deneysel Sonuçlar:

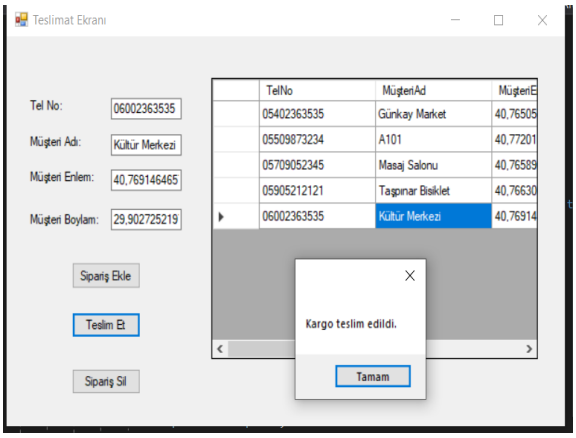
Şifre değiştir ekranındaki kontrol:



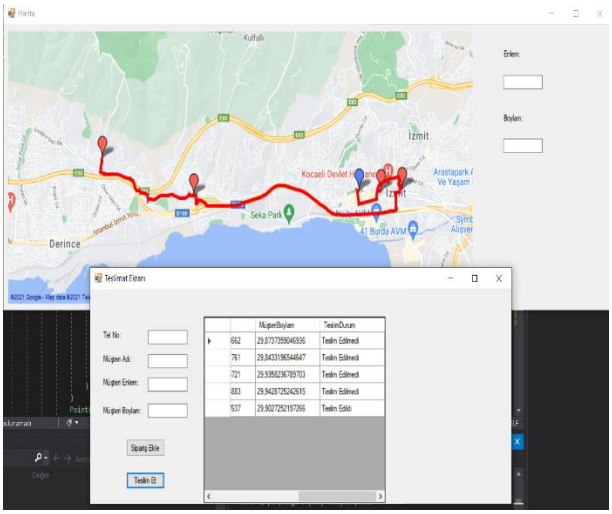
Haritanın veri tabanından bilgi alarak yol çizmesi:



Sipariş teslim ederek haritanın güncellenmesi:



Ortadaki nokta kayboldu ve liste güncellendi.



Sonuç:

Bu projede isterlerden biri 2 GUI arasında iletişim kurulmasıydı. Biz bu iletişimi sağlayamadık. Bu yüzden harita üstünde kuryenin hareketi veya haritaya tıklayarak konum bilgisini teslimat ekranına göndermek gibi işlemleri gerçekleştiremedik. Threadler gibi bu özelliklere imkan sağlayan araçlara daha çok çalışmamız gerektiğini öğrendik. Başka bir ister teslimat noktalarının uzaklıklarının en kısa yol algoritmalarıyla hesaplanarak haritanın çizilmesiydi. Biz bunun yerine Selection Sort kullanarak noktalar arası yol uzaklığına bakarak sıralama yaptık. Ancak bu sorting sonucu bazen hatalı çizimlere yol açabiliyor. Bunun sebebini çözemedik. Tahminimiz algoritmamızın yanlış kurulmasıdır. Bu proje C# diliyle yazdığımız ilk proje oldu. Projenin hazırlanma sürecinde dilin syntaxına alıştık.

Yalancı Kod:

1-Başla.

2-Veri tabanı kur.

3-Login ekranı aç.

4-Kayıt ol ekranı aç.

5-Kullanıcı adı, şifre ve tekrar şifre gir.

6-Login ekranı aç.

7-Şifre değiştirme ekranı aç.

8-Kullanıcı adı, şimdiki şifre, yeni şifre ve tekrar yeni şifre gir.

9-Login ekranı aç.

10-Yeni bilgilerle giriş yap.

11-Kullanıcı adı ve şifreyi veri tabanında kontrol et.

12-Teslimat ekranı ve haritayı aç.

13a-Haritada veri tabanında teslim edilmemiş kargoların adreslerini bir listeye at.

13b-Teslimat ekranında teslimatları listele.

14-Adres listesini selection sort yaparak adreslerin aralarındaki mesafelerine göre sırala.

15-Adres listesine göre haritada yolları çiz.

16a-Teslimat ekranına müşteri bilgilerine gir, ekle tuşuna bas, listeyi ve haritayı yenile.

16b-Teslimat ekranındaki listeden bir siparişe tıkla, teslim et tuşuna bas, listeyi ve haritayı yenile.

16c-Teslimat ekranındaki listeden bir siparişe tıkla, sil tuşuna bas, listeyi ve haritayı yenile.

Kaynakça:

https://www.geeksforgeeks.org/selection-sort/

https://www.youtube.com/playlist?list=PLlD7n_T-mUjVuqIhWVfaNhnpqCZmNcA9e

https://github.com/ziyasal/FireSharp

https://www.youtube.com/watch?v=QE5UV8NyYqg

https://www.youtube.com/watch?v=2GSmMOeP4_g

https://www.youtube.com/watch?v=oeblTSuBjSo