# KOCAELİ ÜNİVERSİTESİ
# BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

## YAZILIM LAB. I 2021-2022 GÜZ DÖNEMİ
## PROJE 1


180201018-Ayşenur Akpınar
Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi
aysenurakpinar98@gmail.com

190201100-Rukiye Canlı
Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi
rukiyecanli435@gmail.com


## 1.ÖZET


Projede programlama dili olarak Java ve Javascript kullanılmıştır. Bu projede akıllı kargo dağıtım sistemi yapan bir masaüstü uygulaması geliştirmemiz beklenmektedir. Masaüstü uygulama olarak iki farklı GUİ ekranı yapmamız istenmektedir.1.ekranda kargo firmasına ait isterler,2. Ekranda ise harita ekranımızın olması istenmektedir.


## 2.GİRİŞ

Bu kısım projenin açıklamasını okuyup edindiğimiz ön bilgiye göre yazılmıştır. Projenin bizden isterleri aşağıdaki gibidir ;

1. GUI:

● Login ekranı: Kullanıcı, kullanıcı adı ve şifresi ile giriş yapabilmeli, şifre değiştirilebilmeli ve kayıt olabilmeli.

● Teslimat adres ekranı: 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.

● Kullanıcının konum bilgileri harita üzerinden alınacaktır ve elle de giriş yapılabilmesi sağlanmalıdır.

● Teslimat durum ekranı: 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. Örneğin İzmit Yürüyüş Yolunda bir kargo dağıtım merkezi bulunmaktadır. Bu dağıtım şirketinin elindeki toplam 5 kargoyu dağıtması gerekmektedir. Teslim edilecek kargolardan 1.Kargo paketi İzmit Cumhuriyet Cad, 2. Kargo paketi Kartepe 3. Kargo paketi, Umuttepe, 4. Kargo paketi Bağçeşme, 5. Kargo paketi ise Yahya Kaptan'dır. Kuryenin en az mesafe giderek tüm kargoları teslim

etmesi beklenmektedir. Geliştirilen masaüstü uygulamasında teslimat adresleri harita üzerinde gösterilecektir. Daha sonrasında ise teslimat sırası en kısa yol algoritmasına göre çizdirilecektir. (Teslimat sırasını bulabilmek için en kısa yol algoritmaları II. GUI üzerinde yapılacaktır)

● Teslimat durum ekranı ile 2. GUI'de bulunan harita senkronizasyonu thread mantığı ile çalışacaktır. Bu ekranda yapılan güncellemeler thread ile en kısa yol algoritmasını çalıştırıp en kısa yol güncellenecektir. Bu işlem sırasında uygulama yeniden başlatılmayacaktır. Güncellemeler gerçek zamanlı olarak o anda gösterilmelidir. Güncelleme sonucu 2. Ekranda harita üzerinde güncel yol çizdirilecek.

2. GUI:

● 2. ekranda sadece harita arayüzünün olması gerekmektedir.

● Haritada teslimat adresleri işaretlenecek ve kargocunun (kuryenin) konumu belirtilecektir.

● 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.

● Teslimat durum ekranında yapılan işlemlere göre teslimat sırası 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.

● Harita İşlemleri için bir sınırlamaya gidilmemiştir. Google Maps Open Street Maps vb kullanabilirsiniz.

Bulut Platformunda

● Veri tabanı oluşturulacaktır. Veritabanı tablolarında KargoID, MüsteriID, MüsteriAdi, MüsteriLokasyon, vb. bilgileri tutulacaktır.

● Bulut platformu üzerinde geliştireceğiniz bir API ile masaüstü uygulamanızın haberleşmesi gerekmektedir.

● Sadece bulutta bulunan API'nizin arayüzünden kargo teslimat noktası girişi yapılabilecektir (Veri tabanına el ile teslimat adresi girişi yapılmayacaktır).

# 3.YÖNTEM

Programımızı gerçekleştirmek için öncelikle Google Cloud hesabı oluşturduk. Bunu bulut sistemde verilerimizi tutabilmek için yaptık. Google Cloud SQL ile bilgisayarımızda bulunan MySQL Workbench'i IP adressimizile birbirine bağladık. Böylelikle sorgu işlemlerimizi rahat bir şekilde gerçekleştirebildik.

Veri tabanı işlemlerinden sonra Java ile veri tabanını bağladık. Daha sonra Java Swing ile projemizin arayüzünü tasarlamaya başladık. Giriş sayfasında kargocu kullanıcı adı ve şifresiyle sisteme giriş yapabilir. Eğer veri tabanında kayıtlı değilse "kayıt ol" sayfasına yönlendirilir; email, kullanıcı adı, şifre bilgilerini girdikten sonra tekrar "giriş yap" butonu aracılığıyla sisteme giriş yapabilir. Ayrıca kargocu şifre değişikliği yapabilir. Giriş yaptıktan sonra ekrana "KargoTakip" sayfası gelir. Kargocu busayfada "Kargo Ekle" butonu ile veritabanına kargo, "Teslimat Bilgilerini Gir" butonu ile bu kargonun musteriAd, musteriAdres gibi bilgilerini manuel olarak ekletebilir. "Kargo Sil" butonu ile kargoID bilgisi girerek dilediği kargoyu silebilir. Ekranda kargo ve müşteri bilgilerini dinamik olarak görebilir.Kargocu bu işlemleri gerçekleştirdikten sonra "Haritayı Görüntüle" butonu ile diğer GUI ekranına geçiş yapar**.**

JSON kütüphanesi yardımıyla veritabanından gelen konumların koordinatlarını aldık. Aldığımız koordinatları lokasyonlar matrixine kaydettik. Lokasyonlar matrixindeki elemanları mesafehesapla fonksiyonuna gönderip her iki konum arasındaki mesafeleri bulduk ve bulduğumuz mesafeleri adjacencymatrix isimli matrixe kaydettik. Bu matrixe ve konum dizimize tsp yöntemini uyguladık. TSP fonksiyonu ile Travelling Salesman Problem algoritmasını kullanarak konum dizisini hesaplanan rotaya göre sıralayıp adres.txt isimli text dosyasına yazdırdık. Jxbrowser kütüphanesi yardımıyla yeni bir tarayıcı tanımladık. Sayfayı yeniden yüklemeye yarayan reload fonksiyonunu kullanan reload button tanımladık. Google Maps Geocoding ve Directions api kullanarak haritamızı oluşturduk. Javascriptte yazdığımız kod ile adres.txt dosyasını okuyup adresleri eleman olarak atadık. Elemanları ve directions apiyi kullanarak harita üzerinden rota çizdik. Çizdiğimiz rota üzerinde kargo teslimi yapmayı ve teslim edilen kargonun işaretini kaldırmayı amaçlamıştık ancak süre yetmezliğinden dolayı projemizi burada bırakmak zorunda kaldık.

# 4.YALANCI KOD

Yalancı Kod

GirisEkrani{

Veri tabanı bağlantısı yap

GirisEkrani arayüzünü görünür yap

Kullanici adi ve şifresi bilgileri doğru girilirse

Bilgileri veri tabanına kaydet ve "TeslimatDurumEkranı" sayfasını görünür yap

Şifre değiştir butonuna tıklanırsa

Kullanıcının şifresini girdiği yeni şifre ile değiştir

Eğer kullanıcı kayıtlarda olmadığı için giriş yapamıyorsa

"Giriş yapmak için kayit olunuz!" mesajini göster

}

KayıtEkrani{

KayitEkrani arayüzünü görünür yap

Kullanicidan kullanici adi, şifre ve email bilgileri al

"Kayıt ol" butonu ile kullanıcıyı veri tabanına kaydet

}

TeslimatDurumEkrani{

TeslimatDurumEkrani arayüzünü görünür yap

Teslimat tablosunu dinamik olarak göster

Kullanici "Kargo Ekle" butonuna basarsa

Kullanicidan kargonun kargoID'sini al ve kargoyu veri tabanından sil

"Teslimat Bilgilerini Gir" butonuna basarsa

Kullanicidan musteriAd, musteriAdres gibi bilgileri al ve veri tabanına kaydet

"Haritayı Göster" butonuna basarsa

"Menu" arayüzünü görünür yap

}

Menu{

Adresler[] dizisini tanımla

Bu diziye "AdresleriAl()" fonksiyonu ile veri tabanındaki musteriAdresleri ata

lokasyonlar[][] martisi tanımla

Döngü içerisinde{

Adres lokasyon bilgilerini almak için "https://maps.googleapis.com/maps/api/geocode/json?address=..+ adresler[]" url'sini tanımla

Url'nin adresleri alabilmesi için "url.openConnection().getInputStream();" kodunu çalıştır

JSon objesi yarat

Objeyle adresin enlem ve boylam bilgileri al

Enlem ve boylamları double lat ve lng değerlerine ata

Bu değerleri her bir adres için ayrı olacak şekilde lokasyonlar[][] matrisine ata

}

"mesafeHesapla(lokasyonlar)" fonksiyonu ile mesafeler[][] matrisini oluştur

TSPNearestNeighbour class'ından bir nesne yarat

Bu nesne ile "TSPNearestNeighbour(mesafeler,adresler)" fonksiyonunu çağır

"TSPNearestNeighbour(mesafeler,adresler)" fonksiyonu ile oluşan en kısa yolu "adres.txt" dosyasına yazdır

"open_site()" fonsiyonunu çalıştır

Bu fonksiyon ile "browser" isimli Browser objesi yarat

Bu obje ile "C:\\HTMLGmaps\\simple_map.html" url'sini yükle

"simple_map.html" dosyası ile

"adres.txt" metninden adresleri okut

Kullanici "Gönder" butonuna basarsa

Adresleri Google Map haritasında çizdir

"Reload" butonuna basarsa

Menü ekranını yeniden yükle

vbBaglanti {

Java.sql.Connection kütüphanesini import et

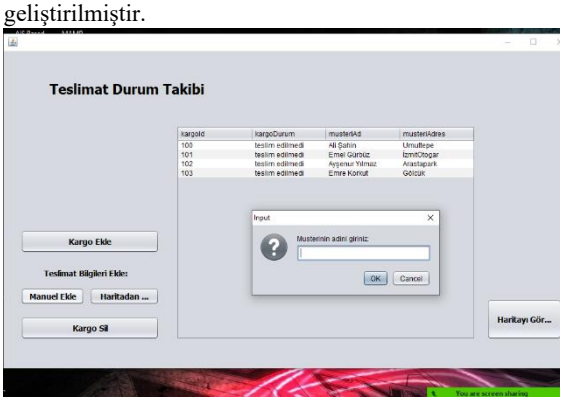
Local'de bulunan jdbc:mysql://34.132.88.53:3306/kargosistemi?user=root&password=1234

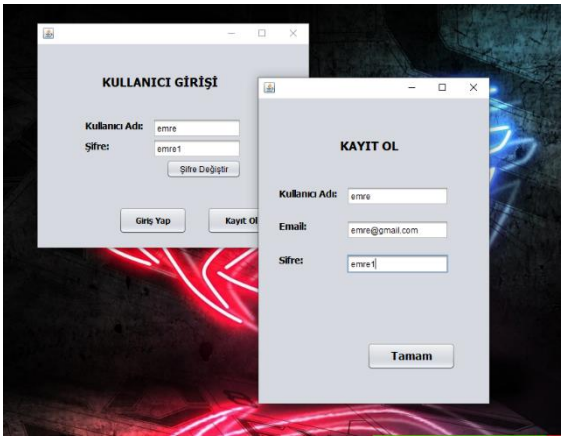
Veritabanına bağlanmayı dene

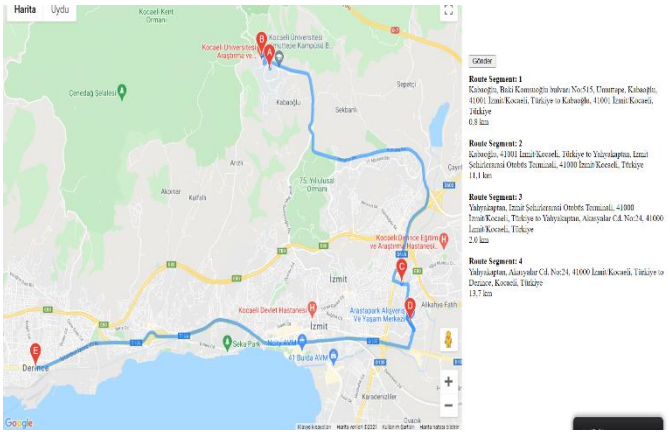
}

## 5.TASARIM

Proje arayüzümüz Java Swing kütüphanesi kullanılarak geliştirilmiştir.




## 6.SONUÇ

Kullanıcı kayıt olabilir , şifre değiştirebilir.Kullanıcı Müşteri ve Kargo teslim adresi ekleyebilir. Program kullanıcının eklediği adresleri kullanarak en kısa yol algoritmasıyla haritadan rota çizer ve kargo teslim adreslerini işaretler.

Kargo teslim sonucu çizilen rotada işaretlenen kargo adreslerinin kaldırılması istenmişti. Fakat süre olarak yetiştiremediğimiz için projeyi burada noktalamak zorunda kaldık.

## 7.KAYNAKÇA


https://cloud.google.com/sql/docs/mysql/quickstart

https://ichi.pro/tr/google-cloud-platform-kullanarak-mysql-ornegi-olusturma-58532728683758

https://www.youtube.com/watch?v=C1sMefyulCQ

https://www.mysqltutorial.org/mysql-delete-join/

https://www.yusufsezer.com.tr/sql-foreign-key/

https://www.ismailgursoy.com.tr/foreign-keys-olusturma/

https://www.youtube.com/watch?v=NgRSW7ZOjLs&t=609s

https://www.sanfoundry.com/java-program-implement-traveling-salesman-problem-using-nearest-neighbour-algorithm/

www.w3schools.com/js/default.asp

www.w3schools.com/html/default.asp

www.stackoverflow.com

www.developers.google.com

www.geeksforgeeks.org