KOCAELİ ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ

# YAZILIM LABORATUVARI I

# PROJE I

# Kargo Dağıtım Sistemi

ÜMMÜHAN TEPEBAŞ-180201088

–

MÜRVET NUR ŞEN - 190201097

## ÖZET:

## Bizden 2 farklı GUI ekranına sahip akıllı kargo dağıtım sistemi yapan bir masaüstü uygulaması geliştirmemiz beklenmektedir.

## Projemiz Apache NetBeans IDE 12.5 geliştirme ortamında Java dilinde, bulut ortamı olarak Firebase ve harita işlemleri için Google Maps kullanılarak hazırlanmıştır.

## GENEL BİLGİLER:

## Projemiz ilk GUI ortamı isterlerini;

## Register ekranı

## Login ekranı

## Storage ekranı

## Fetch ekranı

## ile sağlamaktadır.

## Register Ekranı:

## Bu sayfada yeni kullanıcı kaydını oluşturmak için hazırlanmıştır.

## Aldığı:

## Name

## Email

## Parola

## bilgileri ile Firebase Authentication sayfasında yeni “useruid” oluşturarak gerekli kayıt işlemlerini yapar.

## Login Ekranı:

## Bu sayfa kullanıcı girişleri için hazırlanmıştır.Aldığı:

## Name

## Email

## Parola

## Bilgileri ile Firebase Authentication sayfasında yeni “useruid” ye kayıtlı gerekli gerekli bilgileri doğrulayarak giriş işlemlerini karşılar.

## Store Database:

## Bu sayfada Firebase Realtime Database’e

## 2 adet veri girişi yapılıp, kaydedilebilir.

## Fetch Database:

## Bu sayfada ise Firebase Realtime Database’den “clicks” etiketi ile kaydedilen verileri eklenme zamanına göre ekrana getirir.

## KAZANIMLAR:

Bu projeyle beraber Google Maps Api,Firebase Realtime Database kullanımları ve Prim Algoritmasına daha çok aşina olduk.

Kaynaklara göre Prim, ağırlıklandırılmış ve bağlı bir [çizge](https://tr.wikipedia.org/wiki/%C3%87izge) üzerinde [minimum örten ağaç](https://tr.wikipedia.org/w/index.php?title=Minimum_%C3%B6rten_a%C4%9Fa%C3%A7&action=edit&redlink=1) (minimum spanning tree) problemine çözüm bulma algoritmalarından birisidir.Biz de minimum yol giderek kargoyu iletme aşamasında bunu kullandık.

**Google Maps Platform**, geliştiricilere Google Haritalar’ı sayfalarına entegre etmek, Google Haritalar’dan veri almak, basit kullanım veya kapsamlı kişiselleştirmelere olanak sağlayan bir Google ürünüdür.

A noktasından B noktasına giderken kullanacağı karayolları haritası ile **[rotalama](https://developers.google.com/maps/billing/understanding-cost-of-use" \l "directions" \t "_blank)** ile zaman, benzin yani maliyetten kazanç sağlamalarına yardımcı olabilir.

Firebase Realtime Database mimarisinde ise birçok client bulunmakta. Mobil , Web , Desktop vs. tüm istemciler aynı anda asenkron bir şekilde bilgi alışverişi yapabilmekte.

**PSEUDO KOD:**

Main.java{

//firebase init işlemleri gerçekleşir.

}

Menu.java{

//sırasıyla JButton1ActionPerformed() işlevi RegisterPanel()

//JButton2ActionPerformed() işlevi System.exit()

//JButton3ActionPerformed() işlevi LoginPanel()

//JButton5ActionPerformed() işlevi StoreDatabase()

//JButton6ActionPerformed() işlevi FetchDatabase()

//işlemelerini gerçekleştirir.}

RegisterPanel.java{

//name ,email ve password textfield() kullanarak alır

UserRecord userRecord = FirebaseAuth.getInstance().createUser(request);

kodu ile Firebase'e kullanıcı kaydeder.

}

LoginPanel.java{

//name ,email ve password textfield() kullanarak alır

UserRecord userRecord = FirebaseAuth.getInstance().createUser(request);

kodu ile Firebase'den doğrular.

}

StoreDatabase.java{

//datayı textfield() kullanarak alır

DatabaseReference ref = database.getReference().child("clicks/node/lat");

public StoreDatabase() {

initComponents();}

kodu ile Firebase'e veri kaydeder.

}

FetchDatabase.java{

//firebase e kaydedilen verileri GUI ekranına yansıtır.

Console ekranına kaydedilen koordinatları ve şehirleri Prims algoritması için oluşturulan class ve fonksiyonları kullanarak en kısa yolu verir.

}

distance.java{

//Maps API den tıklayarak elde edilen koordinatlar arası mesafeyi bulur.

}

PGraph.java{

//Prim algoritması çalışır.

}

tutorials\_denme.html{

//Maps API için gerekli HTML kodu javaya gömülür.}

## KULLANILAN KÜTÜPHANELER:

* com.google.firebase.database.Da taSnapshot
* com.google.firebase.database.DatabaseError
* com.google.firebase.database.DatabaseReference
* com.google.firebase.database.FirebaseDatabase
* com.google.firebase.database.ValueEventListener
* com.google.firebase.auth.FirebaseAuth
* com.google.firebase.auth.FirebaseAuthException
* com.google.firebase.auth.UserRecord
* com.google.firebase.auth.UserRecord.CreateRequest
* com.google.firebase.database.snapshot.Node
* com.formdev.flatlaf.FlatLightLaf
* com.kingaspx.FirebaseProjects.util.FirebaseConfig
* java.awt.List
* java.util.ArrayList
* java.util.logging.Level
* java.util.logging.Logger
* javax.swing.JOptionPane
* java.util.HashSet
* java.util.Set

**KULLANILAN FONKSİYONLAR:**

* initComponents()
* jButton1ActionPerformed(java.awt.event.ActionEvent evt)
* onDataChange(DataSnapshot dataSnapshot)
* onCancelled(DatabaseError databaseError)
* child\_fieldActionPerformed(java.awt.event.ActionEvent evt)
* addNode(Node nodeA)
* getNodes() / setNodes()
* getDataByEmail()
* recoverUserData(String uid)

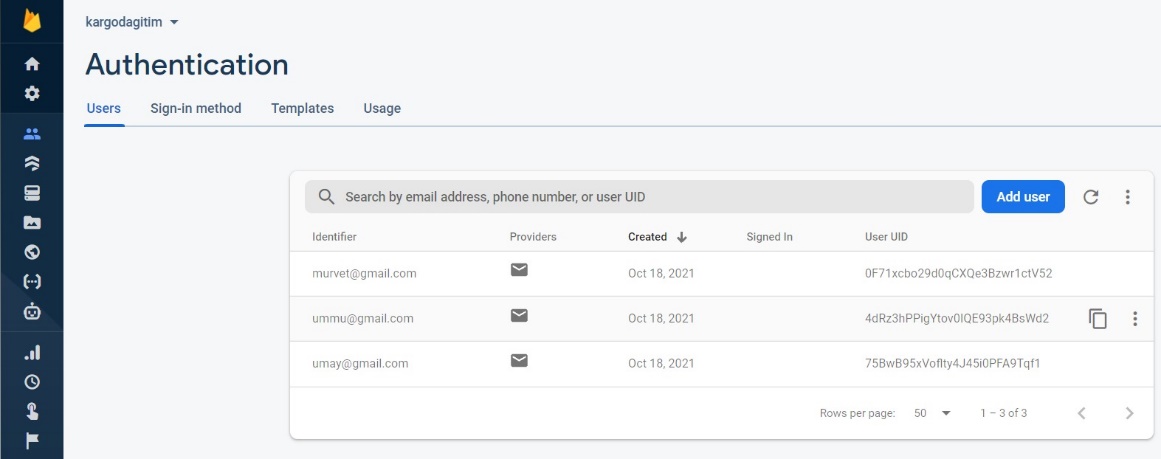
## PROJE GENEL GÖRÜNÜM:



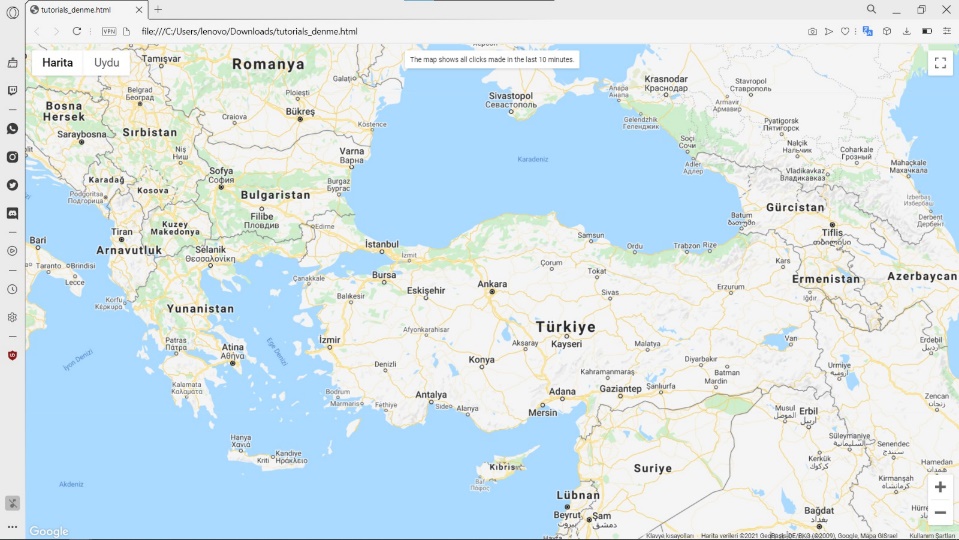
Java Gui ile Register seçeneğinde kullanıcı kaydı isim mail adresi ve şifre ile yapılıyor.



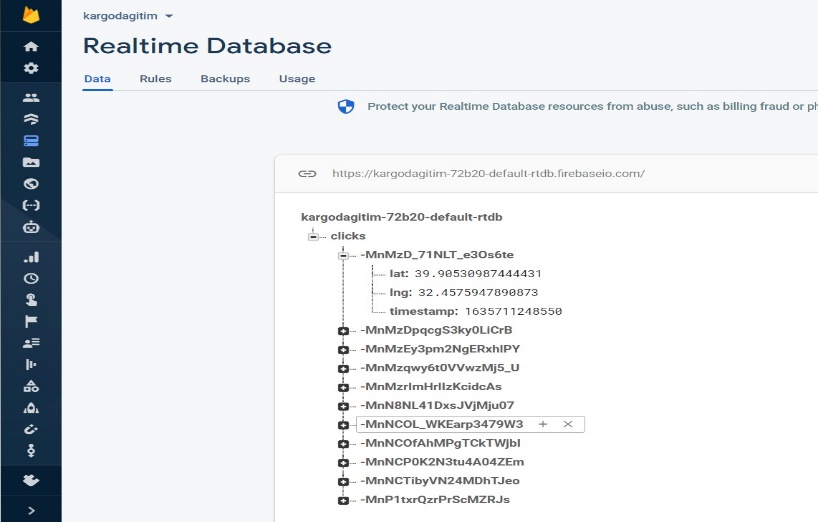
Login sayfasında ise kaydolan kullanıcı girişini yapabiliyor.



Her kullanıcı mailleriyle birlikte Firebase’de saklanıyor.



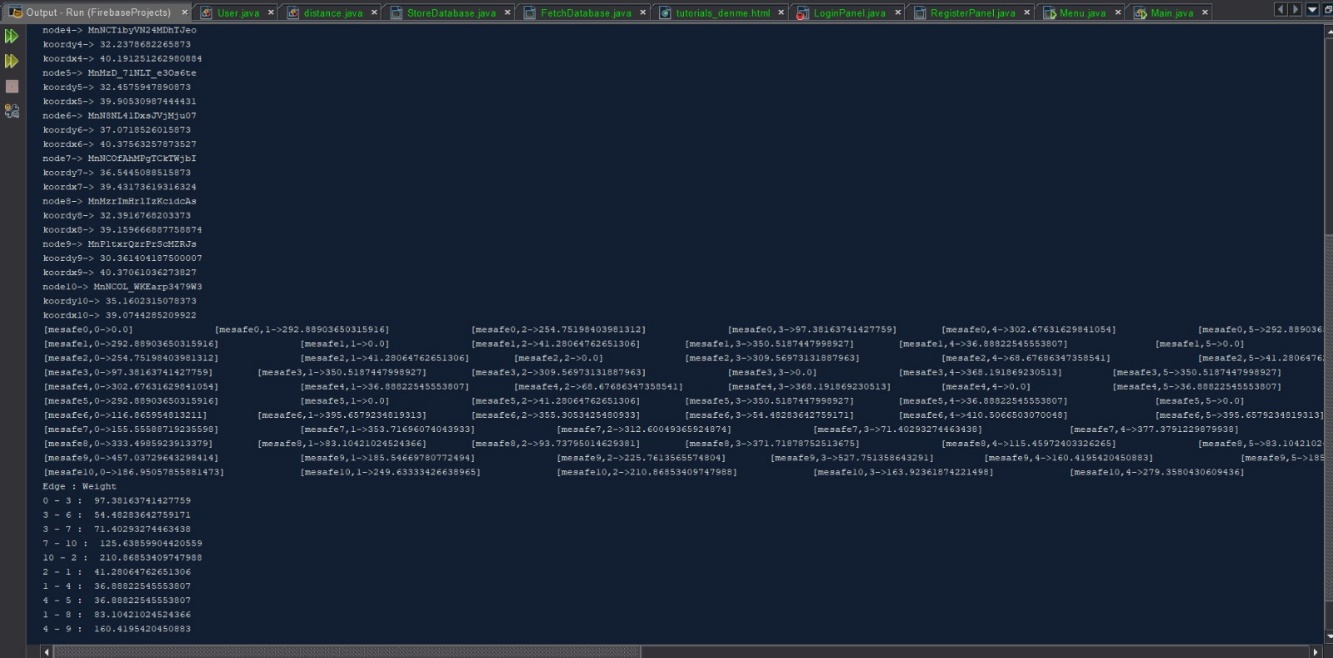
Google maps api kullanarak html’in içine gömdüğümüz javascript kodlarıyla tıklamalar alınıyor.



Tıklanan koordinatlar timestampleriyle beraber Firebase Realtime Database’e kaydoluyor.



Database’e kaydolan tıklama bilgileri java arayüzündeki fetch seçeneğiyle kullanıcı çekebiliyor.



Ardından çektiğimiz bu içeriği pars ederek koordinatlara(latitude,longitude) ulaşıyoruz. Ulaştığımız koordinatları Prim’s algoritmasına sokup tıklanan koordinatlar arasında kargoyu dağıtabilmek için en kısa rotayı oluşturuyoruz.

**KAYNAKÇA:**

* [**https://stackoverflow.com/questions/7701077/add-marker-function-with-google-maps-api**](https://stackoverflow.com/questions/7701077/add-marker-function-with-google-maps-api)
* [**https://stackoverflow.com/questions/12689084/how-to-use-google-geocoding-in-java**](https://stackoverflow.com/questions/12689084/how-to-use-google-geocoding-in-java)
* [**https://stackoverflow.com/questions/2296377/how-to-get-city-name-from-latitude-and-longitude-coordinates-in-google-maps**](https://stackoverflow.com/questions/2296377/how-to-get-city-name-from-latitude-and-longitude-coordinates-in-google-maps)
* [**https://stackoverflow.com/questions/53607082/using-google-maps-api-with-firebase**](https://stackoverflow.com/questions/53607082/using-google-maps-api-with-firebase)
* [**https://www.baeldung.com/java-prim-algorithm**](https://www.baeldung.com/java-prim-algorithm)
* [**https://www.baeldung.com/java-pairs**](https://www.baeldung.com/java-pairs)
* [**https://www.javatpoint.com/prims-algorithm-java#:~:text=Prim's%20algorithm%20in%20Java%20is,algorithms%20for%20Minimum%20Spanning%20Tree.&text=In%20prim's%20algorithm%2C%20we%20maintain,not%20present%20in%20the%20MSP**](https://www.javatpoint.com/prims-algorithm-java#:~:text=Prim's%20algorithm%20in%20Java%20is,algorithms%20for%20Minimum%20Spanning%20Tree.&text=In%20prim's%20algorithm%2C%20we%20maintain,not%20present%20in%20the%20MSP)**.**
* [**https://github.com/eugenp/tutorials/tree/master/algorithms-miscellaneous-5/src/main/resources**](https://github.com/eugenp/tutorials/tree/master/algorithms-miscellaneous-5/src/main/resources)
* [**https://www.youtube.com/watch?v=BkGtNBrOhKU&ab\_channel=SamCodes**](https://www.youtube.com/watch?v=BkGtNBrOhKU&ab_channel=SamCodes)
* [**https://developers.google.com/maps/documentation/distance-matrix/overview**](https://developers.google.com/maps/documentation/distance-matrix/overview)
* [**https://stackoverflow.com/questions/29680904/use-dijkstras-algorithm-to-get-shortest-path-when-distance-are-as-coordinates**](https://stackoverflow.com/questions/29680904/use-dijkstras-algorithm-to-get-shortest-path-when-distance-are-as-coordinates)
* [**https://www.vogella.com/tutorials/JavaAlgorithmsDijkstra/article.html**](https://www.vogella.com/tutorials/JavaAlgorithmsDijkstra/article.html)
* [**https://rosettacode.org/wiki/Haversine\_formula**](https://rosettacode.org/wiki/Haversine_formula)
* [**https://emrecelen.com.tr/javada-diziler/**](https://emrecelen.com.tr/javada-diziler/)
* [**https://www.aspsnippets.com/Articles/Google-Maps-V3-API-Calculate-distance-between-two-addresses-points-locations.aspx**](https://www.aspsnippets.com/Articles/Google-Maps-V3-API-Calculate-distance-between-two-addresses-points-locations.aspx)
* [**https://www.youtube.com/watch?v=hNrCp675A\_I&ab\_channel=AbnerRodrigues**](https://www.youtube.com/watch?v=hNrCp675A_I&ab_channel=AbnerRodrigues)
* [**https://developers.google.com/maps/documentation/javascript/examples/event-click-latlng**](https://developers.google.com/maps/documentation/javascript/examples/event-click-latlng)
* [**https://developers.google.com/maps/documentation/javascript/firebase?authuser=1**](https://developers.google.com/maps/documentation/javascript/firebase?authuser=1)
* [**https://www.baeldung.com/java-prim-algorithm**](https://www.baeldung.com/java-prim-algorithm)
* [**https://firebase.google.com/docs/admin/setup**](https://firebase.google.com/docs/admin/setup)
* [**https://firebase.google.com/docs/auth/admin**](https://firebase.google.com/docs/auth/admin)
* [**https://firebase.google.com/docs/database/admin/save-data**](https://firebase.google.com/docs/database/admin/save-data)
* [**https://firebase.google.com/docs/database/admin/retrieve-data**](https://firebase.google.com/docs/database/admin/retrieve-data)
* [**https://imperceptiblethoughts.com/shadow/getting-started/#default-java-groovy-tasks**](https://imperceptiblethoughts.com/shadow/getting-started/#default-java-groovy-tasks)