Kocaeli Üniversitesi

Bilgisayar Mühendisliği Bölümü

Yazılım Laboratuvarı II II.Proje

Rota Planlama Problemi

*Ömer Faruk Üçer-Tunay Baştürk*

[190201029@kocaeli.edu.tr-190201032@kocaeli.edu.tr](mailto:190201029@kocaeli.edu.tr-190201032@kocaeli.edu.tr)

***Projenin Özeti:***

Bu projede bizden uygulamaya kayıt olan öğrencilerin Kocaeli Üniversitesine gitmek için seçtiği duraktan belirlenen otobüs ve rotalarla okula gitmesi problemini minumum yol maliyeti ve yeterli sayıda araç kiralama ile en uygun rotayı sezgisel algoritma ile çizdirip öğrencileri okula götüren mobil uygulama yazılmasıydı.Otobüsler duraklarda bekleyen yolculara göre en uygun olacak şekilde belirlenen duraklardan kalkacak ve kapasitesi kadar öğrenciyi alıp önceden belirlenen rota ile okula getirecek.Otobüsler 25,30,40 kişilik kapasiteye sahiptir.Otobüslerin kapasitesinin bekleyen yolcu kapasitesinin altında kalması durumunda firma yeterli sayıda otobüs kiralayarak(kiralanan otobüslerin 25 yolcu kapasitesine sahiptir) gene minumum maliyet ile rotaları belirleyerek öğrencileri okula götürmesi gerekmektedir.Ayrıca sistemi kontrol eden admin istediği koordinatları ve durak ismini girerek yeni durak ekleyebilir,tüm durakları kaldırabilir ve duraklarda ne kadar yolcu beklediğini bir tablo şeklinde görebilir.

***1.GİRİŞ:***

Projede Mobil uygulama geliştirilmesi isteniyordu.Mobil Programlama dilleri arasında java,dart gibi dillerden Java programlama dilini seçtik.Bu dili seçmemizdeki en önemli etkenlerden biri bu dile daha önceden hakim olmamız,yararlanabileceğimiz kaynakları bilmemiz ve önceden mobil programlama için çalışırken bu dilden yararlanmamız oldu.Böylece Java dilini ve Android Studio IDE’sini seçerek kodlarımızı yazdık.Projeye yapmaya önceden öğrenmiş olduğumuz bilgileri kullanarak tasarımını ve sayfalar arası geçişleri ayarlayarak başladık.Sayfa tasarımı ve sayfalar arası geçişleri ayarladıktan sonra ugulamada kullanıcıları,durakları ve hangi kullanıcının hangi durağı tuttuğu bilgisini tutmak için veritabanı işlemlerini ayarlamaya başladık.Veritabanı işlemleri için yaptığımız araştırma sonunda SQLite Veritabanın mobil programlama için uygun olduğunu ve javayla birlikte kullanılabildiğini fark ettik ve Daha önce Pythonda kullanmış olduğumuz bu veritabanını bu seferde andoridde javada kullanmaya karar verdik.Diller arasındaki kullanım farkı olduğundan SQLite Veritabanını Javaya entegre etmek için internetten araştırmalar yaptık ve en sonunda SQLite veritabanını programımıza ekledik.Daha sonra Kullanıcılar,Duraklar ve hangi kullanıcının hangi durağı seçtiğine dair tablolar oluşturduk.Tablolar oluştuktan sonra kayıt yaparken tabloya ekleme yapmak için gerekli SQL kodunu,giriş yaparken kullanıcının kullanıcı adı ve şifresinin veritabanında kayıtlı olup olmadığını belirlemek için SQL kodunu,kullanıcı durağını değiştirdiği zaman gerekli güncellemeri yapan SQL kodunu,Adminin durakların hepsini silmesi için için gerekli SQL kodunu yazdık.Bundan sonrası sadece kontrollerle uygulamadaki gerekli yerleri hazırlamaktı.

Kullanıcının harita kısmını görebilmesi için Andorid Studio’nun bize sağladığı kolaylıklardan yararlandık.Android Studioda Activity açarken Google Map Activity açtık ve Android Studio bize default olarak Harita getirme kodunu yazan java class ve xml’ini getirdi.Biz de mobil için gerekli Google Map API Keyini koda yapıştırarak uygulamamıza eklemiş olduk.Daha sonra durakların yerlerini haritada göstermeyi markerlar ile yapacaktık fakat default markerlar yerine kendi belirlediğimiz otobüs ikonlarını kullandık.

***2.YÖNTEM:***

Yaptığımız Projede Algoritma ve Optimizasyon kısmı yoktur.Yaptığımız araştırmalara göre probleme uygun çözüm için algoritmayı bulamadık ve projeye uyaralayamadık.Projemiz sadece kullanıcı girişi,admin girişi,kullanıcı durak seçimi,adminin durağa yolcu eklemesini gerekirse durakları silmeyi ve Adminin Duraklardan kaç kişi bineceğini gösteren ekranı içeriyor.

Sistemde önceden bir Admin belirledik.Bu sayede sistemi kontrol edebilecek bir adminimiz var.

Program ilk açıldığında karşısına üç buton çıkar.Eğer kullanıcı sisteme kayıtlı değilse Register Butonuna tıklayarak Kayıt olma ekranına gider.Burada kullanıcı adı ve şifresini girerek sisteme kayıt olur.Eğer kullanıcı eksik bilgisi girdiyse veya öyle bir kullanıcı zaten sistemde varsa uygulama Toast mesajı ile uyarı döndürür.Eğer her şey eksiksiz ve doğruysa kullancı kayıt olduğunu ingilizce bir Toast mesajıyla alır.

Kullanıcı kayıt olduktan sonra en üstteki User Login butonuna tıklayarak giriş yapma sayfasına gelir.Kullanıcı bilgilerini yanlış girerse ya da girmeden butona tıklasa gene bir uyarı alır.Eğer kullanıcı bilgilerini doğru girerse bir sonraki sayfaya gider.Burada Kullanıcının önüne bir harita yüklenir ve haritada default olarak İzmit ilçesi gösterilir.Altta da bir dropdown liste ile kullanıcı sistemde kayıtlı olan tüm durakları görebilir.Buradan istediği ilçeyi seçtiği anda harita o durağa zoomlanır.Daha sonrasında kullanıcı eğer o durağı seçmek istiyorsa Add Station butonuna basarak durak ekli değilse ekleyebilir.Eğer kullanıcı daha öncesinde eklediği durağı tekrar eklemeye çalışıyorsa bir Toast uyarı ile uygulama tarafından o durağın zaten seçildiğine dair uyarılır.Kullanıcı başka bir durak seçmek isterse dropdown listeden başka bir ilçe seçip Change Station butonuna basabilir.

Uygulama ilk açıldığında admin kendi sayfasına giriş yapmak için Admin Login Butonuna tıklabilir.Bu butona tıkladıktan sonra Karşısına Kullanıcı girişindeki gibi bir ekran gelir ve adminin bu bilgileri doğru girmesi beklenir.

Admin bilgileri doğru girerse karşısına 3 tane buton kaydırmalı bir ekran şeklinde çıkar(Create New Station,Bus Manage System,Manage Passengers).Admin bu butonlardan Create New Stationa tıkladığında karşısına Enlem,Boylam ve Durak adı bilgilerini almak için 3 tane giriş alanı ve 2 buton bulunur.Admin mevcut durakları,duraklardaki yolcu sayılarını görmek için Show Stations butonuna tıklayabilir ve durakları yolcu sayılarıyla birlikte görebilir.Eğer admin yeni durak ekleyecekse enlem ve boylam bilgilerini ve durak adını eksiksiz doldurarak girip Add station butonuna basarak ekleyebilir.

İkinci buton olan Bus Manage System butonuna basınca hazırlanmış rotaları görecekti ancak biz algoritma ve optimizasyon kısmını yapamadğımız için bu buton işlevsiz oldu ve bir yere geçiş yaptırmıyor.

Son buton ise duraklara kullanıcı eklemek için bulunuyor.Bu butona tıklayan admin karşısına gelen ekranda dropdown liste ile istediği ilçeyi seçebilir.Daha sonra yolcu sayısını yazıp Add Passenger butonuna basarak yazdığı kullanıcı sayısı kadar o durağa ekleme yapabilir ya da Reset Passengers butonuna basarak durakta bulunan tüm yolcuları kaldırabilir.

***Kullanıcı ve Durak Tablosu Oluşturma Kodu***

@Override  
public void onCreate(SQLiteDatabase sqLiteDatabase) {  
 sqLiteDatabase.execSQL("create table users(username TEXT primary key,password TEXT,admin TEXT,station\_name TEXT)");  
 sqLiteDatabase.execSQL("create table station(station\_name TEXT primary key,latitude TEXT,longitude TEXT,passengers INTEGER)");  
}

***Tablo Zaten varsa ve tekrar açılmak isteniyorsa***

@Override  
public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {  
 sqLiteDatabase.execSQL("drop table if exists users");  
 sqLiteDatabase.execSQL("drop table if exists station");  
}

***Kullanıcı Sisteme Kayıt Olduğunda Bilgilerini Tabloya Ekleyen Fonksiyon***

public Boolean insertData(String username,String password,String admin,String station\_name){  
 SQLiteDatabase db=this.getWritableDatabase();  
 ContentValues values=new ContentValues();  
 values.put("username",username);  
 values.put("password",password);  
 values.put("admin",admin);  
 values.put("station\_name",station\_name);  
  
 long result=db.insert("users",null,values);  
 if (result==-1)  
 return false;  
 else  
 return true;  
  
}

***Adminin Durak Bilgisi Eklemesini Sağlayan Fonksiyon***

public Boolean insert\_station\_Data(String station\_name,String latitude,String longitude,int num\_passenger){  
 SQLiteDatabase db=this.getWritableDatabase();  
  
 values.put("station\_name",station\_name);  
 values.put("latitude",latitude);  
 values.put("longitude",longitude);  
 values.put("passengers",num\_passenger);  
 long result=db.insert("station",null,values);  
 if (result==-1)  
 return false;  
 else  
 return true;  
}

***Enlem ve Boylam Değerlerini Getiren Fonksiyonlar***

public ArrayList<String> get\_Latitude(){  
 ArrayList<String> station\_Latitude = new ArrayList<>();  
 SQLiteDatabase db=this.getReadableDatabase();  
 Cursor cursor=db.rawQuery("select latitude from station ",null);  
  
  
 while(cursor.moveToNext()) {  
 String data=cursor.getString(cursor.getColumnIndexOrThrow("latitude"));  
 station\_Latitude.add(data);  
 }  
 cursor.close();  
 return station\_Latitude;  
}  
public ArrayList<String> get\_Longitude(){  
 ArrayList<String> station\_Longitude = new ArrayList<>();  
 SQLiteDatabase db=this.getReadableDatabase();  
 Cursor cursor=db.rawQuery("select longitude from station ",null);  
  
  
 while(cursor.moveToNext()) {  
 String data=cursor.getString(cursor.getColumnIndexOrThrow("longitude"));  
 station\_Longitude.add(data);  
 }  
 cursor.close();  
 return station\_Longitude;  
}

***Dropdown Listesi için Durak İsmini Getiren Fonksiyon***

public ArrayList<String> get\_Station\_Name(){  
 ArrayList<String> station\_name = new ArrayList<>();  
 SQLiteDatabase db=this.getReadableDatabase();  
 Cursor cursor=db.rawQuery("select station\_name from station ORDER BY rowid",null);  
  
  
 while(cursor.moveToNext()) {  
 String data=cursor.getString(cursor.getColumnIndexOrThrow("station\_name"));  
 station\_name.add(data);  
 }  
 cursor.close();  
 return station\_name;  
}

***Kullanıcının durağı seçtikten sonra duraktaki yolcu sayısını arttıran fonksiyon***

public void user\_add\_passenger(String inbound\_station){  
 System.*out*.println("Buraya giriyor mu? " + inbound\_station);  
 SQLiteDatabase db = this.getReadableDatabase();  
 SQLiteDatabase db2 = this.getWritableDatabase();  
 System.*out*.println("DB oluşturuyor mu? " + inbound\_station);  
 Cursor cursor = db.rawQuery("select passengers from station where station\_name=?", new String[]{inbound\_station});  
 System.*out*.println("Sorgu yapıyor mu? " + inbound\_station);  
 //String num\_of\_passenger\_str="";  
 int num\_of\_passenger = 0;  
 while (cursor.moveToNext()) {  
 num\_of\_passenger = cursor.getInt(cursor.getColumnIndexOrThrow("passengers"));  
 System.*out*.println("Giriyoor");  
 }  
 num\_of\_passenger += 1;  
 System.*out*.println("Passenger\_str=" + num\_of\_passenger);  
 values.put("passengers",String.*valueOf*(num\_of\_passenger));  
 db2.update("station", values, "station\_name=?", new String[]{String.*valueOf*(inbound\_station)});  
 //db2.execSQL("UPDATE station SET passengers=? WHERE station\_name=? ",new String[]{String.valueOf(num\_of\_passenger),inbound\_station});  
  
}

***Adminin Duraktaki Yolcuları Kaldırmasını Sağlayan Fonksiyon***

public void admin\_delete\_passenger(){  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.execSQL("UPDATE station SET passengers=? ",new String[]{String.*valueOf*(0)});  
 String holder="Not Selected";  
 String holder2="No";  
 db.execSQL("UPDATE users SET station\_name=? Where admin=? ",new String[]{holder,holder2});  
  
}

***Adminin istediği durağa istediği sayıda yolcu eklemesini sağlayan kod***

public void admin\_add\_passenger(String inbound\_station,int arriving\_passengers){  
 SQLiteDatabase db = this.getReadableDatabase();  
 SQLiteDatabase db2 = this.getWritableDatabase();  
 Cursor cursor = db.rawQuery("select passengers from station where station\_name=?", new String[]{inbound\_station});  
 int num\_of\_passenger = 0;  
 while (cursor.moveToNext()) {  
 num\_of\_passenger = cursor.getInt(cursor.getColumnIndexOrThrow("passengers"));  
 System.*out*.println("Giriyoor");  
 }  
 num\_of\_passenger+=arriving\_passengers;  
 db2.execSQL("UPDATE station SET passengers=? WHERE station\_name=? ",new String[]{String.*valueOf*(num\_of\_passenger),inbound\_station});   
}

***Kullanıcının Durağı var mı sorgusunu yapan fonksiyon***

public Boolean has\_user\_station(String username,String password){  
 System.*out*.println("Girdim mi ki :D");  
 String control="";  
 SQLiteDatabase db = this.getReadableDatabase();  
 Cursor cursor = db.rawQuery("select station\_name from users where username=? and password=?", new String[]{username,password});  
 System.*out*.println("geliyorum cursor");  
 while (cursor.moveToNext()) {  
 control= cursor.getString(cursor.getColumnIndexOrThrow("station\_name"));  
 System.*out*.println("Giriyoor "+control);  
 }  
 if (!control.contains("Not Selected")){  
 System.*out*.println("True");  
 return true;  
 }  
 else {  
 System.*out*.println("False");  
 return false;  
 }  
}

***Kullanıcının istediği durağı eklemesini sağlayan sorguuyu yapan fonksşyon***

public void user\_add\_station(String username,String password,String inbound\_station\_name){  
  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.execSQL("UPDATE users SET station\_name=? WHERE username=? and password=? ",new String[]{inbound\_station\_name,username,password});  
  
}

***Kullanıcının İstediği Durağı Değiştirmesini sağlayan sorguyu sağlayan fonksşyon***

public void user\_change\_station(String username,String password,String station\_change){  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.execSQL("UPDATE users SET station\_name=? WHERE username=? and password=? ",new String[]{station\_change,username,password});  
  
}

***Kullanıcının Durak Değiştirdikten sonra değiştirdiği duraktaki yolcu sayısını azaltan fonksiyon***

public void reduce\_previous\_station(String username , String password){  
 SQLiteDatabase db = this.getReadableDatabase();  
 SQLiteDatabase db2 = this.getWritableDatabase();  
 Cursor cursor = db.rawQuery("select station\_name from users where username=? and password=?", new String[]{username,password});  
 String previous\_station="";  
 while (cursor.moveToNext()) {  
 previous\_station= cursor.getString(cursor.getColumnIndexOrThrow("station\_name"));  
  
 }  
 int num\_of\_passengers=0;  
 Cursor cursor2=db.rawQuery("select passengers from station where station\_name=?", new String[]{previous\_station});  
 while (cursor2.moveToNext()) {  
 num\_of\_passengers= cursor2.getInt(cursor2.getColumnIndexOrThrow("passengers"));  
  
 }  
 num\_of\_passengers--;  
 db2.execSQL("UPDATE station SET passengers=? WHERE station\_name=? ",new String[]{String.*valueOf*(num\_of\_passengers),previous\_station});  
  
  
  
  
}

***Kullanıcı Adı ve Şifre Kontrolü Yapan fonksiyonlar***

public Boolean checkusername(String username){  
 SQLiteDatabase db=this.getWritableDatabase();  
 Cursor cursor=db.rawQuery("select \* from users where username=?",new String[]{username});  
 if (cursor.getCount()>0)  
 return true;  
 else  
 return false;  
}  
  
public Boolean checkusernamepassword(String username,String password,String admin){  
 SQLiteDatabase db=this.getWritableDatabase();  
 Cursor cursor=db.rawQuery("select \* from users where username=? and password=? and admin=?",new String[]{username,password,admin});  
 if (cursor.getCount()>0){  
 return true;  
 }  
 else{  
 return false;  
 }  
  
}

***Duraktaki yolcu sayılarını tutup listede saklayan fonksiyon***

public ArrayList<Integer> get\_Passengers\_Number(){  
 ArrayList<Integer> num\_of\_passenger = new ArrayList<>();  
 SQLiteDatabase db=this.getReadableDatabase();  
 Cursor cursor=db.rawQuery("select passengers from station ORDER BY rowid",null);  
  
  
 while(cursor.moveToNext()) {  
 int data=cursor.getInt(cursor.getColumnIndexOrThrow("passengers"));  
 num\_of\_passenger.add(data);  
 }  
 cursor.close();  
 return num\_of\_passenger;  
}

***Deneysel Yöntemler:***

A picture containing text, monitor, electronics, screen

Description automatically generated***Ana Sayfa Ekranı***

Uygulama ilk açıldığında açılan sayfadır.Kullanıcı butonlara tıklayarak giriş yapıp kayıt olabilir.

A close-up of a cell phone

Description automatically generated with medium confidence

***Kayıt Ekranı***

Kullanıcı burada gerekli bilgileri girerek sisteme kayıt olabilir.

A close-up of a cell phone

Description automatically generated with medium confidence***Kullanıcı Giriş Ekranı***

Kullanıcı kullanıcı adı ve şifresini girerek sisteme giriş yapabilir.

A close-up of a cell phone

Description automatically generated with low confidence***Kullanıcı Ana Sayfası***

Kullanıcı başarıyla giriş yaptıktan sonra listeden istediği durağı seçip o durağı ekleyebilir ya da seçtiği durağı değiştirebilir.

***Admin Giriş Ekranı***

A close-up of a cell phone

Description automatically generated with medium confidenceAdmin Gerekli bilgileri girerek kendi ana sayfası için giriş yapabilir.

***Admin Ana Sayfası***

A picture containing text, screenshot, monitor, electronics

Description automatically generatedA screenshot of a phone

Description automatically generated with low confidenceAdmin bu sayfada 3 tane sayfadan istediğine tıklayarak gidebilir.İkinci buton algoritma ve optimizasyon olmadığı için çalışmıyor

***Yeni Durak Ekleme Sayfası***

A screenshot of a phone

Description automatically generated with medium confidenceAdmin bu sayfada Durak adını,enlem ve boylamını girerek yeni durak ekleyebilir.Ayrıca Show Stations Butonuna basarak Durakları ve duraklardaki yolcu sayılarını görebilir.

***Durakları ve Duraklardaki Yolcu Sayısını Görüntüleme Sayfası***

A picture containing text, monitor, electronics, display

Description automatically generatedBu ekranda Admin durakları ve duraklardaki yolcu sayısını görebilir.

Graphical user interface, application

Description automatically generated

***Seçilen Durağa İstenilen Sayıda Yolcu Eklenmesi***

A screenshot of a phone

Description automatically generated with medium confidenceBu ekranda Admin Dropdown Listeden istediği ilçeyi seçerek ve istediği sayıyı yazarak o durağa istediği sayıda yolcu ekleyebilir.Reset Passengers butonu ile duraklara eklenmiş tüm yolcuları sıfırlar.

***Sonuç:***

Bu projenin algoritma kısmına vize haftasının gelmesi nedeniyle ve diğer ödevlerin de teslim tarihinin yaklaşması nedeniyle çok fazla zaman ayıramadık.Algoritmanın biraz zor olması nedeniyle ona uygun bir çözüm bulamadık ve bu konularda başarısız olduk.

Bu projeyle birlikte kendimizi sayfa tasarımı,veri tabanı tasarımı ve mobil veri tabanı bağlantıları konusunda geliştirdik.

Ayrıca projenin buraya kadar olan kısmında tam bir ekip uyumundaydık.Birimizin takıldığı yerde diğeri durumu çözecek şekilde uyumlu çalıştık.

***Akış Şeması***

Projemizde Algoritma kısmını yapamadığımız için ve Akış Şeması Algoritmaya göre olduğu için onu da yazamadık.

***Optimizasyon Yöntemi***

Projede Algoritmaya ve Optimizasyona dair bir şey bulunmadığından bu kısmı da oluşturamadık.

***Kaynakça:***

[***https://youtube.com/playlist?list=PLLTtZhOwITShnOaSJ6TvzfIeuZd76TNf\_***](https://youtube.com/playlist?list=PLLTtZhOwITShnOaSJ6TvzfIeuZd76TNf_)

<https://www.youtube.com/watch?v=hhPFAQLZlhg&t=188s&ab_channel=AmarnathPohanerkar>

<https://www.youtube.com/watch?v=9t8VVWebRFM&t=1099s&ab_channel=AllCodingTutorials>

<https://www.youtube.com/watch?v=_h86cSA2qd8&ab_channel=PankajPanjwani>

<https://www.youtube.com/watch?v=_6p7BqjNANQ&t=78s&ab_channel=CodinginFlow>

<https://stackoverflow.com/questions/29694341/android-getter-and-setter-database-sqlitehelper-class>

<https://stackoverflow.com/questions/12523005/how-set-background-drawable-programmatically-in-android>

<https://stackoverflow.com/questions/71675573/android-studio-sqlite-db-doesnt-update-a-specific-variable-in-table-with-query>

<https://www.youtube.com/watch?v=EBhmRaa8nhE&ab_channel=NekoCode>

<https://www.youtube.com/watch?v=sOJRJtM_iu0&ab_channel=EasyTuto>

<https://www.youtube.com/watch?v=yJ02XTKiuAc&t=986s&ab_channel=CodingWithMe>