Kocaeli Üniversitesi
Bilgisayar Mühendisliği Bölümü
Yazılım Lab.II
2020-2021 Bahar
Proje II

Ramazan Kaan Yarayan
190201138
İstanbul, Türkiye

***Özet*— Bulut bilişim ve google map api kullanarak android platformunda bir uygulama geliştirilmesi. Taksi gezinge (trajectory) verileri kullanılarak android platformunda farklı sorguların yapılabildiği bir uygulama geliştirilmesi.**

I. GIRIŞ

The New York City Taxi and Limousine Commission (TLC) sarı taksi, ye¸sil taksi, kiralık ara¸clarla ilgilenmektedir. TLC d¨uzenli olarak tamamlanan her taksi yolculu˘gu bilgilerini kaydetmektedir. Bu proje kapsamında Aralık 2020'de yayınlanan sarı taksi verisi kullanılmaktır.

Veriler Azure bulut servisi kullanılarak depolanmıştır. Depolanan veriler C# programlama dili kullanılarak yazılan bir Web API ile işlenmekte ve istenen durumlar için uygun çıktılar üretilmektedir. İşlenen veriler React-Native kullanılarak kullanıcıya sunulmaktadır.

II. İSTERLER

Tip 1

- En fazla yolcu taşınan 5 günü ve toplam yolcu sayılarını listeleyiniz.
- En uzun mesafeli 5 yolculuktaki gün ve mesafeleri listeleyiniz.

Tip 2

- Günlük seyahat başına düşen ortalama alınan ücretlere göre; en az ücret alınan iki tarih arasındaki günlük alınan ortalama ücretleri listeleyiniz.
- İki tarih arasında seyahat edilen en az mesafeli 5 yolculuk hangisidir? (Tarihler seçilebilmeli)

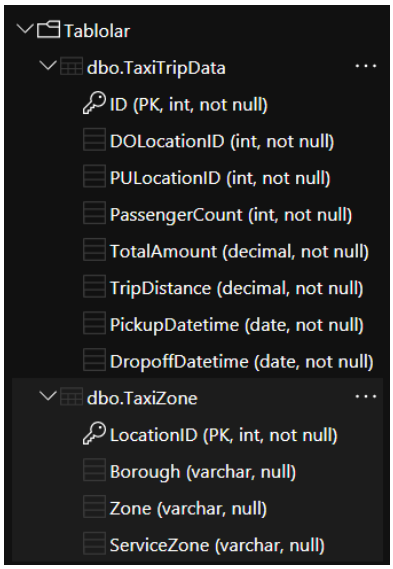
Tip 3

- En az 3 yolcunun bulunduğu seyahatlerden en kısa mesafeli ve en uzun mesafeli yolu çiziniz. Başlangıç ve varış konumları lokasyonun merkezi kabul edip mesafeye göre bir yol bulunmalıdır.

III. YÖNTEM

A. *Azure Bulut Servisi*

Veriler bulut servisi içerisinde SQL ile oluşturulmuş tablolarda tutulmaktadır. 2 adet tablo oluşturulmuştur. Tablo yapısı:



TaxiTripData tablosu:

- ID: Birincil anahtar, her bir kayıt için benzersiz değer alması için, 1'den başlayıp 1'er artarak devam etmektedir, her kayıt için otomatik oluşturulmaktadır.
- DOLocationID: Yolcunun taksiden indiği konumun ID değeri. (ID değerleri TaxiZone tablosundaki verilerle eşleşmektedir.)
- PULocationID: Yolcunun taksiye bindiği konumun ID değeri. (ID değerleri TaxiZone tablosundaki verilerle eşleşmektedir.)
- PassengerCount: Taksideki yolcu sayısı.
- TotalAmount: Yolcunun/yolcuların ödediği ücret.

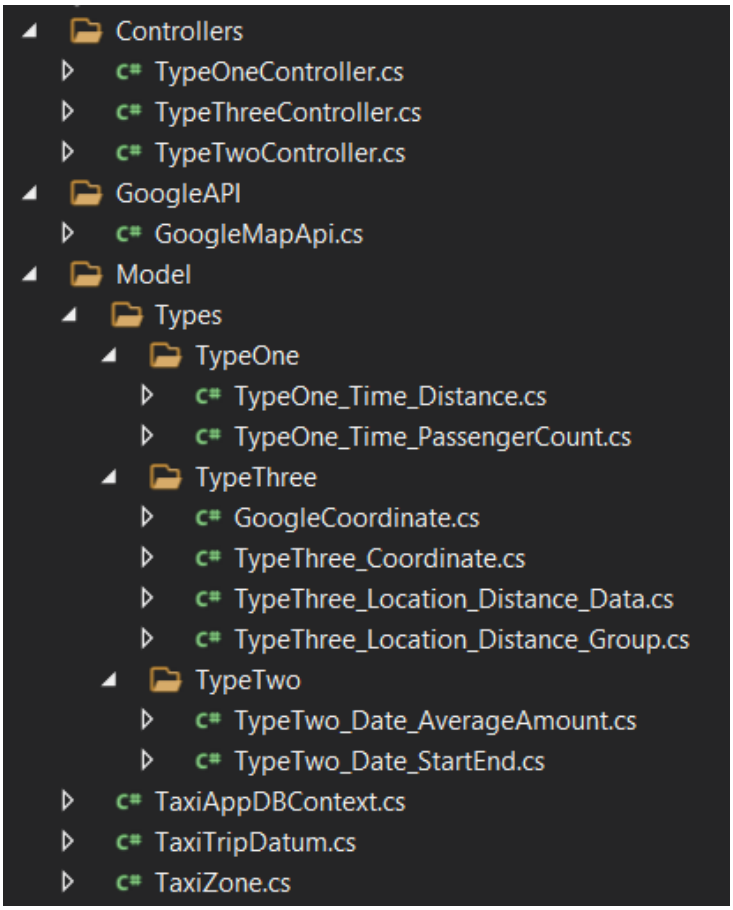
- TripDistance: Yolcunun/yolcuların toplam gittiği mesafe.
- PickupDatetime: Yolcunun/yolcuların taksiye bindiği tarih ve saat.
- DropoffDatetime: Yolcunun/yolcuların taksiden iniş yaptığı tarih ve saat.

TaxiZone tablosu:

- LocationID: Birincil anahtar, her bir kayıt için benzersiz değer alması için, 1'den başlayıp 1'er artarak devam etmektedir, her kayıt için otomatik oluşturulmaktadır.
- Borough: İlçe
- Zone: Bölge
- ServiceZone: Servis bölgesi

B. C# Web API

Web API' nin Azure hizmeti ile haberleşmesi için EntityFramework kullanılmıştır. Entity Framework ile isterler deki her bir tip için ayrı birer Controller oluşturulmuştur. Dönüş verileri için "Model" klasör içerisinde gerekli verileri kapsayacak sınıflar oluşturulmuştur.



1) Types Klasörü

a) TypeOne Klasörü

TypeOne_Time_Distance

```
public DateTime pickUpDatetime { get; set; }
public float tripDistance { get; set; }
```

TypeOne_Time_PassengerCount

```
public DateTime pickUpDatetime { get; set; }
public int passengerCount { get; set; }
```

b) TypeTwo Klasörü

TypeTwo_Date_AverageAmount

```
public DateTime pickUpDatetime { get; set; }
public float averageAmount { get; set; }
```

TypeTwo_Date_StartEnd

```
public DateTime start { get; set; }
public DateTime end { get; set; }
```

c) TypeThree Klasörü

GoogleCoordinate

```
public double lat { get; set; }
public double lng { get; set; }
```

TypeThree_Coordinate

```
public double Latitude { get; set; }
public double Longitude { get; set; }
```

TypeThree_Location_Distance_Data

```
public decimal Distance { get; set; }
public string PULocation { get; set; }
public string DOLocation { get; set; }
public TypeThree_Coordinate PULocationCoordinate { get; set; }
public TypeThree_Coordinate DOLocationCoordinate { get; set; }
```

TypeThree_Location_Distance_Group

```
public TypeThree_Location_Distance_Data shortRoute { get; set; }
public TypeThree_Location_Distance_Data longRoute { get; set; }
```

2) TypeOneController.cs

İki adet "HttpGet" fonksiyonu yazılmıştır.

- GetFirst():
  - TypeOne_Time_PassengerCount sınıfından bir liste oluşturuluyor.
  - TaxiTripData tablosunun içindeki veriler tarihlerine göre gruplanıyor.
  - Gruplanan veride, tarihlerin günlük yolcu sayılarının toplamı hesaplanıyor.
  - Veriler toplam yolcu sayılarına göre sıralanıyor.
  - Sıralanan verilerden ilk 5 tanesi alınıyor.
  - Alınan veriler, oluşturulan liste içerisine ekleniyor.
  - Elde edilen liste, fonksiyonun geri dönüş değeri olarak kullanılıyor.

- GetThird():
    - TypeOne_Time_Distance sınıfından bir liste oluşturuluyor
    - TaxiTripData tablosunun içindeki veriler yolculuk yapılan mesafeye göre büyükten küçüğe sıralanıyor.
    - Sıralanan veri içerisinden ilk 5 tanesinin gün ve mesafe değerleri alınıyor.
    - Alınan veri, oluşturulan listeye ekleniyor.
    - Oluşturulan liste, fonksiyonun geri dönüş değeri olarak kullanılıyor.

3) TypeTwoController.cs

İki adet "HttpGet" fonksiyonu yazılmıştır.

- GetSecond():
    - TypeTwo_Date_AverageAmount sınıfından bir liste oluşturuluyor.
    - TaxiTripData tablosunun içindeki veriler tarihlerine göre gruplanıyor.
    - Küçük olan tarih başlangıç değeri olarak atanıyor.
    - Gruplanan veri içerisinde, tarih aralığında olan veriler listeye ekleniyor.
    - Elde edilen liste, fonksiyonun geri dönüş değeri olarak kullanılıyor.
- GetThird():
    - Başlangıç ve bitiş tarih değerlerinden oluşan TypeTwo_Date_StartEnd sınıfından bir nesneyi parametre olarak alıyor.
    - TaxiTripData tablosu içerisinde, başlangıç ve son tarihleri arasında kalan verileri alıyor.
    - Alınan veriler mesafeye göre sıralanıyor.
    - Sıralanan verilerden ilk 5 tanesi fonksiyonun geri dönüş değeri olarak kullanılıyor.

4) TypeThreeController.cs

- GetThird():
    - TaxiTripData tablosundan yolcu sayısı 3 ve 3'ten büyük olan verileri alınıyor.
    - Mesafelere göre sıralanıp, ilk veri alınıyor.
    - TaxiZone tablosundan ID alanı eşleşen kayıt alınıyor.
    - TaxiZone tablosundan alınan kaydın verileri kullanılarak Google API' ye gönderilecek string bir değer oluşturuluyor.
    - Google API'ye gönderilen string değerinden dönen kordinatlar alınıyor.
    - İşlemler başlangıç noktasının kordinatları ve varış noktasının kordinatları için gerçekleştiriliyor.
    - Mesafe, başlangıç-varış kordinatları ve başlangıç-varış noktalarının isimlerini kapsayan bir sınıf içerisinde fonksiyonun geri dönüş değeri olarak kullanılıyor.

C. React-Native Expo

Kullanıcıya sunulacak sayfalar "screens" klasörü altında isterlerde belirtilen tiplere göre gruplanarak birbirinden ayrılmıştır.

App.js dosyası içerisinde, sayfalara yönlendirme yapılması için "StackNavigator" kullanılmıştır. Oluşturulan "Home" fonksiyonu ile bir giriş sayfası oluşturulmuş ve diğer sayfalara yönlendirmeler bu fonksiyon içerisinde gerçekleştirilmiştir.

1) Screens

a) TypeOne

TypeOne_First.js

Javascript dosyası içerisinde "getData" fonksiyonu içerisinde "fetch()" kullanılarak oluşturulan Web API'ye uygun ister için bir "GET" isteği gönderilip, gelen veri bir değişkene atılmıştır. Değişkenin içerisindeki veri "FlatList" kullanılarak kullanıcının kolaylıkla anlayabileceği bir formatta ekrana basılmıştır.

TypeOne_Third.js

Bir üstte anlatıldığı gibi Javascript dosyasyı içerisinden Web API'ye uygun istek atılarak, gelen veri kullanıcının kolaylıkla anlayabileceği bir şekilde ekrana basılmıştır.

b) TypeTwo

TypeTwo_Second.js

TypeOne_First.js ve TypeOne_Third.js dosyalarında olduğu gibi, Web API'ye uygun istek atılarak, gelen veri ekrana basılmıştır.

TypeTwo_Third.js

Üstte anlatılan sayfalardan farklı olarak, kullanıcıdan 2 adet tarih bilgisi alacak şekilde tasarlanmıştır.

React-native-community/datetimepicker kullanılmıştır. Kullanıcıdan alınacak değerler için state içerisinde değişkenler oluşturulmuştur.

Oluşturulan değişkenlere, datetimepicker için oluşturulan "onChange" fonksiyonu içerisinde atama yapılmıştır. Tarih bilgilerini daha düzgün ve anlaşılır bir formatta alabilmek için 2 adet fonksiyon yazılmıştır. Fonksiyonlar tarih verisinin içinden, Yıl-Ay-Gün formatında olacak şekilde tarihi alıp geri döndürmektedirler. Önceki kullanılan isteklerden farklı olarak burada Web API'ye bir POST atılmıştır. Atılan POST sonucunda dönene veri, aynı şekilde FlatList kullanılarak ekrana basılmıştır.

c) TypeThree

TypeThree_Third.js

Verileri harita üzerinde gösterebilmek için, react-native-maps ve react-native-maps-direction kullanılmıştır. En kısa mesafe, En uzun mesafe ve Google maps api key verilerini tutumak için bir state oluşturulmuştur. Web API'ye uygun istek atılarak, dönen veri değişkenlerin içerisine atılmıştır. Gelen verileri ekranda göstermek için; MapView içerisinde, MapViewDirection ve Marker kullanılmıştır. MapViewDirection harita üzerinde rotayı çizmek için kullanılmıştır. Marker çizilen rotanın başlangıç ve bitiş noktalarını anlaşılabilir şekilde göstermek için kullanılmıştır.

MapViewDirection içinde:

- origin: başlangıç konumu
- destination: varış konumu
- apikey: Google API anahatarı
- strokeWidth: çizginin kalınlığı
- strokeColor: çizginin rengi

atamaları yapılmıştır.

Marker içinde:

- pinColor: gösterge rengi
- coordinate: göstergenin kordinatı
- title: göstergenin üzerine basıldığınad gösterilecek başlık
- description: göstergenin açıklaması(başlangıç/varış noktası adı)

IV. SONUÇ

- Bulut bilişim hizmetlerinin depolama işlevlerinin kullanılması ve proje içine entegre edilmesi.
- C# ile Web API geliştirilmesi.
- EntityFramework kullanarak bulut hizmetindeki veritabanına bağlanma ve işlem gerçekleştirme.
- React-Native Expo yapısı ve kullanımı.
- React-Native Expo projesinde veri tutma ve listeleme.
- React-Native Expo projesi ile bir Web API hizmeti ile iletişim kurulması.
- React-Native Expo projesinde tarih bilgisi edinme.
- React-Native Expo projesinde MapView kullanımı.

KAYNAKLAR

[1] https://www.youtube.com/watch?v=U6SlmoXWf3o&t=734s (C# EntityFramework kullanımı)

[2] https://github.com/mmazzarolo/react-native-modal-datetime-picker (React-Native datetime picker kullanımı)

[3] https://github.com/react-native-maps/react-native-maps (React-Native maps kullanımı)

[4] https://github.com/bramus/react-native-maps-directions (React Native maps kullanımı)

[5] https://material.io/design/color/dark-theme.html#anatomy (Uygulama dizaynı)