

İSTANBUL KÜLTÜR ÜNİVERSİTESİ
FEN-EDEBİYAT FAKÜLTESİ
MATEMATİK VE BİLGİSAYAR BİLİMLERİ BÖLÜMÜ

REHBER BUL MOBİL UYGULAMASI

LİSANS BİTİRME PROJESİ
YİĞİT YAVUZ CEYLAN
1600003260

Danışman: Öğr. Gör. Aydın DEMİRİZ

HAZİRAN 2020

İKÜ, Fen-Edebiyat Fakültesi Matematik-Bilgisayar Bölümü'nün 1600003260 numaralı lisans öğrencisi Yiğit Yavuz Ceylan, Bitirme Projesi İlkelerinin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "REHBER BUL UYGULAMASI" başlıklı bitirme projesini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

JÜRİ ÜYELERİ:

Öğr.Gör.Aydın DEMİRİZ(Danışman)

Dr.Öğr.Üyesi Levent ÇUHACI

Dr.Öğr.Üyesi Bilge ŞİPAL

Proje Sunum Tarihi : 4 Haziran 2020

Önsöz

Bu proje mobil ve web alanında uygulanan en son teknolojilerin kullanımı ile gerçek bir deneyim kazanılmak için yapılmıştır. Bu proje ile birlikte hataların üstesinden gelip iş dünyasında kullanılan araçları kullanıp yeni fikirler öğrenilmiştir.

İçindekiler

Önsöz	iii
Kısaltmalar	vii
Özet	viii
1 React Native	1
1.1 React Nedir ?	1
1.2 React Native Nedir ?	2
1.3 React Native Nasıl Çalışır ?	2
1.4 Expo vs React Native CLI	3
1.5 React Native Alternatifleri	4
1.6 Redux	4
1.7 Firebase	5
2 Deneyimler	6
2.1 Neden React, React Native ?	6
2.1.1 Neden Firebase ?	8
2.2 Uygulama Ekran Geçişi	9
2.3 Kimlik Doğrulama	10
2.4 Tur Oluşturma Bileşeni	12
2.4.1 Girdi Kontrolü	15

2.5	Fotoğraf Bileşeni	16
2.5.1	Cihaz İzinleri	17
2.5.2	Fotoğrafları bir Depoda(Storage) Depolamak	18
3	Uygulama Ekran Görüntüleri	20
3.1	Giriş Sayfası	20
3.1.1	Profil Formu Sayfası	21
3.2	Ana Ekran Sayfası	22
3.2.1	Kategori Ekranı	22
3.2.2	Şehir Ekranı	23
3.2.3	Arama Ekranı	24
3.2.4	Arama Ekranı İşlevi	25
3.2.5	Kategoriye Göre Turlar Ekranı	26
3.2.6	Şehre Göre Turlar Ekranı	27
3.2.7	Filtreleme Ekranı	28
3.2.8	Tur Detay Ekranı	29
3.2.9	Tur Detay Ekranı Yapılacaklar ve Bilgiler	30
3.2.10	Tur Detay Ekranı İletişim Bilgileri ve Kişisel Bilgiler	31
3.2.11	Beğenilen Turlar Ekranı	32
3.3	Tur Oluşturma ve Profil Sayfası	33
3.3.1	Ayarlar Ekranı	33
3.3.2	Kişisel Bilgileri Değiştirme Ekranı	34
3.3.3	Rehber Ol Ara Ekranı	35
3.3.4	Tur Oluşturma Ekranı 1	36
3.3.5	Tur Oluşturma Ekranı 2	37
3.3.6	Tur Oluşturma Ekranı 3	38

3.3.7	Tur Oluřturma Ekranı 4	39
3.3.8	Sahip Olunan Turlar Ekranı	40
3.3.9	Tur Bilgilerini D�zenleme Ekranı	41
3.3.10	Fotoęraf Ekleme Ekranı	42
4	�zge�miř	43
	Kaynak�a	44

Kısaltmalar

API	: Application Programming Interface
CLI	: Command Line Interface
HTML	: Hyper Text Markup Language
HTTP	: Hypertext Transfer Protocol
REST	: Representational state transfer

Özet

Rehber Bul Uygulaması insanların daha önce gezmediği bir şehirde veya bir şehrin turistik yerlerini gezme ihtiyacı duyduğunda başvurabileceği bir uygulamadır. Uygulama, kayıtlı rehberlerin şehir ve kategoriye göre turlarını filtreleyip istenen şekildeki tura ulaşımı sağlıyor. Rehberler oluşturdukları turlar hakkında bilgiler, detaylar ve fotoğraf paylaşıyor. Kullanıcılar ise kendisine uygun bulduğu bir tura rahatça erişebilir.

Uygulama rehberler için tur oluşturup, düzenleyip turu silebiliyor. Kullanıcılar için şehir aratıp kategoriye göre filtreleyebiliyor. Beğendiği turları kendi beğenilenler sayfasına ekleyebiliyor.

Teknik açıdan bakılır ise, Firebase ile kimlik doğrulama gerçekleştirip veritabanında profil bilgilerini tur bilgilerini depolayabiliyor. Fotoğrafları Firebase Storage ile depolayıp veritabanında tutabiliyor ve bunların hepsini React Native ve Expo yardımı ile yapıyor.

Bölüm 1

React Native

1.1 React Nedir ?

React : Kullanıcı arayüzü oluşturmak için kullanılan bir Javascript kütüphanesidir. [3]

React daha kolay, daha hızlı olduğu için kullanılabilir. Fakat React kullanılmasıdaki en önemli sebep tekrar eden fakat içinde farklı bilgiler barındıran öğeleri sadece bir bileşen oluşturularak ve bu bileşeni farklı bilgiler ile özelleştirerek bütün kullanıcı arayüzünü oluşturabilmesidir. Her bileşenin kendi tasarımı, kendi fonksiyonları vardır. Bir diğer React kullanmanın avantajı da, bir bileşendeki bilgi güncellenirse, başka bileşenleri değiştirmeden sadece o bileşene özel olarak güncelleme yapılır. React kütüphanesi aslında onunla ne ürettiğimiz ile ilgilenmiyor. Çünkü react ile web uygulamaları oluştururken kullanılan, görselleştirmeden sorumlu olan React DOM adlı bir kütüphane vardır. Her React web projesinde React DOM görselleştirme (render) amacıyla çağrılır. Bu kütüphane HTML ögesine dönüştürülmesini sağlar. React ise sadece bileşen oluşturmada, bir şeyin değişip değişmediğini bulmada, bir şeyin yeniden görselleşmesi(re-render) gerekiyorsa ve veri aktarımında iyi bir kütüphanedir. Özet olarak React herhangi bir platformda kullanılabilir. Browser, HTML veya web uygulaması ile kısıtlı değildir. Web uygulamasında HTML kısmından React DOM sorumludur.

1.2 React Native Nedir ?

React Native, React'ten ayrı bir kütüphanedir. Özel bileşenlerin birleşiminden oluşur. Bu bileşenlerin özel olma sebebi React Native bu bileşenleri derleyebiliyor ve bu sayede bileşenler iOS ve Android için birer araç haline geliyor. Bu yönüyle React Native, React DOM'a benziyor. Derlenebilir bir çok bileşen ile kullanıcı arayüzleri oluşturmamıza olanak sağlıyor. React Native ayrıca cihaz kamerası gibi doğal platform Uygulama Programı Arayüzlerine (APIs) erişim sağlar. React Native, Javascript dili ile doğal platform dili arasında bağlantı kurmak için araçlar sunar. Çünkü React Native uygulamaları çoğunlukla Javascript dili ile yazılır. [3] [6]

$$React.js + ReactNative = RealNativeMobileApps$$

1.3 React Native Nasıl Çalışır ?

React for Web	Android	iOS	React Native
<div >	android.view	UIView	<View >
<input >	EditText	UITextField	<TextInput >

Daha önceden bahsedildiği gibi React Native kullanıcı arayüzü derlenir iken diğer bütün Javascript kodları, Redux kullanımı, HTTP talebi, veri dönüştürülmesi gibi kodlar doğal koda derlenmez. Onun yerine React Native'in barındırdığı özel bir ip-likte(thread) çalışırlar. React Native'in barındırdığı ve başlattığı uygulamanın içinde ekstra küçük bir Javascript uygulaması çalışır. Bu ekstra uygulama Javascript kodlarını çalıştırır ve doğal platform ile iletişime geçer. Yani Javascript sanal makinesi(Virtual Machine), Doğal platform modulu/API ile bir köprü kurar. Sonuç olarak görünüm, yani kullanıcı arayüzü uygulama performansı için en önemli etkenlerden biri çünkü kullanıcı arayüzünü yeniden işlemek(re-render) ve derlenmesi React Native'i performansını güçlü kılan özelliklerdendir. [3] [6]

1.4 Expo vs React Native CLI

Expo CLI

- 3.Parti bedava bir yazılımdır.
- Birçok karmaşıklığı ortadan kaldırır çünkü React Native çok karmaşık olabilir.
- Özellikle cihaz özelliklerini(Kamera vb) kullanırken oldukça yararlı ve kolaylık sağlar.
- Dezavantajı, Expo içerisinde sınırlı kalmak.
- React Native CLI kullanırken sahip olunan kontrole erişemez.

React Native CLI

- React Native takımı ve topluluğu tarafından oluşturulmuştur.
- Sadece temel kurulumla sahiptir(Bare-Bone). Yani kullanmak için Android Studio ve Xcode yüklenmeli, ayrıca yapılandırma ve yönetme kendi kendine yapılmalı.
- Cihaz özelliklerini kullanmak için 3.Parti bir paket gerekli fakat bu paketin yüklenmesi karışık ve külfetli olabilir.
- Expo'ya kıyasla kolaylığı daha az.
- Avantajı, full kontrole sahip, her bir detay kontrol edilebilir.
- Android ve iOS kodu yazılıp Javascript ile bağdaştırılabilir.(Expo'da geçerli değildir.)

Her zaman Expo'dan React Native CLI'a veya tam tersine geçiş kolaylıkla sağlanabilir.

[6]

1.5 React Native Alternatifleri

React Native iOS, Android platformlarında iyi performans sağlama yeteneğine sahip, platformlar arası uygulamalar oluşturmak için kullanılan mobil yazılım geliştirme aracıdır. Kullanıcı arayüzü derler. Başlangıç için temel bileşenler kümesi sağlar. Genellikle kod yeniden kullanılabilir. Facebook, Airbnb gibi platformlar tarafından kullanılır.

Ionic Çapraz platformlar için uygulama geliştirir. Birden fazla platformda mobil uygulama yapılabilir. Yeniden kullanılabilir. Derleyicisi olmadığından kötü performansa sebep olabilir.

Flutter Google tarafından desteklenir. Dart dilini kullanır. Dart web, mobil ve masaüstü uygulamaları geliştirmek için kullanılan bir dildir. Yeniden kullanılabilir. Performans olarak iyi bir performansa sahiptir. Tasarım olarak varsayılan Material-Design kullanır. Kullanılan platforma göre araçlar değiştirilebilir.

Doğal Diller Yeniden kullanılabilirlik geçerli değildir. Java veya Kotlin ile Android, Swift veya ObjectiveC ile iOS uygulamaları yazılır. [6]

1.6 Redux

3 ekranlı bir uygulama düşünüldüğünde, React Native stack navigator çalışır. Bu prensip bir yığıt sistemidir. Eğer 1. ekrandan bir bilgi 3.ekrana göndermek istenirse bu bilgi ancak 2.ekran üzerinden yapılabilir. Bunu 3 ekran değil çok daha fazla sayıda ekran olarak düşünürsek bilgi aktarımı durumu zorlaşır ve bu durumda Redux devreye girer.

Redux, veri erişiminin tek bir yerden yönetilmesinin yanında, bilgi aktarımından doğan problemi de çözmek üzere tasarlanmıştır. Redux'ta bütün verileri store adı verilen bir yerde tutulur. Böylece her bileşendeki veya ekrandaki bilgi store denilen yerden ulaşılabilir. Veriler, tek bir sorgulama kaynağından (store) getirildiği için, hangi bilgi nereden gelmiş gibi bir veri takibine gerek kalmamış olur. Bu sayede Redux, kodu düzenlemiş olur. [6]

1.7 Firebase

Google Firebase; web ve mobil uygulamalarının sunucu tarafıyla geliştiricinin uğraşmasına gerek kalmadan kullanıcı giriş yetkilendirmeli ve verilerini gerçek zamanlı ve senkron bir şekilde tutulmasını sağlayan bir platformdur. React uygulaması direkt olarak bulut veri tabanına bağlanmak yerine, bir REST API ile iletişim halinde olur sonra veritabanına bağlanır. Bunun nedeni direkt veritabanına bağlanmak güvenlik sorunlarına yol açabilir.

Kimlik doğrulama adımlarının başlangıcı olarak uygulama sunucuya kimlik doğrulama bilgileri gönderir. Sunucu ise hatalı veya başarılı olduğuna dair bir cevap geri döndürür. Eğer uygulama bir web uygulaması olsaydı sunucu bir session depolayacaktı ve session key döndürecek. Tarayıcı bu kullanıcının uygulamayı kullandığını ve kimliğini doğruladığını öğrenecekti. Fakat mobil uygulamalarda bu sistem farklı gelişir. Çünkü sunucu ile RESTful API vb ile iletişime geçiliyor. Bu nedenle mobil uygulamalarda session'a ihtiyaç yoktur. Onun yerine token kullanılır. Sunucu bir token oluşturur. Bunun için belirli algoritmalar vardır ve sadece sunucunun sahip olduğu özel bir anahtar kullanılır. Sonra token uygulamaya iletilir. Token uygulamanın deposunda depolanır. Örneğin; Redux Storage(hafıza-memory). Bu token ile kullanıcının giriş yaptığı kimlik doğruladığı onaylanır. Eğer kullanıcı çıkış yaparsa token silinir.

Ayrıca token uygulamanın içindeki bileşenleri sadece giriş yapmış kullanıcıların kullanabilmesi için bir güvenliktir ve kullanıcının belirli bir süre giriş yapmış olarak kalmasını sağlayabilir. [2] [6] [7]

Bölüm 2

Deneyimler

2.1 Neden React, React Native ?

React bir Javascript kütüphanesidir. Web uygulamaları için en son çıkmış teknoloji ve son zamanların en revaçta programlama dilidir. React Native için de, React'in bir mobil derleyicisi olarak aynı şeylerden bahsedilebilir. Dünya üzerinde yapılan önemli anketlerde bir yazılımcının en çok iş bulabildiği alanların başında, web geliştirici ve mobil geliştirici gelmektedir. Son zamanların gelişen teknolojisi ile birlikte yapay zeka ve istatistik alanında da bir büyüme olduğu bilinse bile, web ve mobil uygulama yazılımları, programlama kültürünün en çok yer kaplayan alanlarıdır. Anketlerdeki en yüksek iş alanının web ve mobil olduğundan bahsettiğimize göre, bunu hangi teknolojiler ile sağladığını yine önemli anketlerde görebiliriz. Genel bir kullanılan teknoloji anketi sıralamasında Javascript yine son yılın en önde gelen programlama dilidir. Arkasında HTML/CSS, Python,SQL, Java gibi diller de yer almaktadır. Javascript'i destekleyen backend sistem ise Node.js yine en üst sırada yer almaktadır ki React'in de

Şekil 2.1: En çok sevilen frameworks



Şekil 2.2: Programlama



arkasında çalıştığı sistem Node.js'dir. Bugüne kadar Javascript bir çok framework ile birlikte kullanılmıştır. Bunların en popülerleri Angular, Vue, JQuery gibi frameworklerdir. Yine son yılın en çok kullanılan framework anketine baktığımızda React'in JQuery'nin ardından 2.sıraya kadar çok kısa bir sürede ulaşabildiği görülmektedir. Programcıların kullanmaktan zevk aldığı bir kütüphane olan React, hızla gelişen teknolojiye ayak uydurmakta güçlük çeken insanların dahi öğrenmekten zevk aldığı bir kütüphanedir.

React için anlatılanlar, React Native için de geçerlidir. Teknik farklılıklar olsa da çalışma prensipleri aynıdır. Her iki işletim sistemi(Android, iOS) için uygulama yazılabilen React Native, yine React gibi son zamanların gözde bir mobil uygulama platformudur. Birçok bileşenden oluştuğu için aslında yazılan kodlar düzenli bir şekilde sunulur. Her geçen gün gelişen bir platformdur. Temeli Javascript'e dayandığından dolayı ve React'e benzerliğinden dolayı da web uygulamaları geliştiren bir programcı için yazımı anlaşılabilir ve geçişi kolaydır.

Dezavantajlarından bahsetmek istersek, en başta yeni ve gelişen bir platform olduğundan dolayı 6 ay önce yazılan bir kod tekrar geri dönülüp güncellenmek zorunda bırakılabilir. Ve yine gelişen bir platform olduğundan dolayı kaynak ve sorun çözümü aranmak istenirse, bulunan çözüm eski versiyonlara ait olabilir ve bundan dolayı da kaynak bulunmasında sorunlar çıkabilir. Birçok paket yüklemek gerekebilir. Teknoloji güncellendikçe takip etmek zorlaşabilir. Hiç beklenmeyen, durduk yere çıkan hatalar ile de karşılaşılabilir, fakat bu projede hataların geneline çözüm bir şekilde bulunmuştur. Bu platformu kullanacak ve genel olarak bir geliştirici olmak isteyen biri kesinlikle araştırmacı olmalıdır.

Sonuç olarak React Native'in gelecekte var olması beklenen büyük bir şirketin geliştiricileri tarafından oluşturulan bir mobil uygulama platformu olarak, geleceğin teknolojisine en yakın ve yaklaşan bir platform olduğuna inanmaktayım. Zaman ilerledikçe birçok yeni platform ile karşılaşp teknolojinin hızına yetişmekte zorlansak bile, React Native'in de aynı hızda ilerleyeceğini düşünmekteyim. [5] [6] [7]

2.1.1 Neden Firebase ?

Firebase, fotoğraf depolama, kimlik bilgileri doğrulama, veri depolama ve daha bir çok işlev ile birlikte Google tarafından geliştirilen bir hizmettir. Yazılan uygulama için ekstra bir sunucu tarafında kod oluşturulmasının yerine, bu hizmet kullanılabilir. İyi bir dökümantasyona sahip olduğundan yazılımcı bir veritabanı sorgusu girmeden dökümantasyonu okuyarak veri girişi gerçekleştirebilir, kimlik doğrulama yapıp buna telefon numarası, email, sosyal medya hesapları gibi seçenekler ekleyebilir. Email doğrulama, email değiştirme, şifre sıfırlama gibi fonksiyonlara da sahiptir. Uygulama ile direkt olarak veritabanına erişim olmadığından güvenli sayılmaktadır. Veritabanı ve şifre korunması konuları bir programcı için en önemli konulardandır. Kullanıcı şifrelerinde en az 12 basamak, büyük küçük harf, sembol ve rakam olsa bile uygulamanın veritabanında güvenli bir şekilde barınmıyorsa, kullanıcı şifresi güvende olmaz. Veritabanında tutulan şifreleri önceden, şifreleme yöntemi ile korumaya çalışan insanlar, şu anda hash denilen bir sistemi uygulamaktadır. Bir uygulamanın veritabanı kötü yazılımlara maruz kalsa bile bu sistem, kötü yazılımların ne kadar güçlü olduğuna bağlı olarak uygulama veritabanını koruyabilir. Fakat bir uygulamaya en güvenli giriş sosyal medya hesapları ile yapılan girişlerdir. Google, Facebook gibi veritabanı güçlü olan internet devlerinin, manuel yazılmış ve orta seviyede güvenli sayılan sistemle şifrelenmiş veritabanlarından daha güvenli olduğu aşıkardır. Firebase bu güvenliğı hem Google'a dayanarak kendi sağlamakta hem de yazılımcıya bazı güvenlik önlemleri alması için bir sistem sunmaktadır. Ayrıca yazılımcıdan kullanılan veri boyutu kadar ücret talep etmektedir. [2] [6]

2.2 Uygulama Ekran Geçişı

React Native mobil uygulamada ekran geçişleri "react-navigation" paketiyle yapılır. Sistem bir yığıt sistemidir. Başlangıç olarak bir ekran bileşeni seçilir ve yığıt modeliyle ileri geçişler ona göre gerçekleşir. Ekran geçişinin temel çalışma prensibi böyledir. Şekil 2.21'de görüldüğü gibi ekranların oluşturulması gerçekleşir. Örneğin, 3.ekranda geri dönüldüğünde en başa dönme, ya da 1.ekrandan 3.ekrana geçiş gibi farklı geçişler de mevcuttur. [4] [6]

Şekil: 2.21 Stack Navigator

```
const SearchNavigator = createStackNavigator({
  Search: SearchScreen,
  SearchEngineS: {
    screen: SearchEngineScreen,
    navigationOptions: {
      headerShown: false
    }
  },
  Category: CategoryScreen,
  City: CityScreen,
  Detail: {
    screen: DetailScreen,
    navigationOptions: {
      tabBarVisible: false,
      headerShown: false
    }
  },
},
```

Yığıt sistemi dışında alt sekme çubuğu oluşturup ekran geçişi sağlanabilir veya çekmeli menu sistemi ile ekran geçişleri sağlanabilir.

2.3 Kimlik Doğrulama

Endpoint : [https://identitytoolkit.googleapis.com/v1/accounts:signUp?key=\[API_KEY\]](https://identitytoolkit.googleapis.com/v1/accounts:signUp?key=[API_KEY])

Method : POST

Varlık İsimleri : email, şifre, secureToken

Yukarıdaki bilgilere firebase dokümantasyonundan ulaşp, bu bilgiler Redux aracılığıyla uygulamaya çekilir. Burada bahsedilen token, kullanıcının giriş yaptığını onaylamak için kullanılır. Kullanıcı geçerli bir giriş yaptıysa uygulama ana ekran sayfasına ulaşır. Eğer kullanıcı yeni bir kullanıcı ise ve üye olmak istiyor ise, geçerli bilgileri girdiği takdirde bir profil oluşturma sayfasına yönlendirilir. Profil oluşturma sayfasında isim, soyisim, telefon numarası, cinsiyet ve kişisel fotoğraf gibi bilgileri doldurulması istenir. Bu form doldurulduktan sonra kullanıcı için bir Token oluşturulur. Artık kullanıcı bu tokeni kullanarak giriş yapabilecektir. Profil bilgileri ise kullanıcı id'si ile yani "uid" ile veritabanına kayıt edilir. Aşağıda örnek bir veritabanı bilgileri görülebilir. [2] [6]



Token ayrıca kullanıcının belirli bir süre içerisinde tekrar giriş yapmadan uygulamaya direkt olarak giriş yapmasını da sağlar. Bu süreye "expiryDate" denir. Bu uygulamada belirlenen süre varsayılan 1 saattir. Bu süre Firebase hizmetinin fonksiyonları ile değiştirilebilir, arttırılabilir veya azaltılabilir.

Şekil 2.3'deki isLoading bilgisi geçerli işlemlerin yapılıp yapılmadığını kontrol eder ve ona göre ilerler. Bu sistem uygulamanın çoğu yerinde görülebilir. Aynı şekilde setError ile hata var ise onun kontrolü yapılır.

Şekil 2.3: Uygulama Örnek Kodu

```
const authHandler = async () => {
  let action;
  let navigate;
  if (isSignUp) {
    action = authActions.signUp(
      formState.inputValues.email,
      formState.inputValues.password)
    navigate = "ApplyInfo"
  } else {
    action = authActions.login(
      formState.inputValues.email,
      formState.inputValues.password)
    navigate = "Main"
  }

  setError(null)
  setIsLoading(true);
  try {
    await dispatch(action)
    props.navigation.navigate(navigate)
  } catch (err) {
    setError(err.message)
    setIsLoading(false)
  }
}
```

2.4 Tur Oluřturma Bileřeni

Endpoint : 'https://rehber-2e983.firebaseio.com/tours.json?auth=*token*'

Method : POST

Body : tCityId, tCategoryId,ownerId:userId, phone, tourImage, tourName, time, language, city, category, price, tourPlan, groupSize, personalDetail, isNatural, isCultural, isPhotography, isNightlife

Tur oluřturabilmek iin kullanıcının giriř yapması gereklidir. Bunu Token'in geerlilięi ile saęlar. Eęer yeni bir tur oluřturmak yerine turu silmek istenirse metodu deęiřtirip "DELETE" yapmak ve buna uygun bir řekilde reducer ekranını doldurmak yeterli olacaktır. Eęer tur bilgileri dzenlenmek istenirse dzenlenmek istenen turun id'si ile tur sınıfındaki turları aratıp o id'nin var olduęu indeksi bulup istenen turun bilgilerini deęiřtirir. [2]

řekil 2.4



Actions kısmında yukarıdaki POST metodu uygulanırken, reducer kısmında tur sınıfından yeni bir yapı oluşturup, oluşturma tamamlandığında da bunu Redux'un getirdiği dispatch fonksiyonu ile veritabanına kaydeden sistem uygulanır.

Şekil 2.401

```
case CREATE_TOUR:
  const newTour = new Tour([
    actions.tourData.id,
    actions.tourData.tCityId,
    actions.tourData.tCategoryId,
    actions.tourData.ownerId,
    actions.tourData.phone,
    actions.tourData.tourImage,
    actions.tourData.tourName,
    actions.tourData.time,
    actions.tourData.language,
    actions.tourData.city,
    actions.tourData.category,
    actions.tourData.price,
    actions.tourData.tourPlan,
    actions.tourData.groupSize,
    actions.tourData.personalDetail,
    actions.tourData.isNatural,
    actions.tourData.isCultural,
    actions.tourData.isPhotography,
    actions.tourData.isNightlife
  ])
  return {
    ...state,
    tours: state.tours.concat(newTour),
    userTour: state.userTour.concat(newTour)
  }
```

Şekil 2.402’da da görüldüğü üzere NameInput isimli özel bir bileşen oluşturulmuştur. Bu bileşene girilen farklı bilgiler ile tek bir bileşeni defalarca kullanarak farklı sonuçlar elde edilmektedir. Bu bilgilere props denir. Örneğin id kısmına "price" girilip etiketi(label) de Fiyat olarak belirleyip ve diğer bilgileri değiştirip bu özel bileşen ile farklı bir girdi oluşturulabilir. React’in bileşen ve tekrar kullanılabilirliğine güzel bir örnektir.

Şekil 2.402

```
<NameInput
  id="tourName"
  label="Tur İsmi"
  errorText="Please enter a valid title"
  autoCapitalize="words"
  autoComplete={true}
  keyboardType="default"
  returnKeyType="next"
  onChange={inputChangeHandler}
  required
/>
```

2.4.1 Girdi Kontrolü

Birden çok bilginin girildiği bu girdilerde, eğer bir kontrol olmazsa fiyat yazan girdiye isim, isim yazan girdiye ise rakam girilebilir. Şekil 2.41 de görüldüğü üzere bir girdiye neler yazılıp neler yazılamayacağını kontrol eden bir fonksiyon yazılmıştır. Bu fonksiyon eğer uygulama geliştirilmek istenirse, "Validation.js" adlı paketi yükleyip geliştirilebilir. Validation paketi kullanılsaydı bunları manuel olarak yazmaya ihtiyaç duyulmayacaktı.

Şekil 2.41

```
const textChangeHandler = text => {  
  const numberRegex = /[0-9]/g  
  let isValid = true;  
  if (props.required && text.trim().length === 0) {  
    isValid = false  
  }  
  if (props.email && !emailRegex.test(text.toLowerCase())) {  
    isValid = false  
  }  
  if (props.min !== null && +text < props.min) {  
    isValid = false  
  }  
  if (props.max !== null && +text > props.max) {  
    isValid = false  
  }  
  if (props.minLength !== null && text.length < props.minLength) {  
    isValid = false  
  }  
  if (props.onlyNumber && !numberRegex.test(text)) {  
    isValid = false  
  }  
  dispatch({  
    type: INPUT_CHANGE,  
    value: text,  
    isValid: isValid  
  })  
}
```

2.5 Fotoğraf Bileşeni

Bu uygulamada cihaz kamerasına ulaşmak için veya cihaz galerisini açmak için bir expo paketi olan ImagePicker kullanılmıştır. ImagePicker.launchImageLibraryAsync fonksiyonu galeriyi açmaktadır. Alttaki girilen bilgiler fotoğrafı düzenlemeye olanak sağlar, fotoğrafın boyutlarını ve kalitesini belirler. image.uri kısmı ise fotoğrafın adresini geri döndürür. Başka bileşenlerde kullanılabilmesi için props kısmı kullanılmıştır. [1] [6]

Şekil 2.5

```
const takeImageHandler = async () => {
  const hasPermission = await verifyPermission()
  if (!hasPermission) {
    return;
  }
  const image = await ImagePicker.launchImageLibraryAsync({
    allowsEditing: true, //CROPS etc
    aspect: props.aspect,
    quality: 0.3 // 0-1
  })
  console.log(image.uri)
  setPickedImg(image.uri)
  if (!image.cancelled) {
    props.onImageTaken(image.uri)
  }
}
```


2.5.1 Cihaz İzinleri

Bir uygulama bir cihaz fonksiyonunu kullanabilmek için cihaz sahibinden izin istemelidir. Expo'nun bu paketinde Android işletim sistemine sahip cihazlar için Şekil 2.51'daki kodu yazmaya gerek olmadan kendi kendine izin sorulmaktadır. Fakat iOS sisteminde çalışabilmesi için bu izin kodunun yazılması gereklidir. Bu izin kodu yine Expo'nun özel bir pakedi olan "permission" pakedi ile gelir. Cihazsan kamera ve galeri kullanımı ile ilgili izin istenir. [1] [6]

Şekil 2.51

```
const ImgPicker = props => {  
  
  const [pickedImg, setPickedImg] = useState()  
  
  const verifyPermission = async () => {  
    const result = await Permissions.askAsync(Permissions.CAMERA,  
      Permissions.CAMERA_ROLL)  
    if (result.status !== "granted") {  
      Alert.alert("Insufficient permission!",  
        "You need to grant camera permissions to use this app.",  
        [{ text: "Okay" }])  
      return false;  
    }  
    return true;  
  }  
}
```

2.5.2 Fotoğrafları bir Depoda(Storage) Depolamak

Cihazın içinde "AsyncStorage" fonksiyonu ile saklanan token, userId ve expiryDate bilgisinden kullanıcı id bilgisi çekilir. Eğer bir fotoğraf değiştiriliyorsa eski fotoğraf depodan silinir. Eğer ki fotoğraf ilk kez konuluyorsa fotoğraf silme işlemi gerçekleşmez.

inputChangeHandler fonksiyonu uygulama içinde fotoğraf bilgisinin tutulduğu alanın güncellenmesini sağlar. isLoading fonksiyonu fotoğraf depoya yüklenene kadar mavi bir ekran döndürür. Fotoğraf yüklendiği zaman yine eski ekrana geri döndülür.

imagePath yüklenen fotoğrafın adresidir ve bu adres "/" işaretleri ile bölünür. split fonksiyonu bu fotoğraf adresinin en son kısmını, yani örneğin test.jpg gibi bir adresi fileName sabitine geri döndürür.

Ayrıca yüklenen fotoğraf fetch fonksiyonu ile çekilirken, put(blob) fonksiyonu ile depoya(storage)'a eklenir. Ref diye belirlenen sabit bu fotoğrafın depoya hangi dosyalardan geçerek, hangi yolu izleyerek kayıt edileceğini belirler.

Eğer fotoğraf belirli adrese doğru bir şekilde yüklenir ise, onResolve fonksiyonu çalışır ve normal ekran geri döner. Eğer fotoğraf yüklenemez ise onReject fonksiyonu çalışır. Fotoğraf büyük boyutlu bir fotoğraf ise yüklenmesi biraz zaman alabilir. Bu durumda ilk başta onReject fonksiyonu geri döndürülebilir. Fakat fotoğraf yüklemesini bitirdiği an onResolve fonksiyonu döndürülür ve her şey çalışır durumda olur.

Şekil 2.52

```

const onTakenHandler = useCallback(async (imagePath) => {

  const userData = await AsyncStorage.getItem("userData")
  const transformedData = JSON.parse(userData)
  const { token, userId, expiryDate } = transformedData

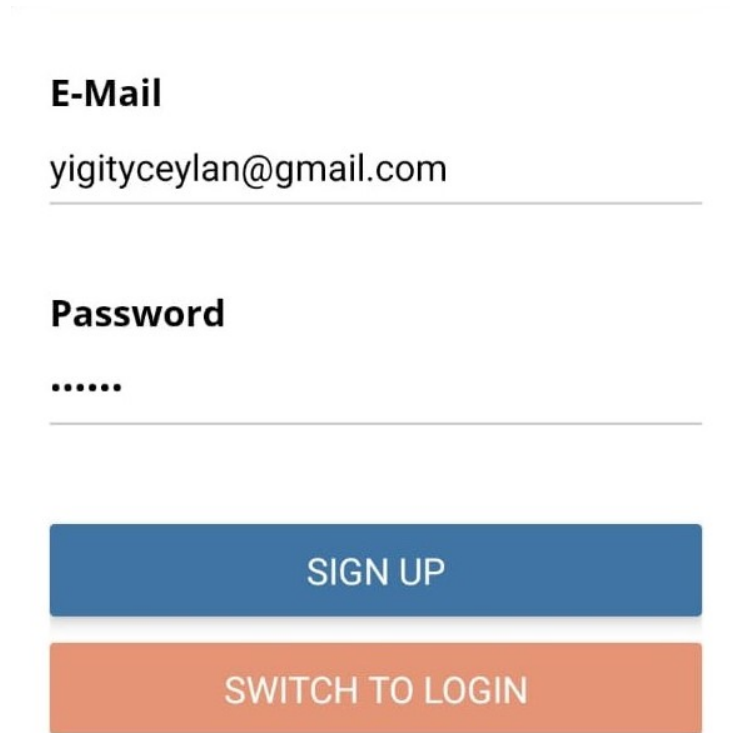
  let deleteFormStateImg = formState.inputValues.images
  if (deleteFormStateImg !== "") {
    let deleteImg = firebase.storage().refFromURL(`${deleteFormStateImg}`)
    deleteImg.delete()
  }
  inputChangeHandler("images", imagePath, true)
  setIsLoading(true)
  const fileName = imagePath.split('/').pop()
  const response = await fetch(imagePath)
  const blob = await response.blob()
  var ref = firebase.storage().ref().child(`images/${userId}/${tourId}/${fileName}`)
  setIsLoading(true)
  try {
    ref.put(blob)
    await firebase.storage().ref().child(`images/${userId}/${tourId}/${fileName}`).getDownloadURL().then(onResolve, onReject)
    function onResolve(foundURL){
      console.log(foundURL)
      setIsLoading(false)
    }
    function onReject(error){
      console.log(error)
      onResolve()
    }
  } catch (err) {
    throw err
  }
})

```

Bölüm 3

Uygulama Ekran Görüntüleri

3.1 Giriş Sayfası



The image shows a login form with two input fields. The first field is labeled 'E-Mail' and contains the text 'yigityceylan@gmail.com'. The second field is labeled 'Password' and contains six dots. Below the fields are two buttons: a blue button labeled 'SIGN UP' and an orange button labeled 'SWITCH TO LOGIN'.

E-Mail

yigityceylan@gmail.com

Password

.....

SIGN UP

SWITCH TO LOGIN

Şekil 3.1: Login Sayfası

3.1.1 Profil Formu Sayfası


ApplyInfoKaydet


İsim
Yiğit Yavuz

Soyad
Ceylan

Cinsiyet
Erkek ▼

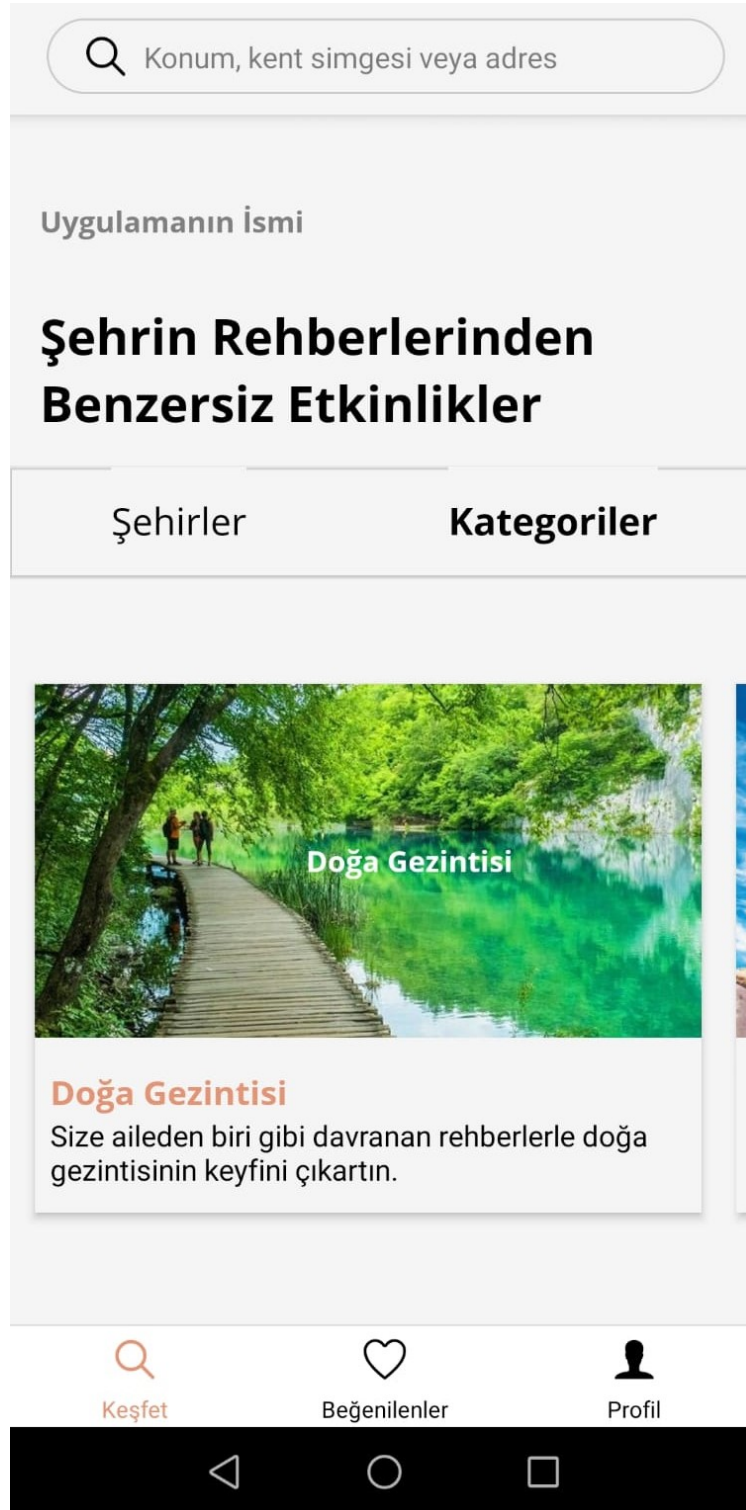
- Telefon Numarası
05568545588



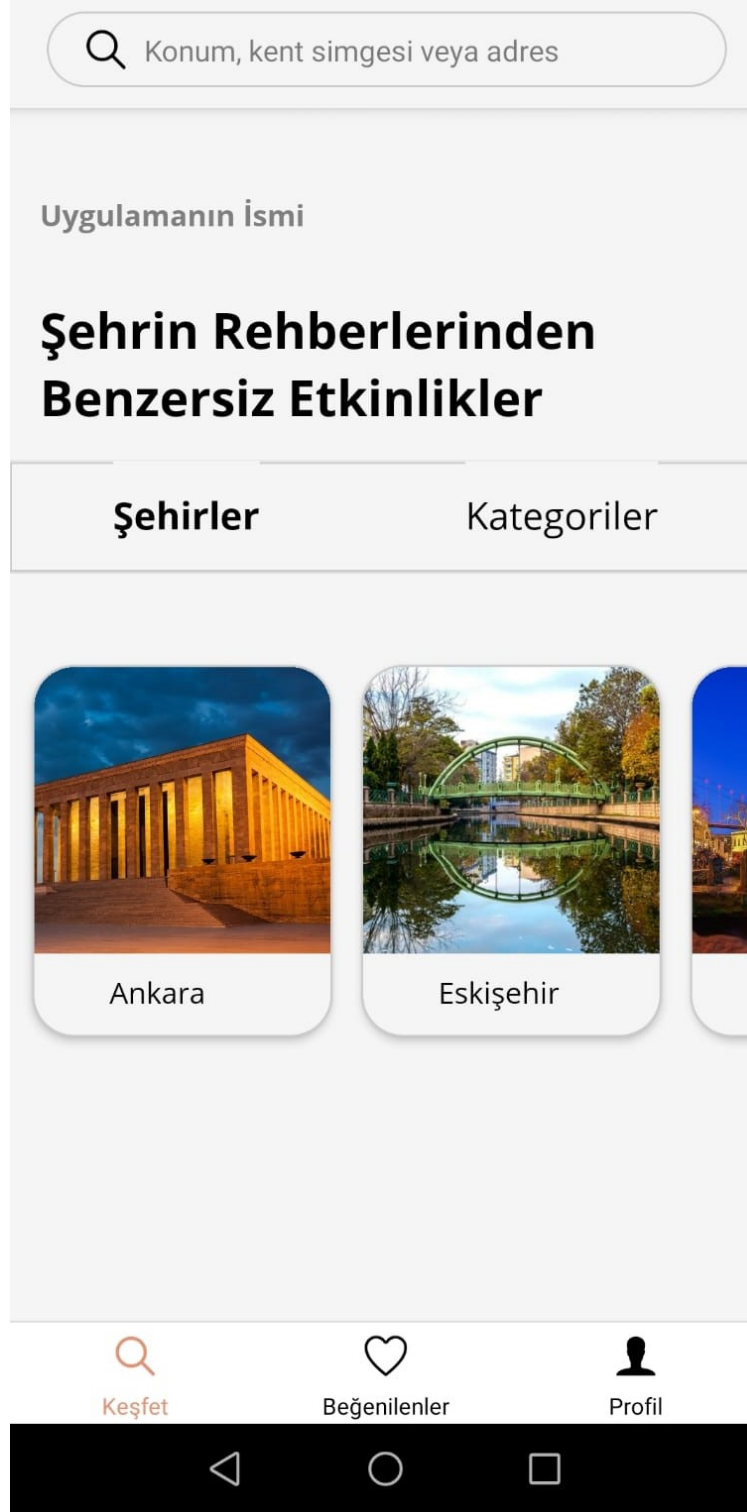


3.2 Ana Ekran Sayfası

3.2.1 Kategori Ekranı



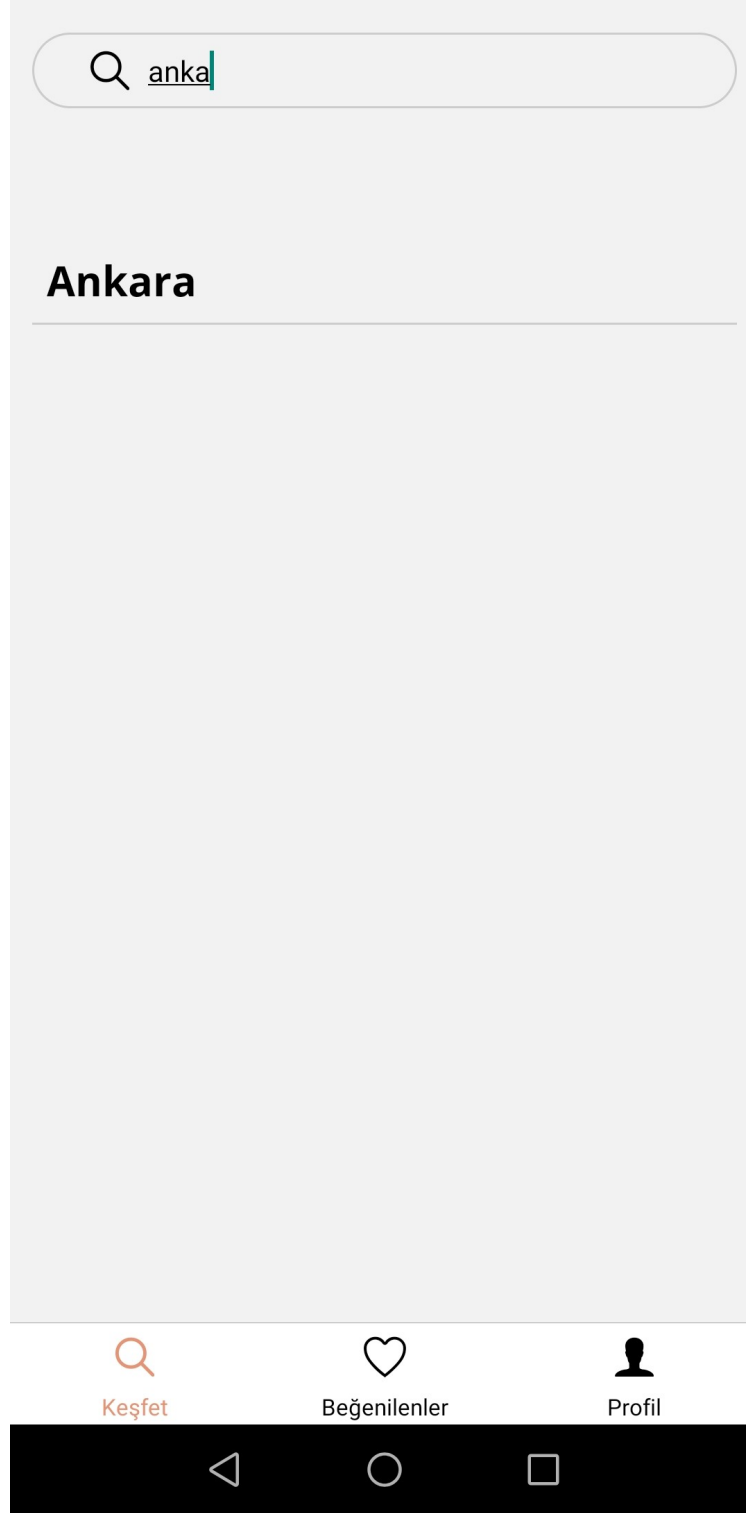
3.2.2 Şehir Ekranı



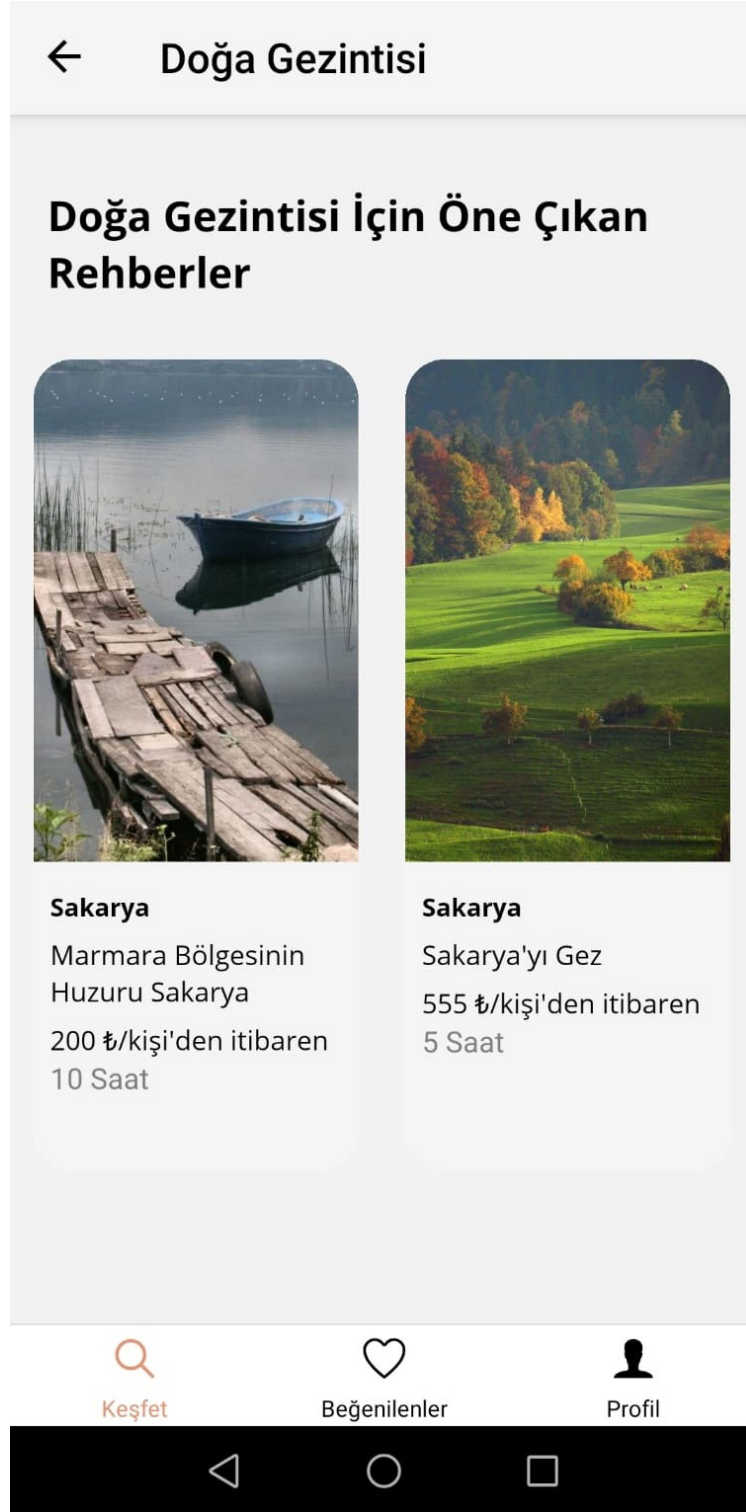
3.2.3 Arama Ekranı



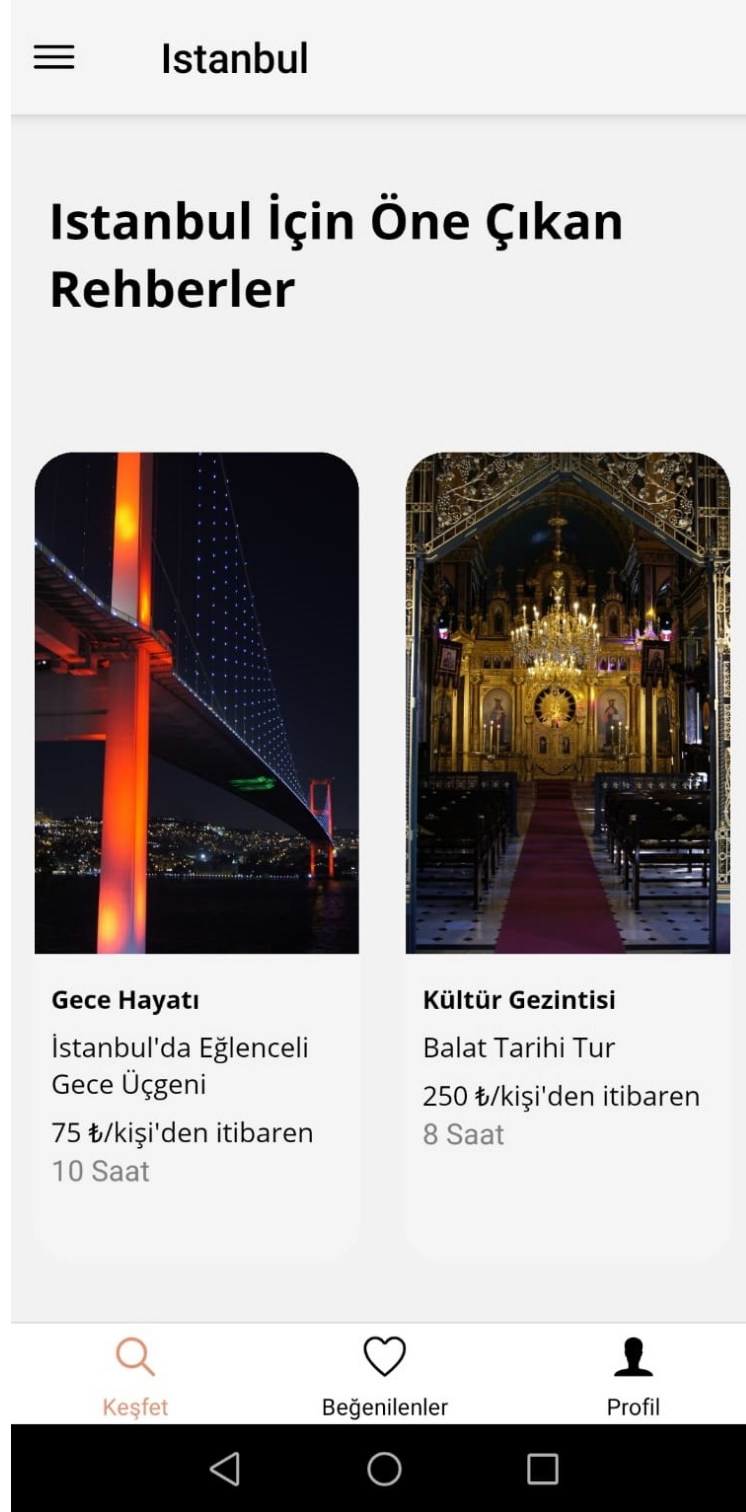
3.2.4 Arama Ekranı İşlevi



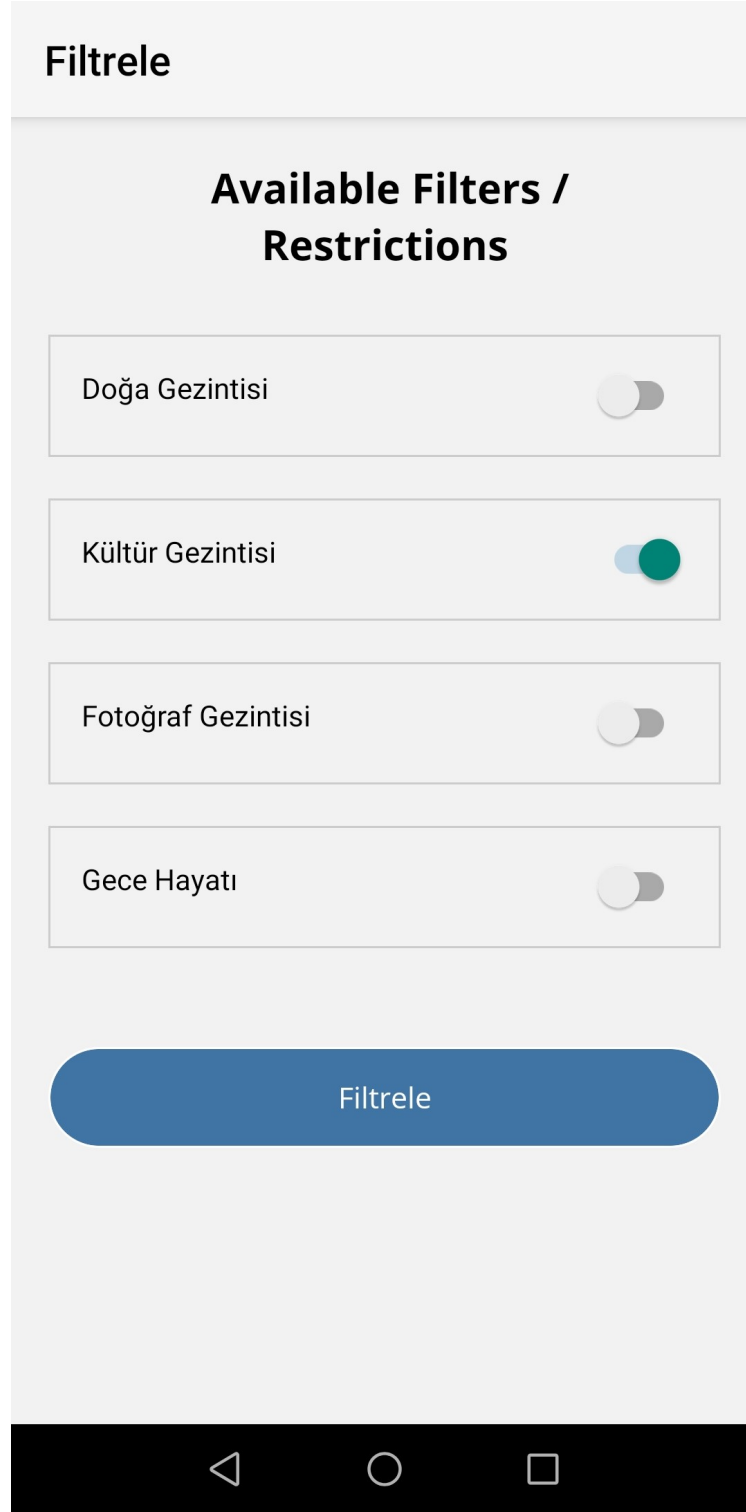
3.2.5 Kategoriye Göre Turlar Ekranı



3.2.6 Şehre Göre Turlar Ekranı



3.2.7 Filtreleme Ekranı



3.2.8 Tur Detay Ekranı




3.2.9 Tur Detay Ekranı Yapılacaklar ve Bilgiler



3.2.10 Tur Detay Ekranı İletişim Bilgileri ve Kişisel Bilgiler






Yiğit Yavuz

Merhaba! Ben Yiğit. İstanbul'un kültürel noktalarını size gezdirmek için buradayım. Üniversiteyi İstanbul'da okuyorum. Hayatımı İstanbul'da geçirdim.

İstanbul'da doğup hala burada yaşayan biri olarak İstanbul'un kültürel bölgelerine gezmeye olan ilgimden dolayı aşinayım ve bu gezmeleri t...

[Daha Fazla](#)

İletişim Numarası : 0555-666-52-58



Keşfet



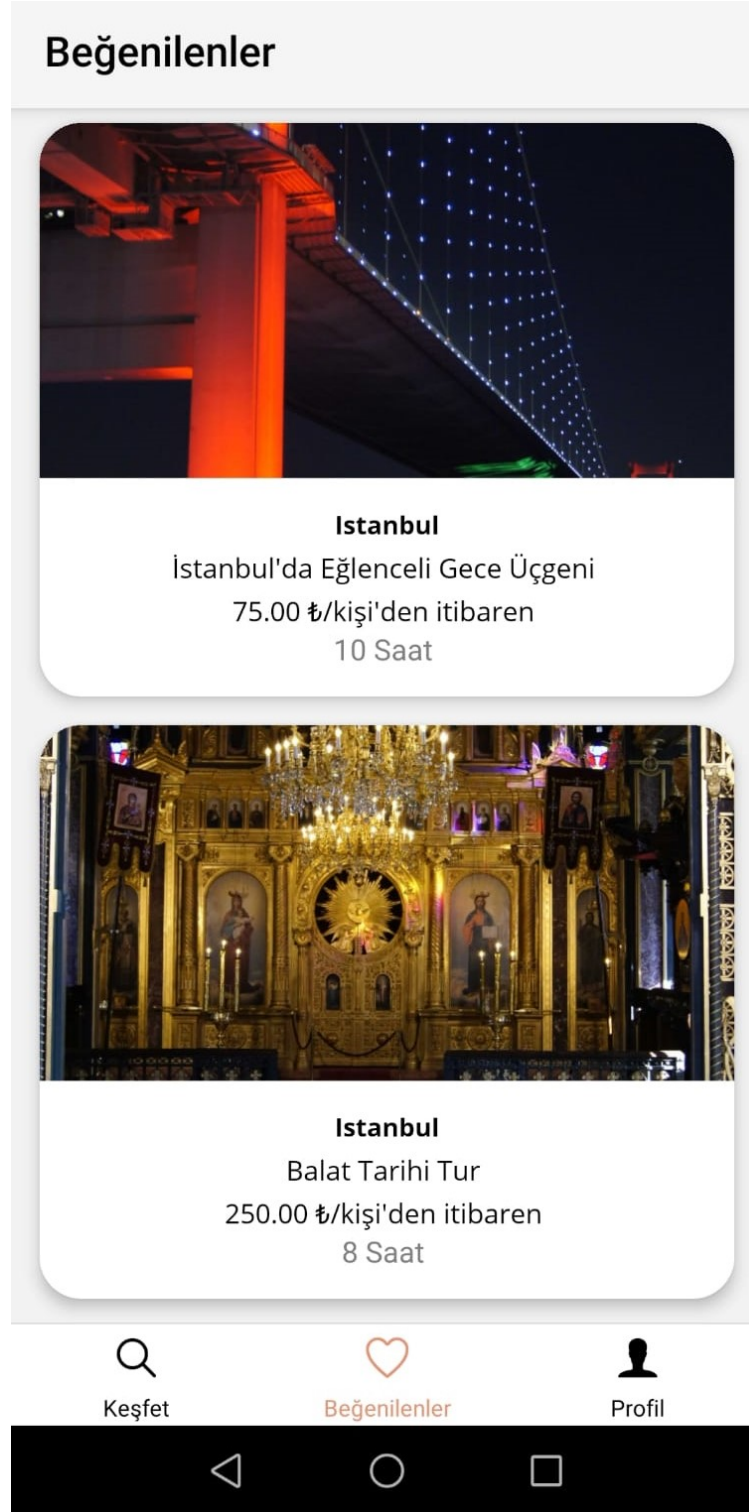
Beğenilenler



Profil

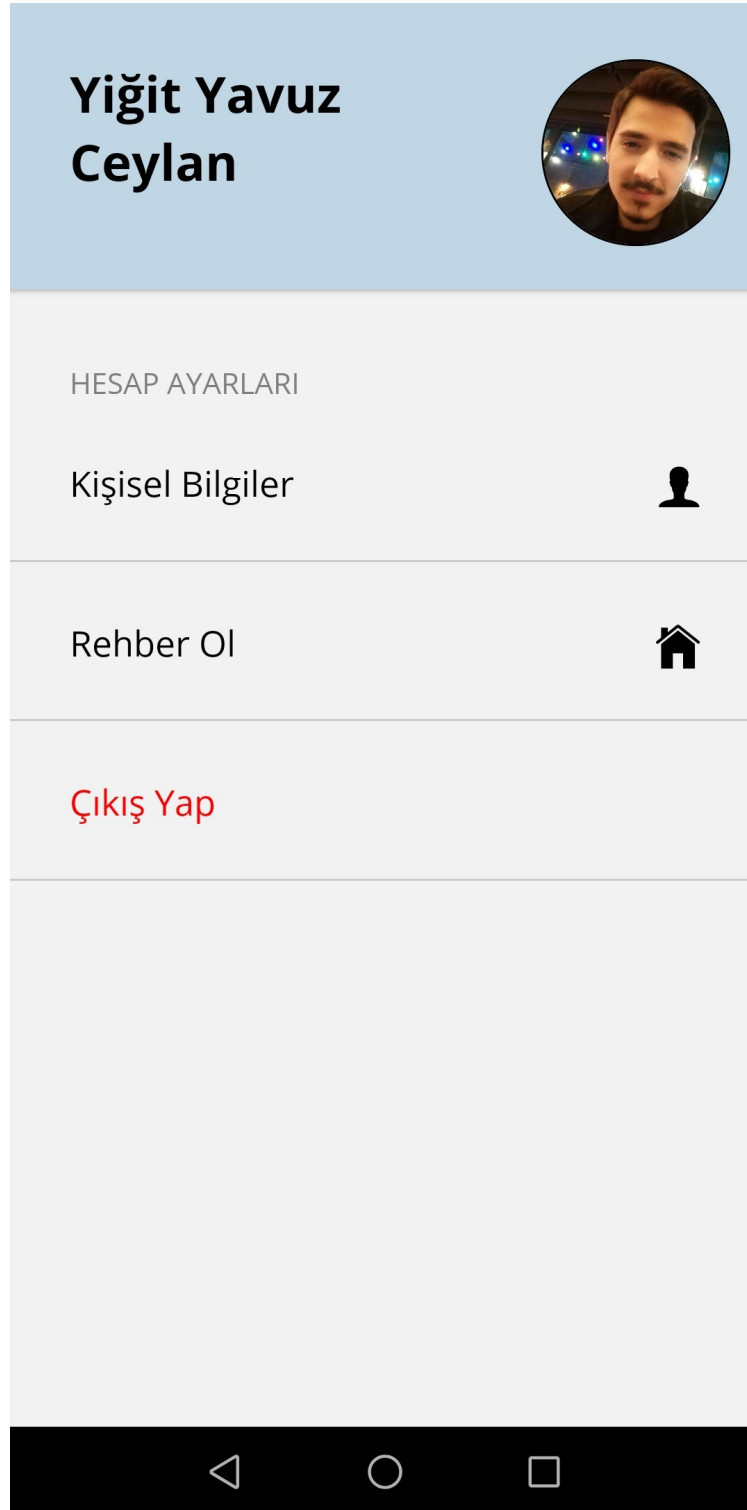


3.2.11 Beğenilen Turlar Ekranı



3.3 Tur Oluřturma ve Profil Sayfası

3.3.1 Ayarlar Ekranı



3.3.2 Kişisel Bilgileri Değiştirme Ekranı

←Kaydet

İsim

Yiğit Yavuz

Soyad

Ceylan

Cinsiyet


Erkek

E-Mail

yigityceylan@gmail.com

- Telefon Numarası

05556665258



◀○◻

3.3.3 Rehber Ol Ara Ekranı



3.3.4 Tur Oluřturma Ekranı 1

← İlerle

İsim
Yiğit Yavuz

Telefon Numarası
05556665258

Kişisel Bilgiler

- Şehir
Eskişehir ▼

3.3.5 Tur Oluřturma Ekranı 2

←İlerle

Şehir ID

Eskişehir

Tur İsmi

Diller

Yapılacaklar

3.3.6 Tur Oluřturma Ekranı 3

← İlerle

- Kategori

Fotoğraf Gezintisi ▼

Saat

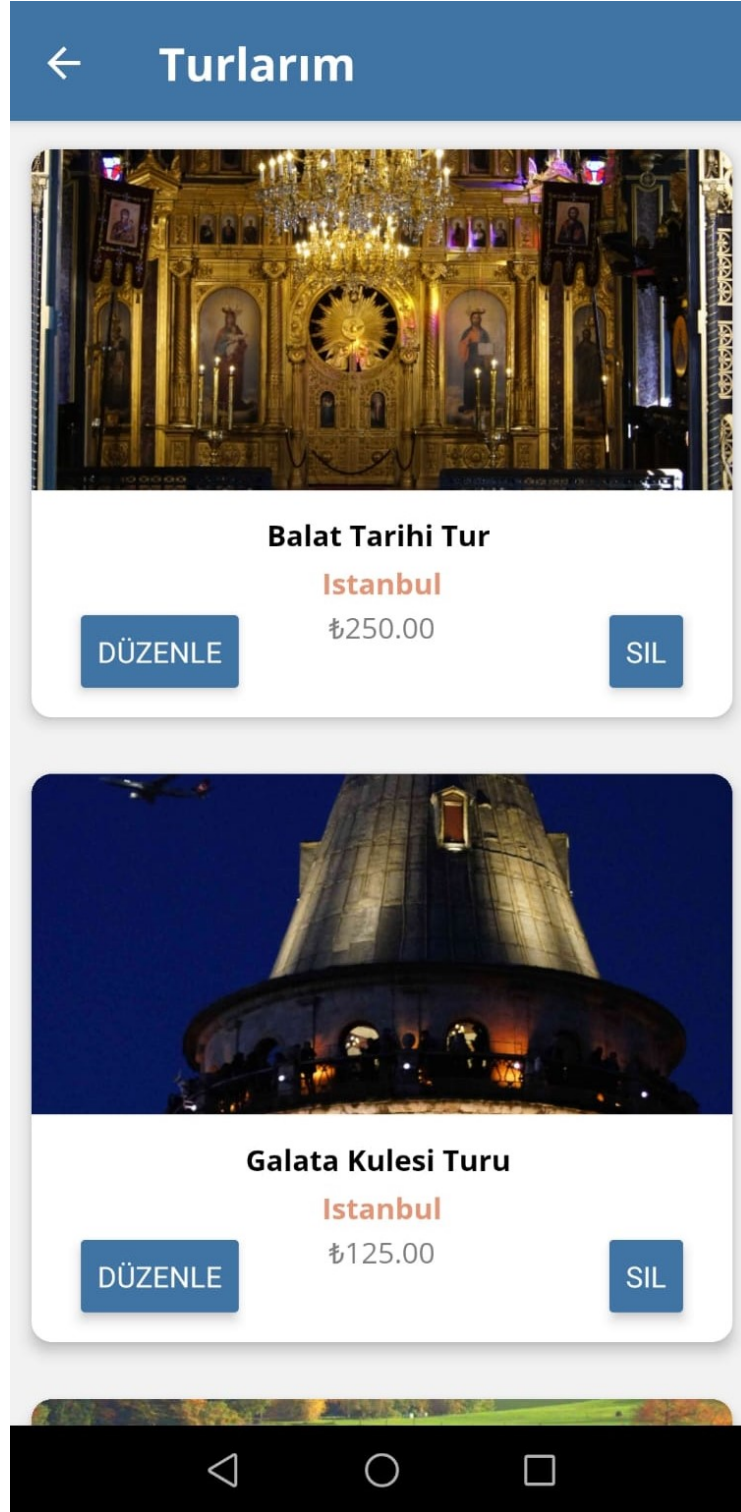
Fiyat

Grup Büyüklüğü

3.3.7 Tur Oluřturma Ekranı 4



3.3.8 Sahip Olunan Turlar Ekranı



3.3.9 Tur Bilgilerini Düzenleme Ekranı

←

Tur Bilgileri

Kaydet

Telefon Numarası

05556665258

Tur İsmi

Balat Tarihi Tur

Saat

8

Fiyat

250

Grup Büyüklüğü

4

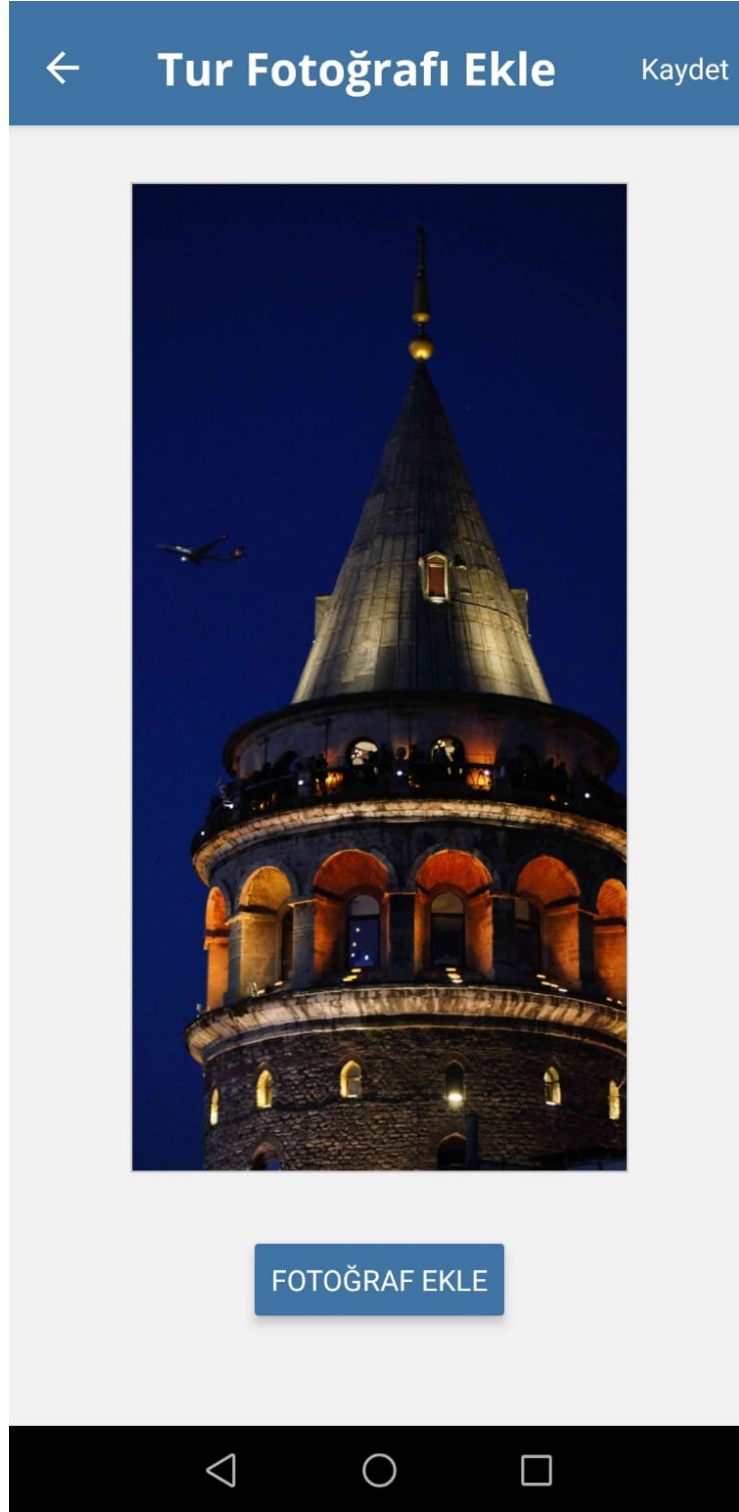
Diller

Türkçe

Kişisel Bilgiler

Merhaba! Ben Yiğit. İstanbul'un kültürel noktalarını size gezdirmek için buradayım. Üniversiteyi İstanbul'da okuyorum. Hayatımı İstanbul'da geçirdim.

3.3.10 Fotoğraf Ekleme Ekranı



Bölüm 4

Özgeçmiş

17 Şubat 1998 tarihli, İstanbul doğumluyum. İlk ve orta eğitimimi İstanbul'un Fatih ilçesinde tamamladım. 2016 yılında Doğa Koleji Anadolu Lisesinden mezun oldum. Bu süre zarfında müzik ile uğraştım. Şimdi mobil ve web geliştiriciliği yönünde ilerliyorum. Bu yönde ilerlerken en son çıkmış teknolojileri kullanarak güncel kalmaya çalışıp teknolojinin hızına yetişmeye çalışıyorum.

Yiğit Ceylan

Kaynakça

- [1] <https://docs.expo.io/>
- [2] <https://firebase.google.com/docs>
- [3] <https://tr.reactjs.org/docs/getting-started.html>
- [4] <https://reactnavigation.org/docs/getting-started>
- [5] <https://www.udemy.com/course/the-complete-web-development-bootcamp/>
- [6] <https://www.udemy.com/course/react-native-the-practical-guide/>
- [7] <https://stackoverflow.com/>