İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

MEKANLARIN KONUMSAL VERİ İLE HARİTA ÜZERİNDE VE ARTIRILMIŞ GERÇEKLİK KULLANILARAK ÜÇ BOYUTTA GÖSTERİLMESİ UYGULAMASI ÖRNEĞİ TOURISTANBUL

YÜKSEK LİSANS TEZİ

Ersel BORA

Bilişim Uygulamaları Anabilim Dalı

Coğrafi Bilgi Teknolojileri Programı

Tez Danışmanı: Prof. Dr. H. Hakan DENLİ

ARALIK 2017

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

MEKANLARIN KONUMSAL VERİ İLE HARİTA ÜZERİNDE VE ARTIRILMIŞ GERÇEKLİK KULLANILARAK ÜÇ BOYUTTA GÖSTERİLMESİ UYGULAMASI ÖRNEĞİ TOURISTANBUL

YÜKSEK LİSANS TEZİ

Ersel BORA
706141008

Bilişim Uygulamaları Anabilim Dalı

Coğrafi Bilgi Teknolojileri Programı

Tez Danışmanı: Prof. Dr. H. Hakan DENLİ

ARALIK 2017

İTÜ, Bilişim Enstitüsü'nün 706141008 numaralı Yüksek Lisans Öğrencisi Ersel BORA, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "MEKANLARIN KONUMSAL VERİ İLE HARİTA ÜZERİNDE VE ARTIRILMIŞ GERÇEKLİK KULLANILARAK ÜÇ BOYUTTA GÖSTERİLMESİ ÖRNEĞİ TOURISTANBUL" başlıklı tezini aşağıda imzaları olan jüri önünde basarı ile sunmustur.

Tez Danışmanı : **Prof. Dr. H. Hakan DENLİ** .....
İstanbul Teknik Üniversitesi

Jüri Üyeleri : **Doç. Dr. Himmet KARAMAN** .....
İstanbul Teknik Üniversitesi

Doç. Dr. Taner Üstüntaş .....
Kocaeli Üniversitesi

Teslim Tarihi : 15 Kasım 2017
Savunma Tarihi : 12 Aralık 2017

Aileme ve arkadaşlarıma,

ÖNSÖZ

Yaşamım boyunca desteklerini esirgemeyen aileme, kendisini tanımaktan ve hocam olmasından büyük onur duyduğum, projenin hayata geçirilebilirliği konusunda bana her zaman destek olan, danışmanım Prof. Dr. H. Hakan DENLİ'ye, mobil uygulamanın grafik tasarımında yardımcı olan ve fikirleriyle güzel bir arayüz çıkmasını sağlayan Nazlı Deniz FETULLAHOĞLU'na teşekkür ederim

Kasım 2017

ErselBORA
Bilgisayar Mühendisi

İÇİNDEKİLER

Sayfa

KISALTMALAR

REST : Temsili Durum Aktarımı (Representational State Transfer)
URI : Tekdüzen Kaynak Tanımlayıcı (Uniform Resource Identifier)
HTTP : Hiper-Metin Transfer Protokolü (Hyper-Text Transfer Protocol)
MVC : Model View Controller Yazılım Mimari Deseni (Model View Controller)
AR : Artırılmış Gerçeklik (Augmented Reality)
JVM : Java Sanal Makinası (Java Virtual Machine)
ER : Varlık/İlişki Modeli (Entity/Relation Model)
SQL : Yapılandırılmış Sorgu Dili (Structured Query Language)
SOAP : Basit Nesne Erişim Protokolü (Simple Object Access Protocol)
WSDL : Web Servis Tanımlama Dili (Web Services Description Language)
JSON : JavaScript Nesne Notasyonu (JavaScript Object Notation)
XML : Genişletilebilir İşaretleme Dili (eXtensible Markup Language)
GIS : Coğrafi Bilgi Sistemleri (Geographic Information System)
GPS : Küresel Konumlama Sistemi
IDE : Tümleşik Geliştirme Ortamı (Integrated Development Environment)
SHA1 : Güvenli "Hash" Algoritması
OSM : Open Street Map
SDLC : Yazılım Geliştirme Yaşam Döngüsü (Software Development Life Cycle)
IOS : Apple cihazlarda kullanılan işletim sistemi

SEMBOLLER

xtile : Sabit bir enlem boyunca değişen boylam değeri
ytile : Sabit bir boylam boyunca değişen enlem değeri

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

MEKANLARIN KONUMSAL VERİ İLE HARİTA ÜZERİNDE VE ARTIRILMIŞ GERÇEKLİK KULLANILARAK ÜÇ BOYUTTA GÖSTERİLMESİ UYGULAMASI ÖRNEĞİ TOURISTANBUL

ÖZET

Ulaşım maliyetlerinin düşmesi ve yaygınlaşması ile beraber şehirler, ülkeler veya kıtalararası yolculuk yapmak günümüzde hem kolaylaşmış hem de ucuzlamıştır. Günümüzde hafta sonlarında, tatillerde uygun bir bütçeyle yurt içine veya yurt dışına kolaylıkla seyahat edilebilir. Seyahatlerimizi planlarken teknolojinin getirdiği kolaylıklardan bolca faydalanılmaktadır. Uçuşlar, kalacak yerler, gezi planları ya internet üzerinden yada bu konularda özelleşmiş akıllı cihaz uygulamalarıyla ayarlanır. Bu işlemler sırasında teknolojiyi çok verimli kullanmamıza rağmen gezilerimiz esnasında çoğunlukla turist haritalarını kullanırız. Halbuki, seyahat edilen şehrin sokaklarında göze hoş gelen yapılar ve eserler hakkında bilgi edinmek, eğer karşılaştılan yapı veya eser herkesçe bilinir değilse, hiç kolay olmamaktadır. Bu probleme ilaveten yurtdışında veri iletişimi ücretlendirmelerinin yüksek olması sebebiyle çoğu zaman gördüğümüz yapı veya eserin bilgisine ulaşmak zorlaşmaktadır. Bu senaryoyu İstanbul'a gelen yabancı veya şehri bilmeyen yerli bir turist için de aynı şekilde düşünebiliriz.

Bu çalışma sonucunda ortaya çıkan proje sayesinde İstanbul'a gelen turistler, bulundukları lokasyona göre çevrelerinde bulunan önemli binaların ve eserlerin konumlarına, bu binalar veya eserler hakkındaki özet bilgilere ulaşabileceklerdir. Kullanıcı uygulamaya üyelik prosedürleriyle uğraşmadan sosyal medya hesaplarından biriyle giriş yapabilmektedir. Giriş yaptıktan sonra kullanıcı ilk olarak iki boyutlu harita görüntüsü sekmesiyle karşılaşır. Haritada lokasyonlar işaretlenmiştir ve kullanıcı bu işaretlerin üzerine tıkladığında ilgili mekanın detay bilgileri gelecektir. Kullanıcı, isterse artırılmış gerçeklik görünüm moduna geçiş yapabilir. Üç yüz altmış derece görünüm sekmesinde kamera açılır ve haritada gelen lokasyonlar bu sefer artırılmış gerçeklik kullanılarak kullanıcıya sunulur. Bu özelliklere ilaveten kullanıcılar internet erişimlerinin kısıtlı olduğu durumlarda veya internet kotalarından kullanmamak için uygulamayı çevrimdışı modda kullanabileceklerdir.Çevrimdışı modu kullanmak için belirli bir alan içindeki harita altlıkları ve o alanda yeralan mekanlar mobil cihazın sabit diskine kaydedilir. Kaydedilen harita altlıkları ve mekanlar internet erişimi olmadan uygulama tarafından gösterilir. Uygulama Android işletim sistemine sahip cihazlar için tasarlanmış olup, akıllı telefonlar ve tabletler tarafından kullanılır.

Lokasyonların üzerine tıklandığında ilgili mekanın bilgileri veritabanından sorgulanıp getirilir. Sorgulama işlemi önyüz Android uygulama ile konumsal veritabanı arasında konumlandırılacak bir orta katman uygulama vasıtasıyla yapılır. Bu uygulamanın görevi; Android uygulamanın internet üzerinden yaptığı REST servis çağrılarını almak ve veritbanında uygun bir veri kümesini Android uygulamaya geri göndermektir. Orta katman uygulama MVC yazılım mimari desenine göre geliştirilmiştir. Bu mimari desene göre orta katman uygulama; REST servis çağrılarının karşılandığı modül ile yapılan çağrıların işlendiği ve veritabanından ilgili verinin getirildiği modül olarak iki ana modelden oluşmaktadır. Bu iki ana modüle ek olarak, tüm işlemlerin kayıt altına alındığı ve uygulama güvenliğinin sağlandığı modül ile veritabanı bağlantılarının ve işlemlerinin yapıldığı modülden oluşmaktadır. Orta katman uygulamanın sağlayacağı bilgiler; tarihi binanın veya eserin ismi, yapı yılı, yapı tipi, tarihi hakkında bilgi ve fotoğrafı olacaktır. Bu sayede kullanıcının mekan hakkında bilgi edinmesi sağlanacaktır.

Projenin ilk fazında İstanbul'un büyüklüğü sebebiyle sadece tarihi yarımadayı kapsayacak veriler toplanmıştır. Verilerin tutulması için mekansal bir veritabanı yapısı kullanılmıştır. Veritabanı dizaynı esnasında veri tutarlılığı ve tekilliği konuları üzerinde çalışılmıştır. Veri girişi işlemleri uygulamadan bağımsız olarak bir veri giriş konsolu yardımıyla yapılmaktadır. Konsola, sadece yetkili kişiler tarafından erişim sağlanır.

Sonuç olarak, projenin ana amacı sosyal sorumluluk bilinciyle yabancı turistlere yardımcı olacak bir uygulama ortaya koymak ve İstanbul'un güzelliklerini daha görünür kılmaktır. Konumsal verilerin yönetilmesi ilerleyen dönemlerde İstanbul'da yeni bölgelerin, İstanbul dışında yeni şehirlerin ve Türkiye dışında farklı şehirler eklenebilmesini ve projenin global olarak kullanılabilmesini sağlayacaktır. Google Maps, Yandex Maps gibi harita sağlayıcılara olan bağımlılığın kalkması sayesinde belirli sayıda konumsal sorgu sonrası ücret ödeme zorunluluğunu ve belirli bir platforma bağlı kalmayı ortadan kaldırmıştır. Projenin ilerleyen fazlarında Emniyet Genel Müdürlüğü'yle veya yaygın olarak kullanılan otel ve turizm uygulamalarının sahibi şirketlerle ortaklaşa çalışılarak alınan geri bildirimlerden oluşturulacak bir suç haritası ile turistler için güvenli yolları belirleyen ve buna göre rota oluşturan bir modül üzerinde durulmaktadır. Proje konusu belirlenip uygulama üzerinde çalışılmaya başlandığında mevcut açık kaynaklı AR modüllerinin dökümantasyon yeterliliği ve kullanım kolaylıkları istenen seviyede değildi. Buna ilaveten belirtilen AR modülleri lokasyon bazlı çalışmamaktaydı. Geçen süre sonunda tekrar yapılan incelemelerde açık kaynaklı AR modüllerinin dökümantasyonunun iyileştirildiği ve kullanımının kolaylaştırıldığı görülmüştür. İlerleyen fazlarda mobil uygulamanın AR modülünün değiştirilmesi düşünülmektedir.

PRESENTING SPECIAL PLACES THROUGH SPATIAL DATA ON MAPS AND THREE DIMENSIONAL SPACE USING AUGMENTED REALITY APPLICATION SAMPLE TOURISTANBUL

SUMMARY

With the decreasing costs and spreading the ways of transportation, traveling between cities,countries or continents is become easy and cheap. Nowadays, it is possible to travel not only domestic but also abroad at weekends or vacations with a convenient budget. When planing travels, technology is used widely. Book flights, otels and trip plans is done via internet or specialized applications on travel. On the other hand, during sightseeing tours, tourist maps are used. In addition, while walking around streets of a city, It is rough to getting information about a pleasing building, historical place or just a place, if it not famous. Furthermore, the data roaming costs are so high for using cellular data to debrief about the place, if mobile data usage is out of the country. It could be thought likewise the use case is same for tourists who come Istanbul from abroad or another cities of Turkey.

The project provides better visuals to the users, using the augmented that has become popular recently. Augmented reality can be explained as computer graphics enrichment of the properties of object in the physical world directly or indirectly simultaneously with the help of a camera . The principle of augmented reality is simply to add new features graphically to the screen of an object that the camera recognizes. However, as the field of work evolves, it has begun to be used independently of the increased reality objects. As an example, a graph can be displayed on the screen when a specific location is reached.

As we examine the mobile application market, the examples often use the map infrastructure of major search engines, although there are examples of augmented reality.In addition, there is a problem that applications are presented as packages, there is a charge after a certain number of spatial inquiries. One of the goals of the project is to create a data model of location and detail information which works on the historical peninsula and to keep data in the database. With the creation of this database, which can be attached to various parts of Istanbul may be established later a working structure in the same way to other towns and cities located in Turkey and in other countries.

For the first phase of the project, due to the size of İstanbul, data has been collected just for the historical peninsula of İstanbul and around that region. Keeping spatial data is one the tough issue on such projects. Therefore, data has been kept in spatial database. While designing the spatial database, it is paid attention to the subject of data singularity and data consistency. Data model is a basic building block that needs to be studied on the first stage as an application that provides mobile application or data service. Therefore, before starting to develop the system, the type of the data to be used, how to sort it, how to relate to the object at the application layer should be considered in detail and the design of the system should be based on this data model. The resulting data diversity and data quality affect data modeling. In this sense,

sample data should be collected before a system is developed. During data modeling, the entity/relationship model structure is used. The entity/relationship model is a diagram that describes the properties of assets and how they relate to other assets.

After the data model is completed, the process of implementing model in the database system is started. Objects are kept as tables in the database system, while properties of objects are kept as columns in the tables. Querying the data from database system and performing various analyzes by processing are done by using the structured query language of the database system. Structured Query Language allows to add records to the database system, delete and update operations, as well as to create entity tables and manage properties of these tables. Database performance has a very important role today. Fast and resilient database; the performance of the application that uses the data and the greater number of end users. Due to this reason, the investments made in the databases are a huge commodity. In addition to this, instead of being held on servers by the classical method, the concept of keeping this service in cloud systems is becoming widespread today

Touristanbul application provides a rich content of special places, which are brief information about places, photographs, locations of places, in the area of historical peninsula of Istanbul for the tourists. For offering easy use, it could be logged in to the application via social media accounts for instance; Google or Facebook. After user login the application, user encounter with a two dimensional map scene page. In the page, the locations of places are marked. In addition, when it is pressed on a location marker the summary information of places is demonstrated. In addition, it could be switched to three dimensional screen page. With 3D screen, the camera of mobile device is become open and the location markers are showed in the screen via using augmented reality. The application is designed for smart devices for istance; smart phones and tablets that use Android operating system

When pressing a marker in 2D and 3D screens show information about the place is shown which is queried from database. For querying information between database and Android application is done within a middleware application. The tasks of middleware application are handling REST calls from Android application, parsing message and querying database for returning a proper dataset. The middleware application is developed by using Model-View-Controller design pattern. In addition, Spring framework is used for practicing a multilayer server-side application. Middleware application is consist of two main module. The controller module that communicates Android application via RESTful services, the model module that controls business flow and data gathering. Also, middleware application has a logging module, database access module that uses Hibernate and security module which is used to make authentication and secure application against attacks. Middleware application provides information which it is consist of construction date, name of place, brief information and photograph.

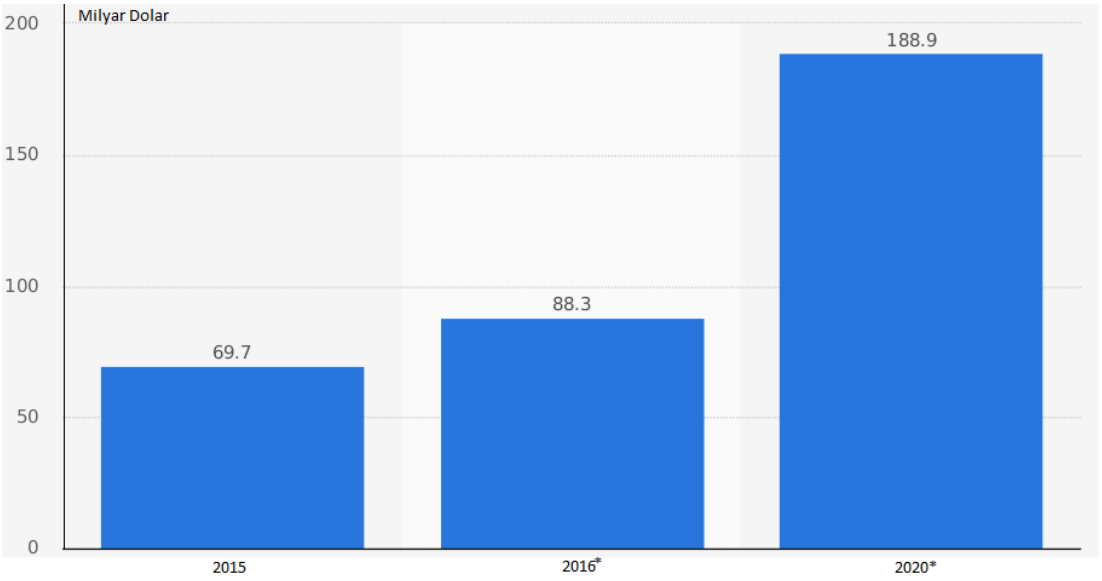
As a result, the main goal of the project is helping tourists with an easy to use application and of course, making the beautiful places of İstanbul visible within social responsibility. The management of spatial data will enable; adding new regions in Istanbul, new cities outside of Istanbul and different cities outside of Turkey. Therefore the project could be used globally. Thanks to getting out of dependency on map providers for instance; Google Maps and Yandex Maps, the need to pay after a certain number of positional queries and the to being stuck to a specific platform have been removed. In the later phases of the project, a criminal map will be formed from the General Directorate of Security or the feedback received from working with commonly used hotel and tourism applications in cooperation with the proprietors, and a module will be designed that brings out safe areas for tourists and determines a route accordingly.When the project theme was identified and studied on the application; documentation sufficiency and ease of existing open source AR modules were not at desired level. In addition the specified AR modules did not work as location based. In the re-examinations at the end of the period, it was seen that the documentation of the open-source AR modules was improved and the usage was facilitated. In the following phases, it is considered to replace the AR module of the mobile application.

1. GİRİŞ

Günümüzde hayatımızı düzenlemek işlerimizi kolaylaştırmak için çoğunlukla bilgisayarlar kullanılmaktadır. Bilinen anlamda ilk bilgisayar 19. yüzyılda Charles Babbage tarafından geliştirilen analitik hesap makinasıdır. 1946 yılında genel amaçlı olarak ENIAC (Electronic Numerical Integrator And Computer) geliştirilmiştir. ENIAC'ın ağırlığı 30 tondu ve 18 bin vakum tüpünden oluşmaktaydı. 1951 yılında ticari amaçlı olarak UNIVAC 1 (Universal Automatic Computer) geliştirildikten sonra, 1953'de IBM 650 ve 700 serileri kullanılmaya başlanmıştır [1]. 1963'ten sonra bilgisayarların boyutları, gelişen transistor teknolojisiyle beraber küçülmüştür. Şekil 1.1'de Münih Alman Müzesi'nde bulunan IBM 2203 gösterilmiştir. Teknolojinin ilerlemesi ve üretim maliyetlerinin düşmesi bilgisayarların kabiliyetlerini de geliştirmiş ve görüntü işleme, dosya transferleri, transferi, internetin internet kullanımı gibi işlemleri kolaylaştırmıştır. Teknik olarak bakıldığında kullandığımız mobil cihazlar ve akıllı telefonlar da birer bilgisayardır. 2016 yılı itibarıyla dünya genelinde 4.3 milyar adet akıllı telefon satılmıştır [2]. Bu istatistikten yapılabilecek çıkarım, insanların artık kişisel ve orta ölçekli bilgisayarlar yerine küçük ve mobil cihazları tercih ettikleridir.

Şekil 1.1 : IBM 2203 Bilgisayarı.

Akıllı telefon ve cihaz pazarının bu kadar gelişmesi, akıllı cihazlar için uygulama geliştirilmesinin etkileyici bir şekilde hızlanmasını sağlamıştır. Pazarının büyümesi bu cihazlar için tasarlanan uygulamalardan elde edilen kazançları da büyük boyutlara taşımıştır. Şekil 1.2'deki grafikte, mobil uygulamalardan elde edilen elde edilen ve 2020 yılında beklenen toplam gelir gösterilmiştir [3].



Şekil 1.2 : Mobil uygulama pazarından elde edilen gelir.

Uygulama çeşitliliğine baktığımızda genellikle oyun, iş, sağlıklı yaşam, sosyal medya, eğlence ve eğitim alanlarının baskın olduğu ve bu çeşitlilikte, seyahatle ilgili uygulamaların toplamda %3.93'lük bir payının olduğu görülmektedir [4]. Ayrıca turist rehberliği konusunda çok çalışma yapılmadığı, kullanıcıların çevrimiçi çalışan mevcut harita, otel ve uçak rezervasyonu uygulamalarını kullandıkları saptanmıştır [5].

Seyahat edebilmenin kolaylaşmasıyla, ülkemize gelen yabancı turist veya yurt içi yerli turist sayısında artış ve çeşitlilik görülmektedir. Ülkemizde turistleri cezbeden yerlerin büyük çoğunluğu kuşkusuz İstanbul'dadır. İstanbul'a gelen yerli ve yabancı turistler tarihi ve turistik mekanları bulmak için geleneksel yöntem kullanıp turist haritalarından yararlanmaktadırlar. Buna ek olarak akıllı cihazlarındaki harita uygulamalarıyla gitmek istedikleri yeri kolayca bulabilmektediler. Ancak, özellikle yabancı turistler için, dolaşırken göze hoş gelen yapılar ve eserler hakkında bilgi edinmek, eğer ilgili yapı rehber haritalarında yer almıyorsa zordur.

Bu probleme ilaveten yurtdışında bulunma durumunda akıllı cihazlardan ücretlendirmeler sebebiyle çoğu zaman veri iletişimi yapılamamakta gördüğümüz yapı veya eserin bilgisine ulaşmak zorlaşmaktadır.

Bu kapsamda öncelikle kullanıcıların bir harita üzerinden veya kamera vasıtası ile artırılmış gerçeklik kullanılarak gördükleri eser ve yapı hakkında bilgi edinebilmesi amaçlanmaktadır. Projenin esin kaynağı seyahatler esnasında aynı problemin yaşanması ve Türkiye'de söz konusu çalışmaya benzer bir proje yapılmamış olmasıdır. Genel olarak bakıldığında bu tarz uygulamaların var olmasına rağmen, bu uygulamalarda genellikle hazır veritabanları kullanmıştır. Hazırlanan altyapı esnek olup, uygulamanın ilerleyen fazlarda farklı şehirlerde ve ülkelerde kullanılması düşünülmektedir.

1.1 Tezin Amacı

Projenin öncelikli amacı öncelikle İstanbul'da tarihi yarımadayı kapsayacak şekilde turistlere rehberlik edebilmek, çevrelerinde gördükleri eserler hakkında kolaylıkla bilgi verebilmektir. Uygulama, Android işletim sistemine sahip cihazlara yüklenebilecek ve kullanıcının sosyal medya hesaplarından biriyle giriş yapılıp uygulamanın basit ve hızlı bir şekilde kullanması sağlanacaktır.

Projede son zamanlarda popülerleşen artırılmış gerçeklik kullanılarak, kullanıcılara daha iyi görsellik sunulmuştur. Artırılmş gerçeklik, bir kamera yardımıyla eş zamanlı olarak doğrudan veya dolaylı bir şekilde fiziksel dünyada bulunan nesnenin özelliklerinin bilgisayar grafikleriyle zenginleştirilmesi olarak açıklanabilir [6]. Artırılmış gerçekliğin çalışma prensibi basitçe kameranın tanıdığı bir nesneye ekranda grafik olarak yeni özellikler eklenmesidir. Ancak çalışma alanının gelişmesiyle beraber artırılmış gerçeklik nesnelerden bağımsız olarak da kullanılmaya başlanmıştır. Örnek olarak belirli bir lokasyona gelindiğinde ekranda bir grafiğin gösterilmesi verilebilir.

Mobil uygulama pazarını incelediğimizde, dünyada artırılmış gerçeklik ve mekansal veriyi harmanlayan örnekler bulunmasına rağmen, incelenen örnekler genellikle büyük arama motorlarının harita altyapılarını kullanmaktadırlar. Bu durum uygulamanın özelleştirilmesini zorlaştırmaktadır. Buna ilaveten, paket olarak sunulan çözümlerde belirli bir mekansal sorgulama sayısından sonra ücret talep edilmesi sorunu vardır. Projenin hedeflerinden bir tanesi de, tarihi yarımadada bulunan yapı ve eserlerin lokasyon ve detay bilgilerinin bir veri modeli oluşturulup veritabanında tutulmasıdır. Bu veritabanının oluşturulmasıyla beraber, İstanbul'un çeşitli bölgeleri altyapıya eklenebilecek, daha sonra Türkiye'deki diğer şehirler ve son olarak başka ülkelerde bulunan şehirler için de aynı şekilde çalışan bir yapı kurulabilecektir.

1.2 Kullanılan Kavramlar ve Denklemler

Bu bölümde, tezin kapsamında kullanılan kavramlar ve denklemler hakkında açıklamalar yapılacak ve tezin altyapısının daha anlaşılır olması sağlanacaktır.

1.2.1 Artırılmış gerçeklik

Artırılmş gerçeklik, bir kamera yardımıyla eş zamanlı olarak doğrudan veya dolaylı bir şekilde fiziksel dünyada bulunan nesnenin özelliklerinin bilgisayar grafikleriyle zenginleştirilmesi olarak açıklanabilir [6]. Artırılmış gerçekliğin amacı kullanıcıya daha zengin bir grafik deneyim sunmaktır Artırılmış gerçekliğin iki çeşidi bulunmaktadır. Bunlar; işaretleyici tabanlı ve işaretleyici tabanlı olmayan artırılmış gerçekliktir [7]. İşaretleyici tabanlı artırılmış gerçeklikte, uygulamanın kullandığı algoritma kamera yardımıyla bir işareti, şekli veya çizimi tanıdığında kullanıcıya ekstradan grafikleri gösterir. İşaretleyici tabanlı olmayan uygulamalar genellikle lokasyon verisini kullanır. Belirlenen lokasyona gelindiğinde, uygulama lokasyonu algılayıp kullanıcıya uygun grafikleri gösterir.

1.2.2 Mobil uygulama

Donanım ve çip üretimi alanlarındaki ilerlemeler sonucunda ilk etapta son kullanıcılar için tasarlanan bilgisayarlar küçülmeye başladı. Bu küçülmenin devamında Apple şirketinin öncülüğünde günlük hayatta kullandığımız telefonlara bir çok kabiliyet eklendi ve mobil akıllı cihazların doğuşu gerçekleşti. Esasen kullandığımız her akıllı cihaz ölçek olarak küçültülmüş birer bilgisayardır.

Kullandığımız akıllı cihazlar için geliştirilen, son kullanıcıya çeşitli alanlarda yardımcı olmayı amaçlayan uygulamalara mobil uygulama denilmektedir. Her mobil uygulama cihazın konumlandırıldığı işletim sistemine uygun bir şekilde kullanılan kodlama diliyle geliştirilmektedir. Mobil uygulamalar için kullanılan programlama dilleri Çizelge 1.1'de gösterilmiştir.

Çizelge 1.1: Mobil uygulama geliştirmede kullanılan programlama dilleri ve işletim sistemleri.

Programlama Dili	İşletim Sistemi
Java	Android
Objective-C	IOS
Visual C#,C++	Windows Mobile
Java ME	Blackberry RIM

1.2.3 Java programlama dili

Java programlama dili, 90'lı yıllardan itibaren nesne tabanlı, "bir kere yaz tüm ortamlarda çalıştır" prensibi göz önüne alınarak geliştirilen bir dildir. Java'da gerçek dünyadaki tüm nesneler ve özellikleri, programlama esnasında birer sınıf yapısında temsil edilir. Bu sayede, düşük düzeyli programlama dillerinden farklı olarak nesnelerin özellikleri kolaylıkla işlenebilmektedir [8]. Örnek bir sınıf yapısı Şekil 1.1'de verilmiştir. Java'da programlama yapıldıktan sonra yazılan programın işletim sistemine uygun bir şekilde eklemesini, arka planda JVM sağlar. Java programlama dili günümüzde yoğun bir şekilde çeşitli alanlarda programlama yapılıp sistem geliştirmek için kullanıldığı gibi, Android işletim sistemi tabanlı cihazlara uygulama geliştirmek için de kullanılır. Bu sebepten Android bir uygulama yapabilmek için Java öğrenilmesi gereklidir. Java programlama dilini kullanıp bilgisayar programları yazabilmek için kod geliştirme ortamına ihtiyaç duyulmaktadır. Java dilinde programlama yapabilmek için genellikle Eclipse, NetBeans, IntelliJ geliştirme ortamları kullanılır. Bu çalışmanın Java kısımları için Eclipse geliştirme ortamı kullanılmıştır.

1.2.4 Android yazılım geliştirme

Android yazılım geliştirme, Android işletim sistemine sahip cihazlar için genellikle Java programlama dili kullanılarak yeni uygulamalar hazırlanmasıdır. Java programlama dilini bilen yazılım geliştirici bile ilk etapta Android uygulamaları geliştirmekte zorlanabilir. Bunun sebebi, Android uygulamaların ön yüz tasarımlarının kullanıcı deneyimi bakımında daha detaylı olmasından kaynaklanır Kullanılan dil Java olmasına rağmen, Android ve Java uygulamalar konsept olarak tamamen ayrışırlar. Kullanılan konseptlerin oturması ve düzgün bir uygulama ortaya konması ilk etapta uzun zaman alabilmektedir. Esasen Android sabır isteyen kültürdür [9].

Android uygulamaları genel olarak Android Studio ve IntelliJ geliştirme ortamlarında kodlanır ve test edilebilir. Java uygulamalardan farklı olarak Android uygulamalar genellikle bir android işletim sistemine sahip cihaz üzerinde test edilebilir. Bu şekilde test etmenin iki faydası vardır. Öncelikle, geliştirme ortamlarının sunduğu sanal makinaların çalışma hızı çok yavaştır ve ilk açılışları zaman alır. Testler gerçek bir Android işletim sistemine sahip cihaz üzerinde yapılırsa zaman kaybının önüne geçilmiş olur. İkinci olarak, gerçek bir cihazda yapılan test cihazın tepkisinin ölçülmesi, uygulamanın gereksinimleri, enerji kullanımının anlaşılması açısından daha etkilidir.

1.2.5 Veri modelleme ve veritabanı kavramı

Fiziksel dünyada yer alan nesnelerin, bilgisayar programlarında nesneye dayalı programlama metodolojisi ile temsil edildiğine; Java programlama dili bölümünde değinilmiştir. Program boyutunda nesneler hakkında edinilen bilgilerin belirli bir şekilde tutulmasını ve gelecekte bu bilgilere tekrardan erişilmesini sağlayan büyük boyutlu özelleşmiş veri saklayan sistemlere veritabanı; bu verinin nasıl tutulacağını belirleyen modele de veri modeli adı verilmektedir [10].

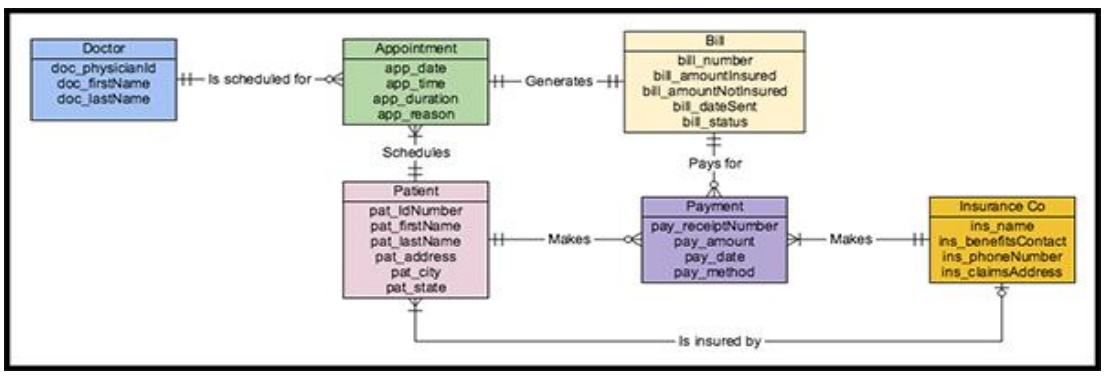
Veri modeli, mobil uygulama veya veri servisi veren bir uygulama olsun ilk etapta üzerinde çalışılması

gerekli olan temel yapıtaşıdır. Bu sebepten, sistem geliştirmeye başlamadan önce kullanılacak verinin özelliklerinin tipi, nasıl sıralanacağı, uygulama katmanında nesnelerle nasıl ilişkili olacağı üzerine detaylı bir şekilde düşünülmeli ve sistemin dizaynı, veri modeli baz alınarak çıkartılmalıdır. Elde edilecek veri çeşitliliği ve veri kalitesi veri modellemeyi etkilemektedir. Bu sebepten, bir sistem geliştirilmeye başlanmadan önce örnek veriler toplanmaya çalışılmalıdır. Veri modelleme esnasında Varlık/İlişki model yapısı kullanılır. Varlık/İlişki modeli, varlıkların özelliklerini ve diğer varlıklarla nasıl ilişkili olduğunu belirten bir diagramdır [11]. Bu modelde varlıklar arasındaki ilişkiler Çizelge 1.2'de verilmiştir.

Çizelge 1.2: Varlık/ilişki tipleri.

İlişki Tipi	Açıklaması
Birebir İlişli	İki varlık arasında eşsiz ve tek bir ilişki vardır
Bir ile Çok İlişki	Bir varlık diğer varlıktan birden fazlasıyla ilişkilidir
Çok ile Çok İlişki	Varlıklar birbirleriyle birden fazla olacak şekilde ilişkilidir.

Veri modeli tasarımı genel olarak şu adımlardan oluşur; varlık olabilecek yapıların listelenmesi, varlıkların özelliklerinin belirlenmesi, varlıklar arsındaki ilişkilerin belirlenmesi, basitçe bir Varlık/İlişki diagramının çizilmesi ve son olarak diagramın son haline getirilmesi. Örnek bir Varlık/İlişki modeli Şekil 1.3'de gösterilmiştir. Şekil 1.3'de görüldüğü üzere bir doktorun birden çok randevusu olabilir bu bir ile çok ilişki'ye örnektir [12]. Her randevunun sadece bir tane faturası olur, bu birebir ilişki örneğidir.



Şekil 1.3 : Örnek varlık/ilişki diagramı.

Veri modeli tamamlandıktan sonra tasarlanan modelin veritabanı sisteminde gerçeklenmesi işlemine geçilir. Varlıklar veritabanı sisteminde tablolar olarak, varlıkların özellikleri ise tablolardaki kolonlar olarak tutulur. Veritabanı sisteminde verinin sorgulanması ve işlenerek çeşitli analizlerin yapılması işlemi veritabanı sisteminin Yapılandırılmış Sorgu Diliyle yapılmaktadır. Yapılandırılmış Sorgu Dili sayesinde veritabanı sistemine kayıt ekleme, silme güncelleme işlemleri yapılabildiği gibi, varlık tablolarının oluşturulması ve bu tabloların özelliklerinin yönetilmesini sağlamaktadır.

Veritabanı performansı günümüzde çok önemli bir yere sahiptir. Hızlı ve dirençli bir şekilde çalışan veritabanı; veriyi kullanan uygulamanın performanslı çalışmasını ve daha fazla sayıda son kullanıcıya hizmet verilmesini sağlamaktadır. Bu sebepten dolayı veritabanlarına yapılan yatırımlar büyük paralara mal olmaktadır. Buna ilaveten veritabanlarının; klasik yöntemle sunucularda tutulması yerine, bu hizmeti veren bulut sistemlerinde tutulması konsepti günümüzde yaygınlaşmaktadır. Çizelge 1.3'de günümüzde yaygın olarak kullanılan veritabanı sistemi çeşitleri ve sorgulama dilleri verilmiştir.

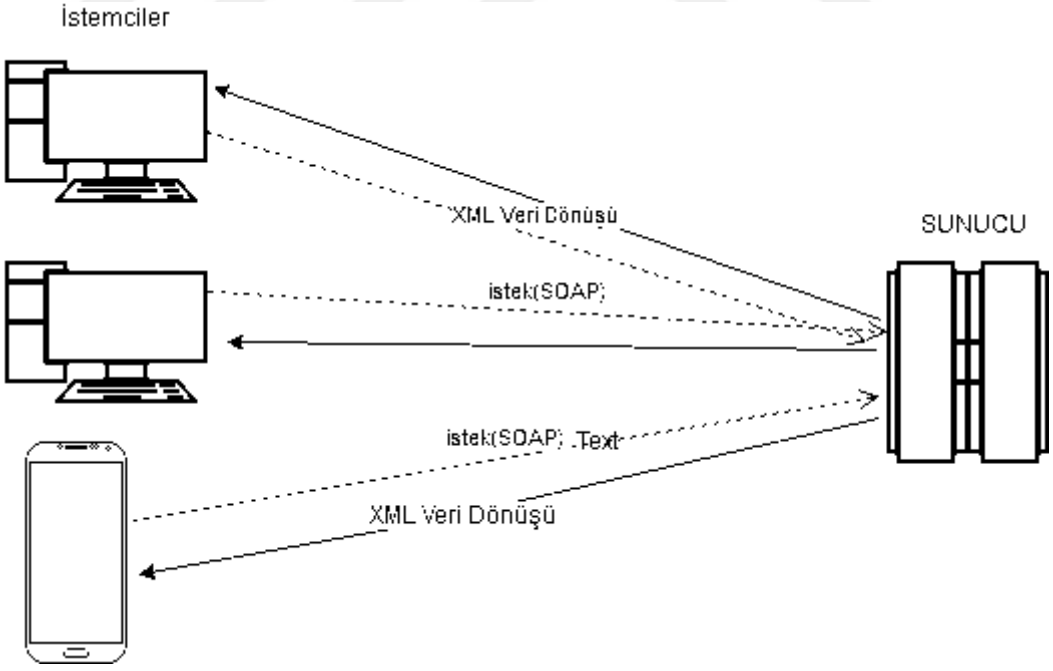
Çizelge 1.3 : Yaygın veritabanı sistemleri ve sorgulama dilleri.

Veritabanı Sistemi	Yapılandırılmış Sorgu Dili
Oracle	PL-SQL
Microsoft Sql Server	T-SQL
MySQL	Standard SQL
PostgreSQL	PostgreSQL
IBM DB2	Standard SQL

1.2.6 Web servis

Web servis, HTTP protokolüyle sistemlerin kendi aralarında belirlediği bir mesajlaşma yapısı üzerinden veri alıp vermelerini sağlayan programlardır. Web servisler üzerindeki bu veri alışverişi standart XML dökümanlarıyla olmaktadır [13]. XML tabanlı web servislerin tanımlamaları Web Servis Tanımlama Dili (WSDL) kullanılarak yapılır.

Bir WSDL dökümanında genel olarak; web servisin adı, yapılabilecek yordam çağrıları, yordam çağrılarının alabileceği parametreler, parametrelerin tipleri, döndürülecek cevap ve biçimi, hata zamanında döndürülecek mesaj tanımlamaları bulunmaktadır. Bu tanımlamalarla birlikte sunucu ve istemci arasında veri alışverişi belirli bir haberleşme protokolüne göre yapılmaktadır. Web servis mimarsinde kullanılan bu protokole Basit Nesne Erişim Protokolü (SOAP) denilmektedir. Web servise ait haberleşme yapısı Şekil 1.4'de gösterilmiştir.



Şekil 1.4 : Web servis haberleşme yapısı.

SOAP altyapısını kullanan web servislerin XML formatında haberleşmesi ve haberleşme esnasında XML'in dökümanının büyüklüğüne göre servisin geç cevap verebilmesi veya cevap verememesi problemlerini aşabilmek için günümüzde daha

hafif ve kolaylıkla parçalanabilecek bir veri yapısının kullanılması gerekliliği doğmuştur. Kullanılan bu yeni veri yapısına JavaScript Nesne Notasyonu (JSON) adı verilir [14]. JSON veri yapısı insanlar tarafından okunulurluğnun kolaylığı ve istemci tarafında kolaylıkla işlenebilmesi bakımından son yıllarda yaygın bir şekilde kullanılmaktadır. Şekil 1.5'de örnek JSON dökümanı gösterilmiştir. Dökümanda belirlenen bir nesnenin özelliklerinin nasıl verileceği gösterilmiştir. Şekil 1.5'de gösterilen dökümanda roman türündeki kitaplar ürün kataloğunda kitapların kendi öz nitelikleriyle verilmişlerdir.

```
{
  "tip": "kitap",
  "tur": "roman",
  "urunler": [
     {"ad": "Milenaya Mektupelr", "yazar": "Franz Kafka","fiyat":"22.5"},
     {"ad": "Martin Eden", "yazar": "Jack London","fiyat":"36.0"},
     {"ad": "Kar", "yazar": "Orhan Pamuk","fiyat":"28.0"}
     ]
}
```

Şekil 1.5 : Örnek JSON dökümanı.

JSON veri yapısının geliştirilmesiyle beraber web servis çalışma mantığı da değişmiştir. Günümüzde servis mimarisi REST'e evrilmiştir. REST dağıtık sistemler için oluşturulan yazılım mimarisi stilidir. REST'in prensipleri şu şekilde sıralanır; veriler kolay anlaşılabilir bir dizin yapısıyla URI'lere dönüştürülmelidir, veriler JSON veya XML tipinde gösterilebilmelidir, mesajlaşma HTTP protokolüyle sağlanabilmelidir, mesajlaşma oturumsuz olmalıdır [15]. REST mimarisini kullanarak haberleşen web servislere RESTful Web Servis'ler denilmektedir [16]. Genel mantık olarak XML tabanlı web servisler gibi çalışmalarına rağmen bir WSDL tanımlamasına ihtiyaç duymazlar. Bunun yerine tüm veriyi bir JSON nesnesine yükleyip istemciye gönderirler. RESTful servislerin geliştirilmesi ve diğer websitelerle entegrasyonu daha kolaydır. RESTful servislerin tüm bu artı niteliklerine rağmen, SOAP altyapısı daha stabil, notasyonu belirli bir hizmet sunar. Yapılacak olan projeye göre iki servis mimarisinden biri seçilmelidir. Seçim; kullanıcının ihtiyaçları, performans parametreleri ve güvenli erişim konularına göre farklılık gösterebilir.

1.2.7 Uygulama programlama arayüzü

Uygulama programlama arayüzü (API); kendi içinde kullanıma özel veya genel fonksiyonlar bulunduran, bu fonksiyonları kullanarak programlamanın daha hızlı bir şekilde tamamlanmasını sağlayan hazır alt program olarak tanımlayabiliriz [17]. Örnek olarak, bir hesap makinası programı yazmak istediğimiz varsayalım. Basit aritmetik işlemleri kendimiz programlayıp, Trigonometri fonksiyonlarına sahip bir API'yi kendi programımıza ekleyerek bu arayüzün hazır fonksiyonlarını kullanıp hesaplamalarımızı yapabiliriz. Bu sayede arka planda API'nin nasıl çalıştığı, hesaplamaları nasıl yaptığıyla ilgilenmeden sonuç odaklı bir şekilde programımızı tamamlamış oluruz. Günümüzde kullanılan API'lerin sınıflandırılması Çizelge 1.4'de verilmiştir.

Çizelge 1.4 : API sınıflandırılması ve örnekleri.

API	Örnek
Web Servis API	SOAP,REST,JSON-RPC
Kütüphane Tabanlı API	JavaScript,AngularJs,JQuery
Sınıf Bazlı API	Java,Android,IOS
Donanım Tabanlı API	Depolama Birimleri, Grafik Kartlar
Yazılım Tabalı API	GoogleMaps,Wikitude

1.2.8 Yazılım geliştirme yaşam döngüsü

Yazılım geliştirme yaşam döngüsü (SDLC); fikir aşamasından son kullanıcının isteklerinin toplanıp isteklere yönelik bir yazılımın geliştirilmesi için projenin çeşitli safhalara ayrılması ve bu süreçlerin tümü şeklinde açıklanabilir [18]. Geleneksel SDLC genellikle beş fazdan oluşurken günümüzde yedi fazdan oluşan bir SDLC yapısı kullanılmaktadır. Bu adımlar sırasıyla;

- Planlama
- Sistem analizi, ihtiyaçların çıkartılması ve sistem dizaynı,
- Geliştirme entegrasyonu ve sistemin test edilmesi
- Sistemin kurulumunun yapılması

- Bakım ve operasyon desteği

SDLC sayesinde bir projenin zaman planlaması ve projenin belirlenen plana göre devam edip etmediği kolaylıkla yönetilebilir.

1.2.9 Haversine formülü

Haversine formülü, Dünya'nın şeklini küre alarak enlem ve boylam bilgileri verilen iki nokta arasındaki mesafeyi hesaplamamıza yarar [19]. Formülde, Dünya'nın tam küre olduğu varsayımı kullanıldığı için yapılan mesafe ölçümünün %0.5'e kadar yükselebilen bir hata payı bulunmaktadır. Haversine formülü denklem 1.1'de gösterilmiştir. $\boldsymbol{\alpha}$ enlemdeki toplam değişimi $\boldsymbol{b}$ ise boylamdaki toplam değişimi belirtir.

$$d = 2r\sin^{-1}\left(\sqrt{\sin^2\left(\frac{a_2 - a_1}{2}\right) + \cos(a_1) * \cos(a_2) * \sin^2\left(\frac{b_2 - b_1}{2}\right)}\right) \quad (1.1)$$

1.2.10 Vicenty formülü

Vicenty formülü, Dünya'nın şeklini kutuplardan yassılaşmış küremsi olarak ele alır ve enlem boylam bilgileri verilen iki nokta arasındaki mesafeyi hesaplamamıza yarar [20]. Vicenty formülü, direkt ve ters metot olmak üzere iki hesaplama yolundan oluşur. Direkt metot, belirli bir noktadan uzaklık ve yön bilgisi verilerek ikinci noktanın bulunmasını sağlar. Ters metot ise, verilen iki nokta arasındak coğrafi uzaklığı ve yönün bulunmasını sağlar. Vicenty formülü hata payı beş milimetreye düşene kadar iteratif olarak devam eder. Denklem 1.2'de ters metot ve denklem 1.3'de direkt metot gösterilmiştir.

$$\text{Sin } \sigma = \sqrt{[\ (\cos U_2 * \sin\lambda)^2 + (\cos U_1 * \sin U_2 - \sin U_1 * \cos U_2 * \cos\lambda)^2\ ]}$$

$$\cos\sigma = \sin U1 \cdot \sin U2 + \cos U1 * \cos U2 * \cos\lambda$$

$$\sigma = \text{atan}(\sin\sigma / \cos\sigma)$$

$$\sin\alpha = \cos U_1 * \cos U_2 * \sin\lambda / \sin\sigma$$

$$\cos^2\alpha = 1 - \sin^2\alpha$$

$$\cos 2\sigma_m = \cos\sigma - 2 * \sin U_1 * \sin U_2 / \cos^2\alpha$$

$$C = f/16 * \cos^2\alpha * [4 + f \cdot (4 - 3 \cdot \cos^2\alpha)]$$

$$\lambda' = L + (1 - C) * f * \sin\alpha * \{\sigma + C * \sin\sigma * [\cos 2\sigma_m + C * \cos\sigma * (-1 + 2 * \cos^2 2\sigma_m)]$$

λ′ 5 milimetreye indiğinde;

$$u^2 = \cos^2\alpha \cdot (a^2 - b^2)/b^2$$

$$A = 1 + u^2/16384 \cdot \{4096 + u^2 \cdot [-768 + u^2 \cdot (320 - 175 \cdot u^2)]\}$$

$$B = u^2/1024 \cdot \{256 + u^2 \cdot [-128 + u^2 \cdot (74 - 47 \cdot u^2)]\}$$

$$\Delta\sigma = B * \sin\sigma * \{\cos 2\sigma_m + B/4 * [\cos\sigma * (-1 + 2 * \cos^2 2\sigma_m) - B / 6 \cdot \cos 2\sigma_m * (-3 + 4 * \sin^2\sigma) * (-3 + 4 * \cos^2 2\sigma_m)]\}$$

$$s = b * A * (\sigma - \Delta\sigma)$$

$$\alpha_1 = \text{atan}(\cos U_2 * \sin\lambda / \cos U_1 * \sin U_2 - \sin U_1 * \cos U_2 \cdot \cos\lambda)$$

$$\alpha_2 = \text{atan}(\cos U_1 * \sin\lambda / -\sin U_1 * \cos U_2 + \cos U_1 * \sin U_2 * \cos\lambda)$$

(1.2)

$$\sigma_1 = \text{atan}(\tan U_1 / \cos\alpha_1)$$

$$\sin\alpha = \cos U_1 * \sin\alpha_1$$

$$\cos^2\alpha = 1 - \sin^2\alpha$$

$$u^2 = \cos^2\alpha * (a^2 - b^2)/b^2$$

$$A = 1 + u^2/16384 * \{4096 + u^2 * [-768 + u^2 * (320 - 175 * u^2)]\}$$

$$B = u^2/1024 * \{256 + u^2 * [-128 + u^2 * (74 - 47 * u^2)]\}$$

$$\sigma = s / (b*A)$$

$$\cos 2\sigma_m = \cos(2\sigma_1 + \sigma)$$

$$\Delta\sigma = B \cdot \sin\sigma * \{\cos 2\sigma_m + B/4 * [\cos\sigma * (-1 + 2 * \cos^2 2\sigma_m) - B/6 * \cos 2\sigma_m \cdot (-3 + 4 * \sin^2\sigma) * (-3 + 4 * \cos^2 2\sigma_m)]\}$$

$$\sigma' = s / b*A + \Delta\sigma$$

σ′ 5 milimetreye indiğinde;

$$\varphi_2 = \text{atan}(\sin U_1 * \cos\sigma + \cos U_1 * \sin\sigma * \cos\alpha_1 / (1-f) * \sqrt{\sin^2\alpha + (\sin U_1 * \sin\sigma - \cos U_1 * \cos\sigma * \cos\alpha_1)^2})$$

$$\lambda = \text{atan}(\sin\sigma * \sin\alpha_1 / \cos U_1 * \cos\sigma - \sin U_1 * \sin\sigma * \cos\alpha_1)$$

$$C = f/16 * \cos^2\alpha * [4 + f * (4 - 3 * \cos^2\alpha)]$$

$$L = \lambda - (1-C) * f * \sin\alpha * \{\sigma + C * \sin\sigma * [\cos 2\sigma_m + C * \cos\sigma * (-1 + 2 * \cos^2 2\sigma_m)]\}$$

$$\lambda_2 = \lambda_1 + L$$

$$\alpha_2 = \text{atan}(\sin\alpha / -(\sin U_1 * \sin\sigma - \cos U_1 * \cos\sigma * \cos\alpha_1))$$

(1.3)

Düz ve ters metotta bulunan işaretler ve anlamları;

- a, b büyük ve küçük yarı eksenler
- f = düzleştirme oranı $(a-b)/a$
- φ_1, φ_2 = jeodezik enlemler
- L = boylamlar arasındaki fark
- α_1, α_2 azimutlar

2. TOURISTANBUL PROJESİ ANALİZİ VE TASARIMI

Bu bölümde genel olarak SDLC adımları takip edilerek Touristanbul uygulaması için kullanıcı gereksiminimleri ve sistem dizaynı yapılacaktır.

2.1 Sistem Analizi ve İhtiyaçlar

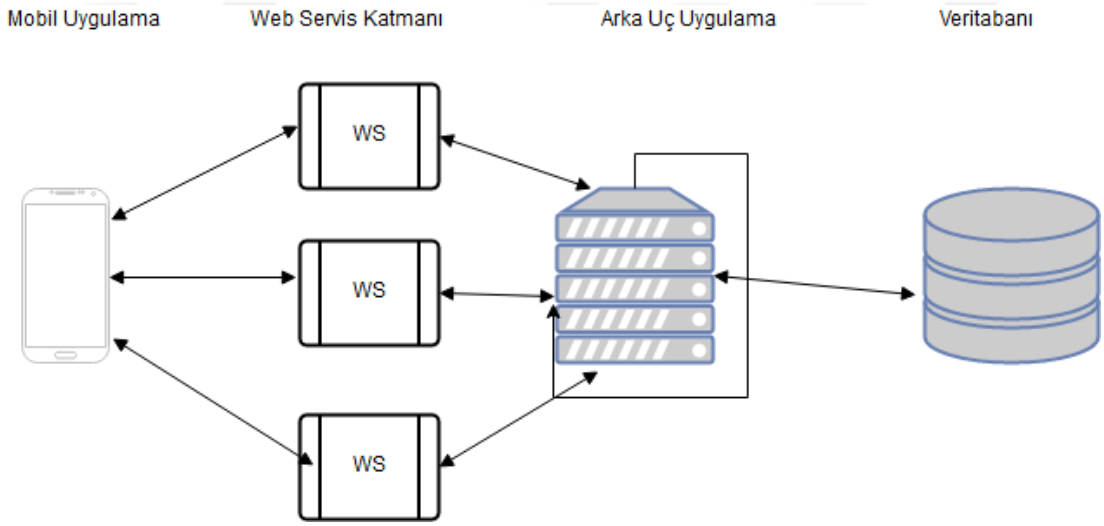
Touristanbul uygulamasının kapsamı; yerli veya yabancı turistlerin İstanbul Tarihi Yarımada'da bulunan anıtları, tarihi ve önemli eserleri veya binaları kamerayla görmelerini ve bu mekanlar hakkında bilgi almalarını sağlamaktır. Uygulama son kullanıcının cihazının enerjisinin az kaldığı durumlarda, iki boyutlu düzlemde çalışmasını sağlamak için haritayla desteklenmektedir. Ayrıca son kullanıcının internete erişimi olmadığı durumlarda da uygulamanın çevirmdışı olarak çalışması üzerine düzenleme yapılacaktır. Genel olarak ihtiyaçlar aşağıda listelenmiştir;

- Kullanıcının mevcut uygulamaya giriş yapabilmesi
- Kullanıcının gerekli bilgilerinin uygulamada tutulması
- Kullanıcının iki boyutlu bir düzlemde ilgili mekanlara ve bu mekanların detay verilerine ulaşabilmesi
- Kullanıcının üç boyutlu düzlemde ilgili mekanları ve bu mekanlara ait detay bilgilerine ulaşabilmesi
- Kullanıcının mobil cihazında ayarlı olan dile göre verilerin bu dilde kullanıcıya gösterilmesi.
- Kullanıcının mobil cihazında ayarlı olan dil sistem tarafından desteklenmiyorsa varsayılan dilin İngilizce olması ve buna göre verilerin sorgulanması
- Kullanıcıya gerekli uyarı yazılarının çıkartılması
- Kullanıcının istediğinde sistemden çıkış yapabilmesinin sağlanması
- Ekranlar arasında akıcı bir şekilde geçiş yapılabilmesi

- Uygulamanın enerji ihtiyaçlarına cevap verebilmesi
- İnternetsiz ortamda uygulamanın destek verebilmesi
- Mobil uygulamanın haberleşeceği bir arka uç uygulama
- Arka uç uygulamanın mobil uygulama için web servisler sağlaması
- Mekansal verilerin tutulacağı konumsal bir veritabanı

2.2 Sistem Tasarımı

İhtiyaçların belirlenmesinden sonra ortaya çıkacak sistemin ana hatları Şekil 2.1'de gösterilmiştir.



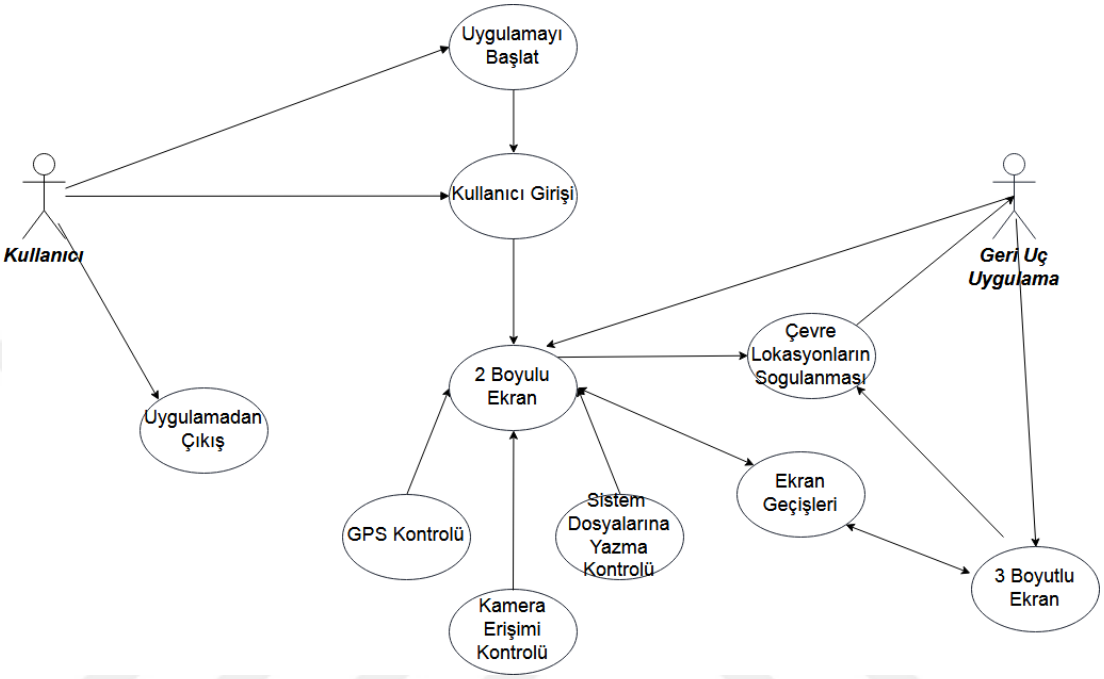
Şekil 2.1 : Genel sistem tasarımı.

2.2.1 Sistem kullanım diagramı

Şekil 2.2'de verildiği üzere Touristanbul uygulamasının kullanım senaryosu oluşturulmuştur. Bu senaryoya göre;

- Kullanıcı uygulamayı başlatır.
- İlk giriş yaptığı durumda herhangi bir sosyal medya hesabıyla kayıt olur
- Kayıt olduktan sonra uygulamanın GPS, kamera ve dosya sistemine erişimi kontrol edilip gerekli yetkiler alınır.
- İki boyutlu harita ekranına çevrede bulunan mekanların verisi getirilir
- Üç boyutlu ekrana isteğe bağlı olarak geçiş yapılabilir

- Kamera ekranında artırılmış gerçeklikle üç yüz altmış derecede mekanlar gösterilir
- İsteğe bağlı olarak harita ve kamera ekranları arasında geçiş yapabilir.
- Kullanıcı isterse uygulamadan çıkabilir

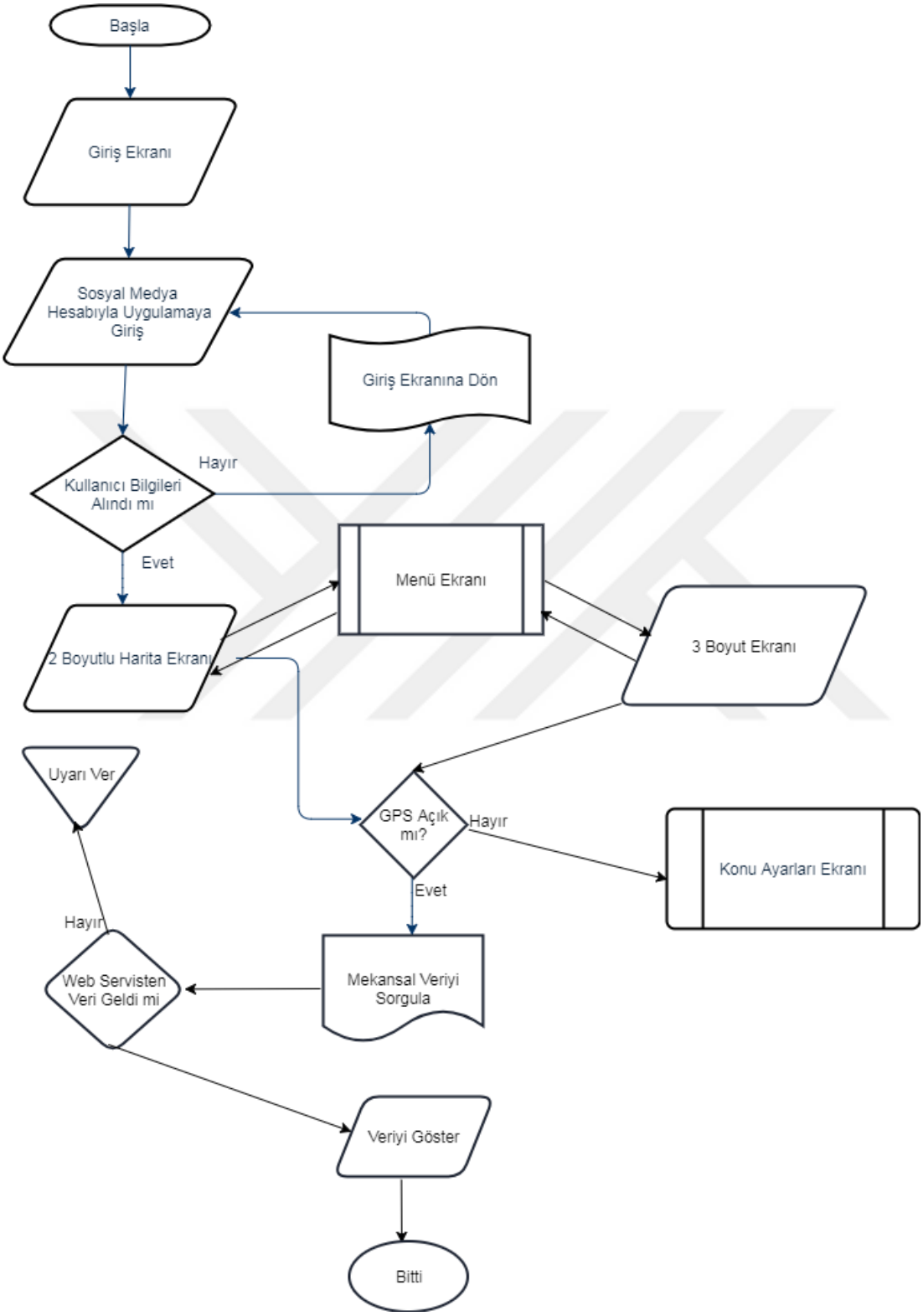


Şekil 2.2 : Touristanbul kullanıcı senaryosu.

2.2.2 Akış diagramları

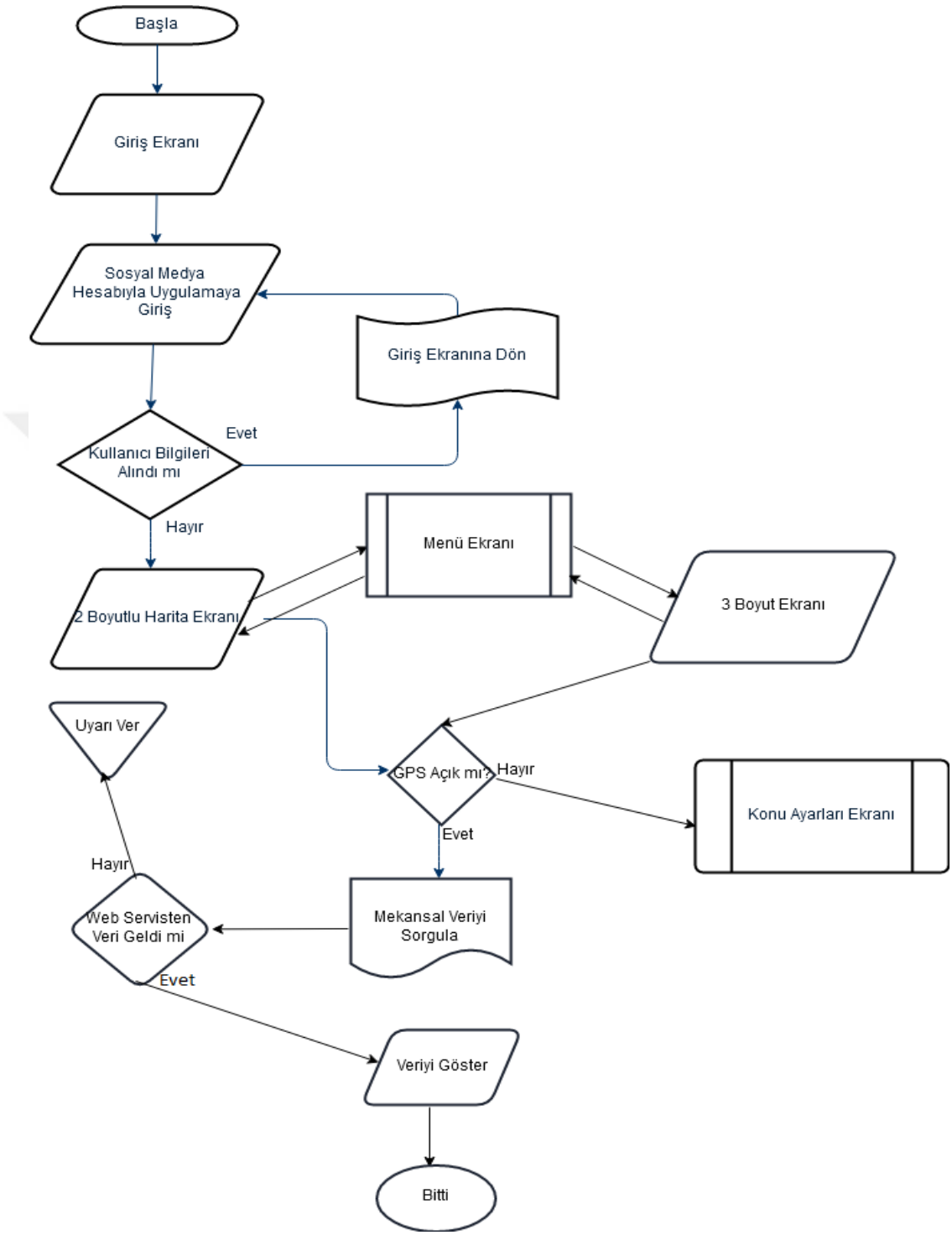
Bu bölümde uygulamanın başlatılmasından kapatılmasına kadar geçen süre içinde yapılabilecek işlemler diagramlar yardımıyla gösterilecektir. Akış diagramları sayesinde yazılımın geliştirilmesi sırasında yazılım mühendisine kullanıcı davranışı hakkında bilgi verilmiş olur. Böylece uygulama içinde kullanıcının potansiyel olarak yapabileceği tüm hareketlerin kapsam içinde kalması sağlanmaya ve uygulamanın hata üretmesinin önüne geçilmeye çalışılır. Şekil 2.3'de iki boyutta mekansal verinin nasıl getirileceğinin akış diagramı verilmiştir. Kullanıcı öncelikle uygulamayı başlatır. Sosyal medya hesaplarından birisiyle giriş yapmaya çalışır. Eğer kullanıcı verisi alınamazsa başlangıç ekranına dönülür. Başarılı giriş yapıldıktan sonra harita ekranı gelir. Bu ekranda GPS'in aktif olup olmadığı kontrol edilir. GPS açık değilse ayarlar sayfasına yönlendirilir ve GPS özelliği aktif hale getirilir. Uygulamaya geri dönülür. GPS ile konum bilgisi alındıktan sonra, bulunulan konumun çevresindeki

mekanlar için sorgulama yapılır. Hiç mekan yoksa uyarı ekranı çıkar. Eğer mekanlar sorgulandıysa haritada lokasyonları gösterilir.



Şekil 2.3: İki boyutta mekansal verinin gösterilmesi akış diagramı.

Şekil 2.4'de artırılmış gerçeklikle verinin nasıl gösterileceğinin akış diagramı verilmiştir.

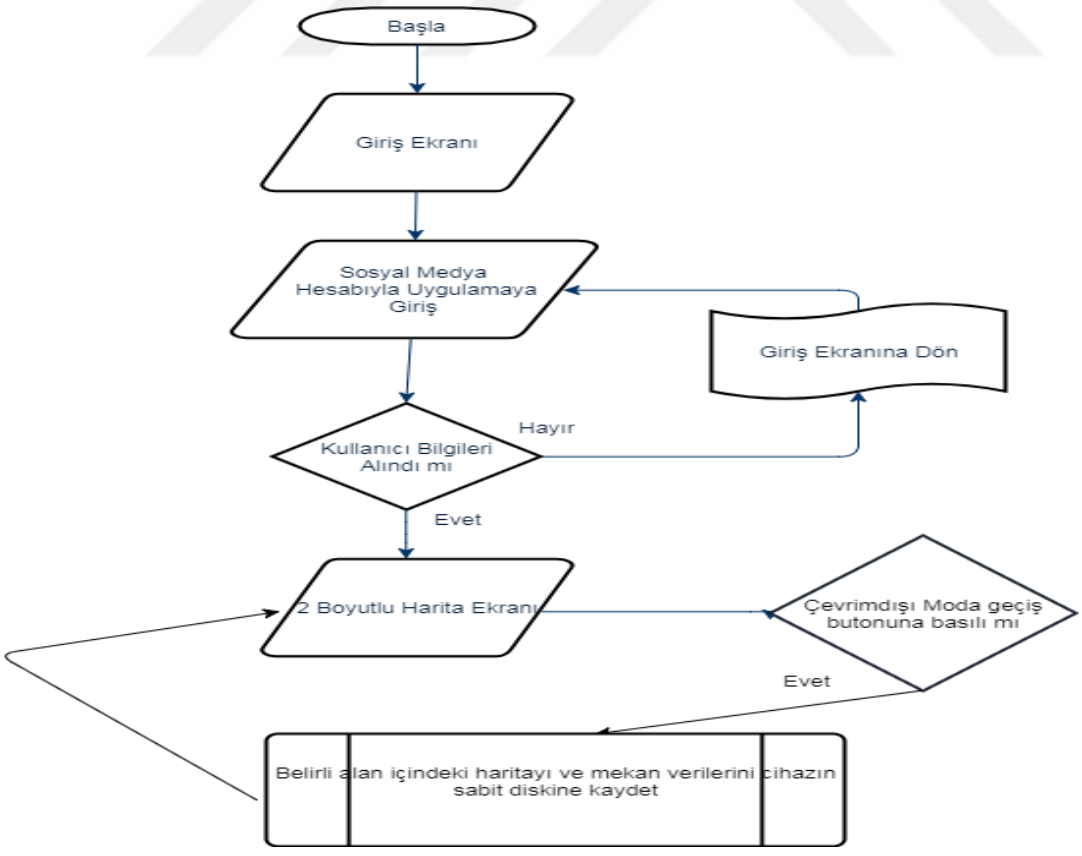


Şekil 2.4 : Üç yüz altmış derecede mekansal verinin gösterilmesi akış diagramı

Kullanıcı bir önceki akış diagramında olduğu gibi uygulamaya giriş yapar. Giriş yaptıktan sonra solda bulunan menü tuşuna basar. Menünün sol üst kısmında yer alan bölümde kullanıcının varsa sosyal medyada bulunan fotoğrafı, adı ve soyadı yer alır.

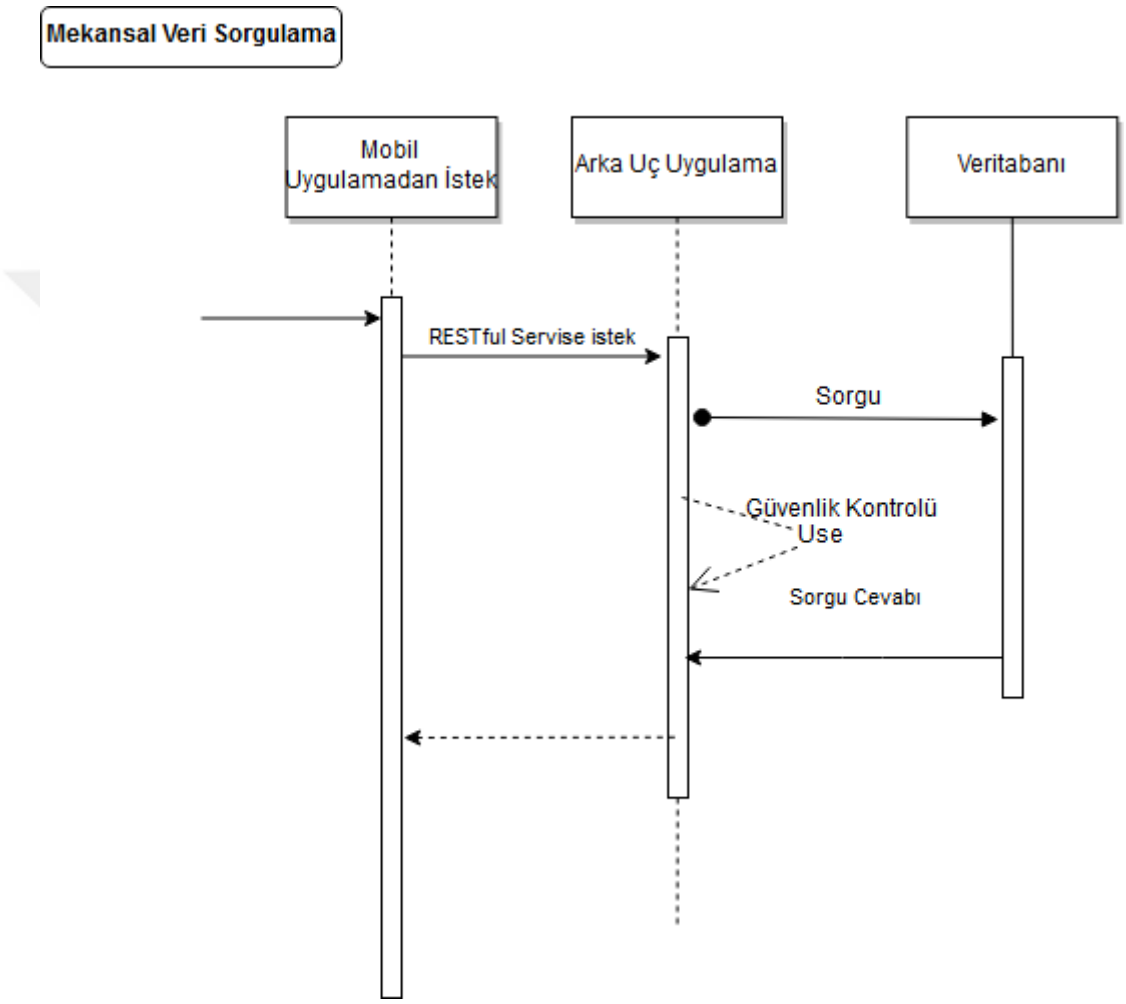
Menü elemanları iki boyutlu ve artırılmış gerçeklikle üç yüz altmış derece görüntüleme ekranlarıdır. Uygulamanın varsayılan ekranı iki boyutlu harita ekranı olduğu için menüden artırılmış gerçeklik ekran sekmesi seçilir. Bu seçimden sonra artırılmış gerçeklik ekranı açılacaktır. Artırılmış gerçeklik ekranında GPS'in aktif olup olmadığı kontrol edilir. GPS açık değilse ayarlar sayfasına yönlendirilir ve GPS özelliği aktif hale getirilir. Uygulamaya geri dönülür. GPS ile konum bilgisi alındıktan sonra, bulunulan konumun çevresindeki mekanlar için sorgulama yapılır. Hiç mekan yoksa uyarı ekranı çıkar. Eğer mekanlar sorgulandıysa, ekranda lokasyonları gösterilir.

Şekil 2.5'de kullanıcının isteğine bağlı olarak çevrimdışı çalışma moduna nasıl geçileceğinin akış diyagramı verilmiştir. Kullanıcı, internete erişemediğinde veya veri kotasından kullanmak istemediğinde, ilk adımda, harita ekranında bulunan indime butonuna basacak, ikinci adımda, kullanmak istediği alanı işaretleyecek ve onay panelinden onayla butonuna basacaktır. İşlemi onayladıktan sonra uygulama seçili alanda bulunan mekanları ve harita katmanlarını sabit diskine indirecektir.



Şekil 2.5 : Çevrimdışı çalışma moduna geçiş.

Şekil 2.6'da veri sorgulamanın nasıl yapılacağı sekans diagramında verilmiştir. Mobil uygulama, GPS üzerinden konum bilgisini aldıktan sonra arka uç uygulamada bulunan web servisleri çağırıp, edindiği konum bilgisiyle çevresinde bulunan mekanların listesini çekecektir. Mekansal verinin önce lokasyon bilgisi ekrana getirelecektir. Eğer kullanıcı detay verisini görmek isterse tekrardan sorgulama yapılacaktır.



Şekil 2.6 : Mekansal veri sorgulama sekans diagramı.

2.2.3 Kullanılacak teknolojiler ve uygulama programlama arayüzleri

Proje kapsamında mobil uygulamanın geliştirilmesi Android(Java) programlama dili kullanılacaktır. Android uygulama geliştirmek için Android Studio IDE'si kullanılacaktır. Ayrıca hataların hızlıca bulunması ve uygulamanın anlık testedilebilmesi için test ortamı olarak Samsung Galaxy S6 model telefon kullanılacaktır. Arka uç uygulama seçenekleri Microsoft .Net ve Java SE 8 olmak

üzere iki adettir. Bu iki platformun arasında seçim yapılırken göz önüne alınması gerekenler aşağıda maddeler halinde verilmiştir;

- Mobil uygulamanın geliştirilmesi esnasında Java programlama dili kullanılacak olması
- Java'nın sunduğu API'lerin daha kapsamlı ve herkes tarafından kullanılır olması
- Performans açısından Javanın daha etkin olması
- Sistem tamamlandıktan sonra veritabanı ve arka uç yazılımın işletim sisteminden bağımsız olarak çalıştırılmak istenmesi
- Çalışma hayatında Java programlama dilinin kullanılması ve bu dile olan yatkınlık

Tüm bu kıstaslar gözönüne alındığında programlama dili olarak Java SE 8'in kullanılmasına karar verilmiştir. Java'nın kullanılmasıyla beraber günümüzde kullanımı yaygınlaşan, performanslı ve modüler yazılım yapmayı kolaylaştıran Spring uygulama arayüzü de kullanılacaktır.

Spring modern, Java tabanlı kurumsal uygulamalar için her türlü dağıtım platformunda kapsamlı bir programlama ve yapılandırma modeli sağlar. Spring'in en önemli unsuru uygulama düzeyinde altyapı desteğidir: Spring, kurumsal uygulamaların üzerine odaklanır [21]. Spring'in özellikleri; RESTful web servislerin kolayca yazılabilmesi, modüler programlama desteği, veritabanıyla kolaylıkla haberleşmesi ve Java'nın diğer modüllerine destek vermesi olarak sıralanabilir. Proje kapsamında Spring'in aşağıda sıralanan modülleri Çizelge 2.1'de gösterilmiştir.

Çizelge 2.1 : Spring modülleri ve özellikleri.

Spring Modülü	Modül Özelliği
Spring Boot	Java ve Spring uygulamasının çalışma ortamını düzenler ve hazır hale getirir
Spring HATEOAS	RESTful web servislerin geliştirilmesinde kullanılır
Spring DATA	Veritabanı bağlantısını sağlar

Veritabanı sistemi olarak konumsal veri tipini tutabileceğimiz bir veritabanı sistemi kullanılmalıdır. Market araştırması yapıldığında konumsal veri tipini tutan veritabanı sistemleri Çizelge 2.2'de gösterilmiştir.

Çizelge 2.2 : Konumsal veritabanı sistemleri.

Konumsal Veritabanı
Oracle Spatial
PostgreSQL
MongoDB
Neo4J
Teradata GeoSpatial
CouchDB
MySQL

Çizelge2.2'de verilen veritabanı sistemlerinden MySQL kullanılacaktır. MySQL kullanılmasının sebepleri;

- Açık kaynak kodlu olması,
- Dökümantasyonunun iyi olması,
- İnternet ortamında kullanışlı bir blogunun olması ve nesne ilişkisel veritabanı olması

MySQL veritabanı geometri, nokta, çizgi ve poligon konumsal veri tiplerini desteklemektedir. Konusal verilerle birlikte konumsal indeks yapısı da kurulabilmektedir. Bu sayede veri sorgulamanın daha performanslı ve hızlı olması sağlanır. Konumsal indeks yapısı R-Ağaç kuadratik bölme algoritması şeklindedir [22]. MySQL'de belirlenen bir noktanın çevresi daire olarak değil kare olarak alınır. Bu sebepten belirli uzaklık içindeki noktalar topluluğu alınmak istendiğinde bu hesaplama daireye göre değil kareye göre yapılmalıdır. Şekil 2.7'de MySQL'de örnek bir konumsal sorgu ve sonucu gösterilmiştir. Sorgu örneğinde, alanları belirlenen bir poligonun geom tablosundaki kapsadığı alanlar sorgulanıp gösterilmiştir.

```
mysql> SET @poly =
    -> 'Polygon((30000 15000,
                 31000 15000,
                 31000 16000,
                 30000 16000,
                 30000 15000))';
mysql> SELECT fid,ST_AsText(g) FROM geom WHERE
    -> MBRContains(ST_GeomFromText(@poly),g);
+-----+-------------------------------------------------------------+
| fid | ST_AsText(g)                                                |
+-----+-------------------------------------------------------------+
|  21 | LINESTRING(30350.4 15828.8,30350.6 15845,30333.8 15845,30 ... |
|  22 | LINESTRING(30350.6 15871.4,30350.6 15887.8,30334 15887.8, ... |
|  23 | LINESTRING(30350.6 15914.2,30350.6 15930.4,30334 15930.4, ... |
|  24 | LINESTRING(30290.2 15823,30290.2 15839.4,30273.4 15839.4, ... |
|  25 | LINESTRING(30291.4 15866.2,30291.6 15882.4,30274.8 15882. ... |
|  26 | LINESTRING(30291.6 15918.2,30291.6 15934.4,30275 15934.4, ... |
| 249 | LINESTRING(30337.8 15938.6,30337.8 15946.8,30320.4 15946. ... |
|   1 | LINESTRING(30250.4 15129.2,30248.8 15138.4,30238.2 15136. ... |
|   2 | LINESTRING(30220.2 15122.8,30217.2 15137.8,30207.6 15136, ... |
|   3 | LINESTRING(30179 15114.4,30176.6 15129.4,30167 15128,3016 ... |
|   4 | LINESTRING(30155.2 15121.4,30140.4 15118.6,30142 15109,30 ... |
|   5 | LINESTRING(30192.4 15085,30177.6 15082.2,30179.2 15072.4, ... |
|   6 | LINESTRING(30244 15087,30229 15086.2,30229.4 15076.4,3024 ... |
```

Şekil 2.7 : Örnek MySQL konumsal sorgusu.

Mobil uygulamanın ön yüzünde AR görünümü sağlayabilmek ve uygulamanın hızlı bir şekilde geliştirilmesi için hazır API kullanılmıştır. Öncesinde AR üzerine API sağlayan uygulamalar araştırılmıştır. Araştırma esnasında aşağıda belirtilen maddelere dikkat edilmiştir;

- Seçilecek API'nin açık kaynak kodlu veya ücretli olup olmaması
- IOS ve Android ortamına destek vermesi, lokasyon bazlı AR desteğinin olması
- Html 5 desteğinin olması ve JavaScript kütüphanesine destek vermesi
- Detaylı dökümantasyonunun olması
- Sosyal medya topluluğunun olması
- Kolay kullanılması
- Teknik desteğinin iyi olması

Bu gereksinimler göz önüne alınarak Şekil 2.8'de genel bir karşılaştırma yapılmıştır [23].

	Type	iOS	Unity (3D)	Marker	NaturalFeature	Windows Mobile	Web	PC/Mac /Linux	3D Object Tracking	GPS	IMU Sensors	VisualSearch	FaceTracking	ContentAPI
Metaio SDK (now Apple inc)	Free + Commercial SDK option	✓	✓	✓	✓		✓	PC/Mac	✓	✓	✓	✓ Client-based +100 unique objects, cloud-based continuous visual search engine	✓	✓ OpenGL support, in-house 3-D renderer
ARLab	Free + Commercial SDK option	✓		✓ QR code						✓	✓	✓ Support for thousands of images in pools of 50-60 images	✓	
ARToolkit	Open Source	✓	✓	✓ Basic	✓	✓		PC/Mac /Linux	✗					
Vuforia	Free + Commercial SDK option	✓	✓	✓ Advanced + VUMark	✓	✓ Vuforia now available for windows app dev & also for MS Hololens	✗		✓ Only on box and cylinder and small size 3D objects too	✓	✓	✓	✗	✓ With Vuforia Cloud
Wikitude	Free + Commercial SDK option	✓	✓ 3D Tracking Included	✓ Advanced	✓	✓			✓ Beta	✓	✓	✓ Cloud Recognition and Offline (on device)	⊖ Face Detection	✓ with Wikitude Studio and Cloud Recognition
Win AR	Free + Commercial SDK option				✓									✓

Şekil 2.8 : Artırılmış gerçeklik API'lerinin karşılaştırılması.

Yapılan karşılaştırmalar ve araştırmalar sonucunda; dökümantasyon, teknik destek ve API kullanımının kolaylığı maddeleri önceliklendirilmiştir. Bunun sebebi geliştirme ortamının hızlı bir şekilde kurulumu ve demo amaçlı bir uygulamanın hızlıca çıkarılabilmesidir. Tüm bu önceliklendirmeler ve amaçlar doğrultusunda Wikitude API kullanılmasına karar verilmiştir. Wikitude API üzerinde bir çok özellik barındırır. Bu özellikler;

- Wikitude - Native API
- Wikitude - JavaScript API
- Wikitude - Plugins API

Proje kapsamında sadece Native API ve JavaScript API kullanılmıştır. Native API, Android ve IOS için mobil cihazın görüş motorunun kullanılmasını sağlar. Ayrıca Plugin API ve JavaScript API ile beraber iletişimi sağlayan ana modüldür. JavaScript API, HTML ve Javascript bazlı artırılmış gerçeklik yazılımı yapabilmek için kullanılan modüldür. Bu modül de Android ve IOS için kullanılabilir. Lokasyon bazlı AR uygulamaları için kullanılan ana modüldür [24].

İki boyutta konumsal verinin gösterilmesi için bir harita altlığına ihtiyaç duyulacaktır. Harita altlığı seçilirken aşağıdakı maddeler önemlidir;

- Yapılacak değişiklerde kısıtlamasının olmaması
- Belirli bir kullanım ve sorgu sayısı sonrasında ek ücretlendirme olmaması
- Mümkün ise açık kaynak kodlu olması
- Güncel JavaScript kütüphaneleri desteklemesi
- Html 5 ve Mobil desteğinin olması
- İnternetsiz ortamda haritayı gösterebilmesi

Çizelge 2.3’de güncel harita uygulamaları gösterilmiştir.

Çizelge 2.3 : Güncel harita uygulamaları ve üretici firmalar.

Harita Uygulaması	Üretici Firma
Google Maps	Google
Bing Maps	Bing Microsoft
Yandex Maps	Yandex
Open Street Map	GPU Açık Kaynak

Tüm istekler sıralandığında Open Street Map (OSM) uygulamasının açık kaynak kodlu olması ve OSM üzerinde JavaScript kütüphanelerinin kullanılabilmesi sonucu harita altlığı olarak OSM kullanılmasına karar verilmiştir.

Harita altlığının seçilmesinden sonra bu altlığı kullanan JavaScript kütüphaneleri araştırılmıştır. Çizelge 2.4'de harita oluşturmak için kullanılan örnek kütüphaneler ve kullanıldıkları haritalar verilmiştir.

Çizelge 2.4 : Harita oluşturmak için kullanılan güncel JavaScript kütüphaneleri [25].

JavaScript Kütüphanesi	Harita Altlığı	Lisans
GMaps	Google Maps	MIT
jHere	Here Maps	MIT
Kartograph	Kartograph	AGPL & LGPL
Mapael	Raphael.js	MIT
DataMaps	D3js	MIT
GeoChart	Google Maps	Google Maps TOS
Maplace	Google Maps	MIT
OpenLayers	Open Street Map	
Leaflet	Open Street Map	

Harita altlığı olarak OSM kullanılacak olması, OpenLayers ve Leaflet'in mobil desteklerinin olması sonucu, bu iki kütüphaneden birinin seçilmesi gerekmiştir. Kullanım kolaylığı ve dökümantasyonunun iyi olması sebebiyle OpenLayers kullanılmasına karar verilmiştir. Sonuç olarak iki boyutlu harita ekranında OSM ve Openlayers kullanılarak konumsal veriler gösterilecektir.

2.3 Benzer Uygulamalar

Yapılmak istenen uygulamaya fikren benzeyen ve hali hazırda kullanımda olan uygulamalar mevcuttur. Bu bölümde benzer uygulamalar hakkında bilgi verilecektir.

2.3.1 Layar

Layar, kullanıcının dış dünyayla ilgili içeriği görüntülemesi için cep telefonunun dahili kamerasını, pusulasını ve GPS'ini kullanıyor. Kullanıcı örneğin, belirli semtler hakkında bilgiler edinebilir ve geçtiği restoranların yorumlarını okuyabilir [26].

2.3.2 Tuscany+

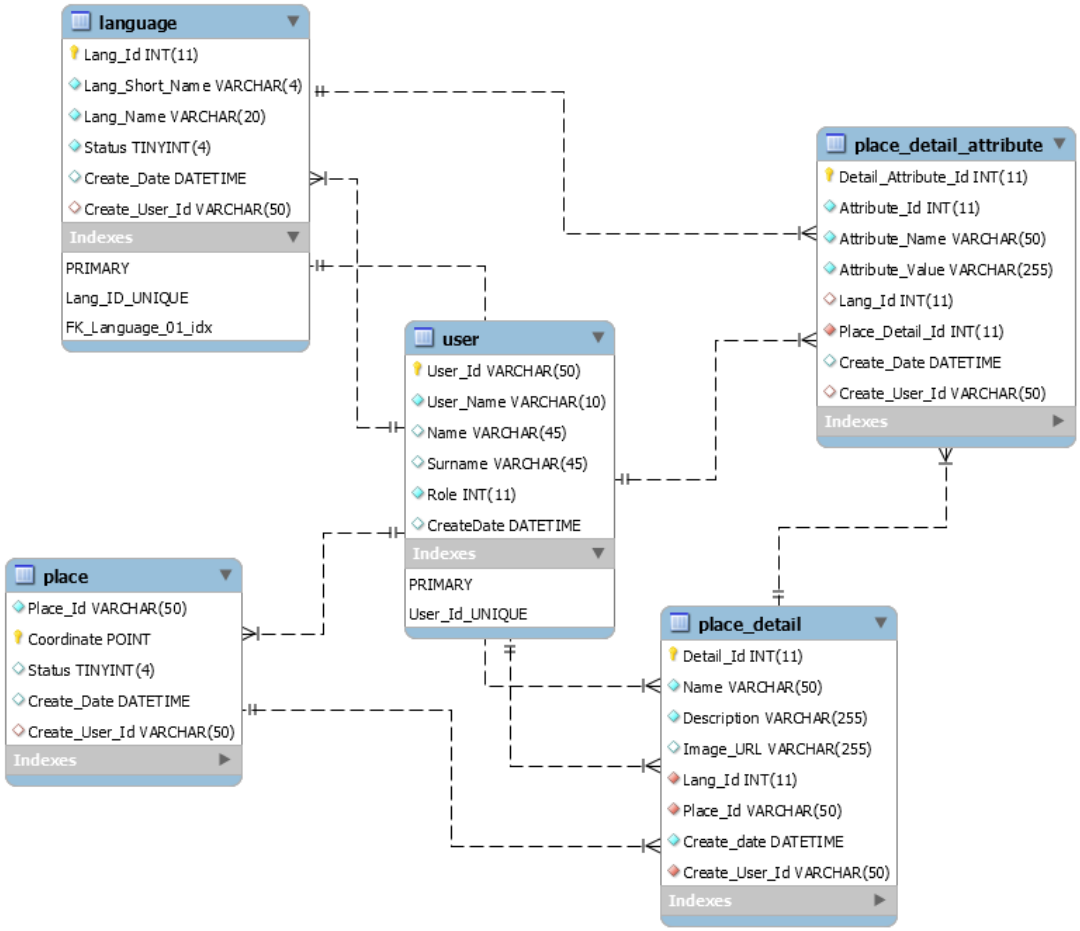
Tuscany +, turizm sektörü için ilk AR uygulamasıdır. Telefonun kamerasıyla gerçek bir alana işaret edildiğinde, multimedya, sanal öğeler, coğrafi bölgelere özgü veriler veya web sitelerinden gelen bilgiler gibi bir dizi bilginin yer aldığı yenilikçi teknolojiyi temel alır [27]. Bu yönüyle projede yapılmak istenen uygulamaya en çok benzerlik gösteren uygulamadır. Şekil 2.9'da Tuscany+'a ait bir ekran gösterilmiştir [27].

Şekil 2.9 : Tuscany+.

3. TOURİSTANBUL PROJESİ VERİTABANI TASARIMI VE ARKA UÇ UYGULAMASI

3.1 Veritabanı Tasarımı

Veritabanı dizaynı esnasında kurulan yapının esnek ve genişletilebilir olmasına dikkat edilmiştir. Oluşturulan tablolarda, verilerin tekrar etmemesi üzerinde durulmuştur. Gelecekte uygulamaya yeni bir özellik eklenmek istendiğinde mevcut tasarımın üzerinden bu eklemelerin yapılabilmesi sağlanacaktır. Şekil3.1'de veritabanı tasarımı gösterilmiştir.

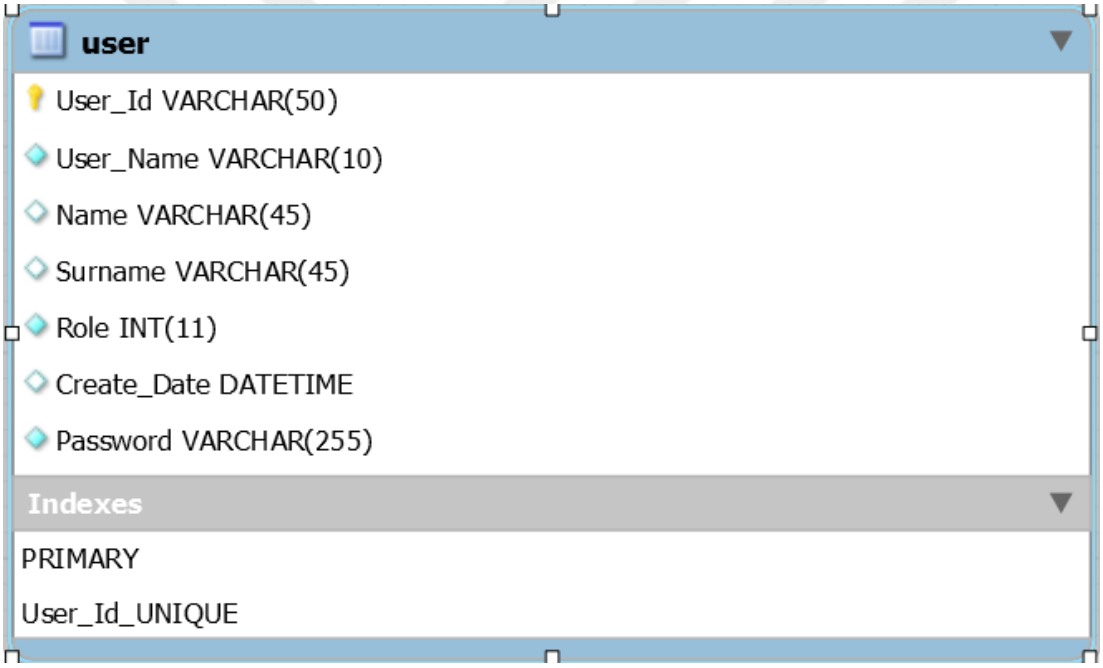


Şekil 3.1 : Touristanbul veritabanı tasarımı.

Şekil 3.1'de veritabanı tasarımı görüldüğü üzere veri, beş tablo üzerinde tutulmuştur. Bu tablolar; kullanıcı (user), dil (language), mekan (place), mekan detayı (place_detail) ve mekan detay özellikleri (place_detail_attribute) tablolarıdır.

3.1.1 Kullanıcı tablosu

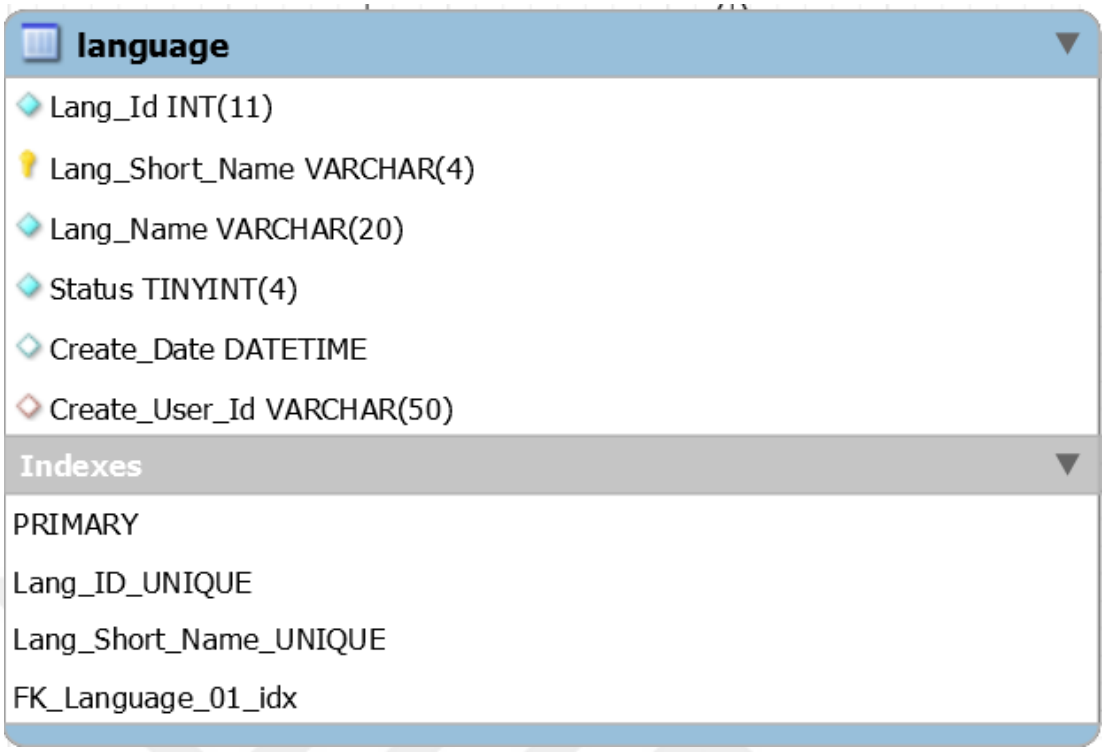
Sistemde ilk olarak oluşturulması gereken veri kullanıcı verisidir. Bir kullanıcı üç rolde bulunabilir. Bunlar; yeni, normal ve yönetici rolleridir. Bu yapıya göre sistemde yönetici olarak tanımlanan kullanıcılar, sisteme yeni dil, lokasyon, lokasyon detayı ekleme, güncelleme, silme işlemlerini yapabilirler. Şekil 3.2'de kullanıcı tablosu detaylı olarak gösterilmiştir. Kullanıcı tablosundaki user_id alanı her kullanıcı kaydı için özel ve bir tanedir.



Şekil 3.2 : Kullanıcı tablosu.

3.1.2 Dil tablosu

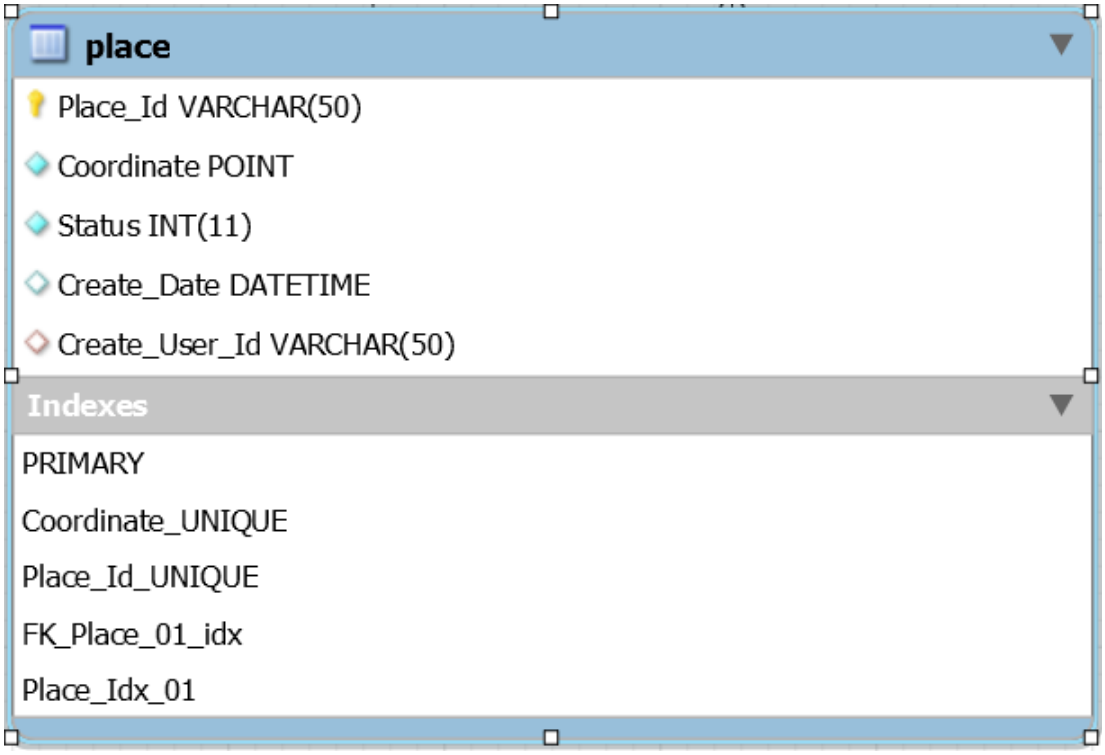
Sistemde son kullanıcılara sunulacak dil opsiyonlarının tutulduğu tablodur. Bu tablo sayesinde bir lokasyonun detayının hangi dilde olacağı belirlenebilir. Tabloya eklemeyi sadece yönetici pozisyonundaki kullanıcılar yapabilir. Şekil 3.3'de dil tablosu detaylı olarak gösterilmiştir. Dil tablosundaki lang_id alanı her dil için özel ve bir tanedir. Ayrıca tablo üzerindeki FK_Language_01_idx dış indeksi tablonun kullanıcı tablosuyla ilişkisini tanımlar. İndekse göre bir kullanıcı birden fazla dil tanımlayabilir yani tablolar arasında bir ile çok ilişki vardır.



Şekil 3.3 : Dil tablosu.

3.1.3 Mekan tablosu

Lokasyonların tutulduğu tablodur. Mekan tablosu Şekil 3.4'de gösterilmiştir.

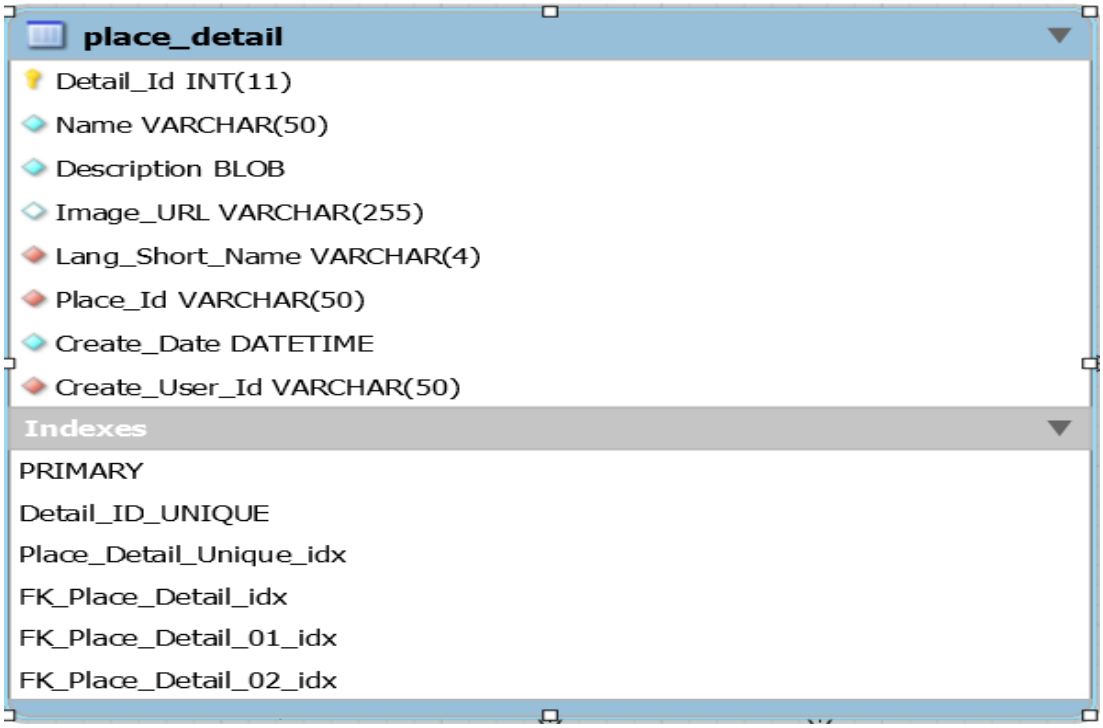


Şekil 3.4 : Mekan tablosu.

Bu tabloda sadece; ilgili lokasyonun koordinat bilgileri, oluşturulma tarihi, oluşturan kullanıcı, lokasyonun aktif olup olmadığının bilgisi yer almaktadır. Mekan tablosu içinde ilgili mekan ile herhangi bir detayın tutulmamasının sebebi, verilerin tekrar edilmesini önlemektir. Tasarım esnasında kullanılan mantığa göre, bir mekanın birden çok dilde detay bilgisi olabilir. Bu sebepten, detayların tutulduğu veriler ayrı bir tabloda tutulmaktadır. Ayrıca bir kullanıcı birden fazla mekan tanımlayabilir. Place_id alanı ve coordinate alanı üzerindeki indeksler bir mekanın tek olmasını sağlar. Tablo üzerinde mekan detayı tablosuyla bire ile çok ilişkisi vardır. Buna göre bir mekanın birden çok mekan detayı olabilir.

3.1.4 Mekan detayı tablosu

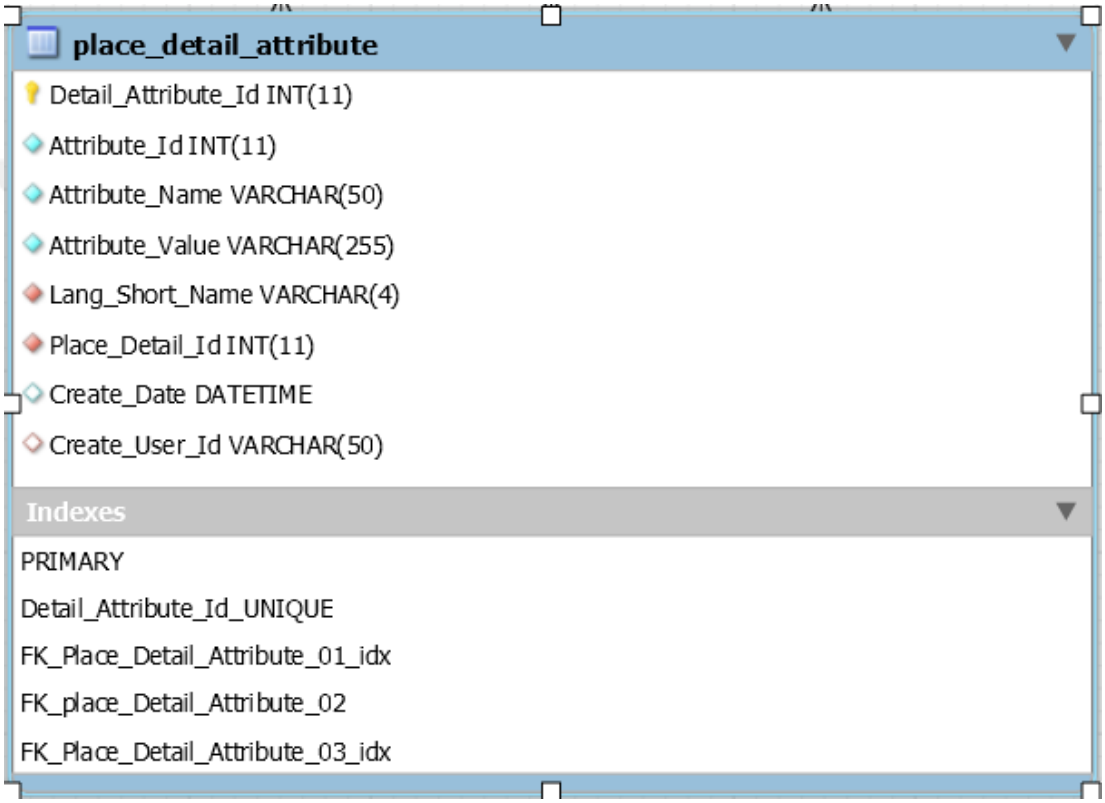
İlgili mekanların detaylarının tutulduğu tablodur. Mekanların detay bilgileri, dil tablosunda tanımlanan diller kadar çeşitlilik gösterebilir. Örnek olarak sistemde Türkçe ve İngilizce dilleri tanımlıysa bir mekanın detay bilgisi bu iki dilden biri veya ikisinde verilmiş olabilir. Mekan detayı tablosunda ilgili mekanın adı, varsa görselinin dosya bilgisi, kısa açıklaması, dil seçenekleri tutulmaktadır. Şekil 3.5'de mekan detayı tablosu gösterilmiştir.



Şekil 3.5 : Mekan detayı tablosu.

3.1.5 Mekan detay özellikleri tablosu

Mevcut detay bilgisine ek olarak, bir mekanla ilgili daha fazla özel detay girişi yapılmak istenmesi durumu için bu tablo eklenmiştir. Bir mekanla ilgili genel bilgiler detay alanına metin olarak yazılabileceği gibi, bu metin parçalanıp Place_Detail_Attribute tablosuna da yazılabilir. Örnek olarak, mekanın yapılış tarihi, mimarı, kat bilgisi, yapılış mimarisi. Şekil 3.6'da mekan detay özellikleri tablosu gösterilmiştir.



Şekil 3.6 : Mekan detay özellikleri tablosu.

Mekan detayı tablosu ile mekan detay özellikleri tablosu arasında bir ile çok ilişki vardır. Ayrıca farklı dillerde mekan detay özellik bilgileri olabileceği için dil tablosu ve mekan detay özellikleri tablosu arasında çok ile çok ilişki vardır.

3.2 Arka Uç Uygulama Tasarımı

Veritabanı oluşturulduktan sonra, eldeki verilerin düzgün bir şekilde sunulmasını sağlama ve verinin sorgulanması esnasında oluşacak yoğunluğu düzgün bir şekilde yönetebilme özelliklerine sahip bir ana yazılım tasarımı yapılmıştır. Tasarımı yapılan yazılım arka uç uygulama olarak isimlendirilir. Yazılım geliştirme esnasında en çok

dikkat edilmesi gereken husus, yazılımın modüler olmasıdır. Modüler yapı sayesinde mevcutta kullanılan yazılım kolaylıkla güncellenebilir ve yeni teknolojilere entegre edilebilir. Kodlama esnasında bu modülerliği sağlamak için genellikle uygulama katmanlara ayrılır ve her katmanın kendi görevi olur. Bu sayede hem ilgili katmanla ilgili yeni bir teknoloji çıktığında bu teknoloji kullanılabilir hem de hata ayıklamak kolaylaşır. Arka uç uygulama, üç katmandan oluşmaktadır. Bu katmanlar; veri katmanı, iş mantığı katmanı ve servis katmanıdır.

3.2.1 Veri katmanı

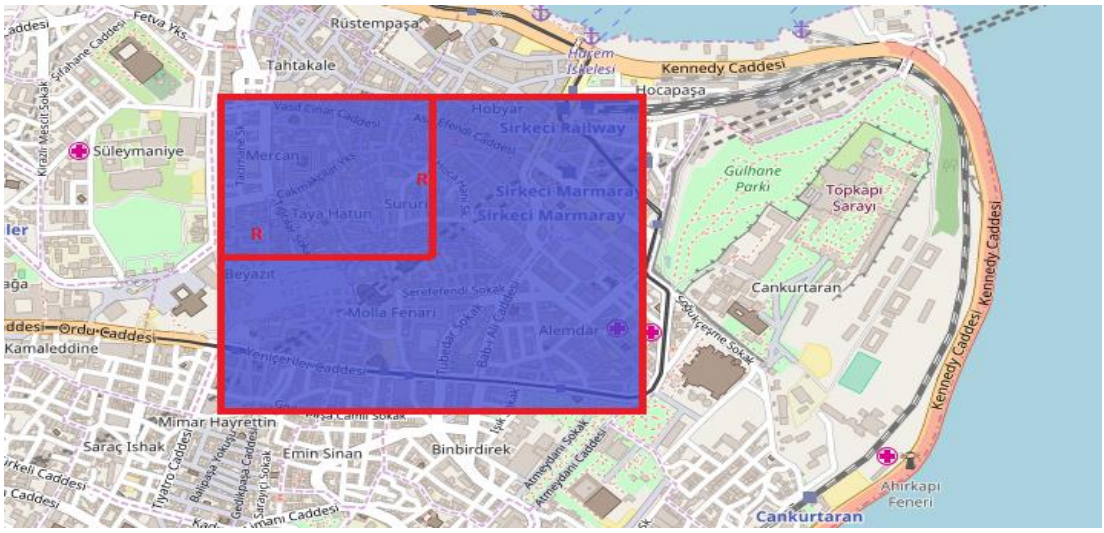
Veri katmanı bir uygulamanın ilk ve en önemli katmanıdır. Bu katmanda veritabanı bağlantıları, verinin veritabanına eklenmesi, güncellenmesi, silinmesi ve getirilmesi işlemlerini yapan fonksiyonlar bulunur.

3.2.2 İş mantığı katmanı

İş mantığı katmanı servis katmanıyla veri katmanı arasındaki bağlantıyı sağlar. Bu katmanda veriyle ilgili değişiklikler, eldeki veri kullanılarak yeni veri üretilmesi, sistem raporlarının üretilmesi gibi işlemleri yapan fonksiyonlar bulunur.

3.2.2.1 Belirli bir alan içindeki mekanların bulunması

Mekanlar veritabanında birer nokta geometrisi olarak tutulmaktadır. Uygulamanın belirli uzaklıktaki mekanları arka uç uygulamadan çekebilmesi için kullanıcının enlem boylam ve uzaklık bilgisi giriş parametresi olarak verilmelidir. Giriş parametresi olarak verilen enlem ve boylam bilgisi bir karenin orta noktası olacak şekilde ilk adımda enlem bilgisine uzaklık parametresi eklenerek ve çıkartılarak karenin üst ve alt kenarları belirlenir. İkinci adım olarak boylam bilgisine uzaklık parametresi eklenerek ve çıkartılarak karenin sol ve sağ kenarları belirlenir. Oluşturulan kare içinde kalan mekanlar MySQL'in MBRCONTAINS fonksiyonuyla [28] bulunur. Şekil 3.7'de taralı alan içindeki mekanlar veritabanından getirilir.



Şekil 3.7 : Belirli bir alan içindeki mekanların bulunması.

3.2.2.2 Mekan ve Kullanıcı arasındaki uzaklığın hesaplanması

Enlem boylam bilgisi verilen iki nokta arasındaki uzaklık bulunurken iki boyutlu kartezyen düzlemi kullanılamaz. Kartezyen düzleminin kullanılması durumunda mesafe hesabında dünyanın şeklinin küremsi olmasından dolayı hata olacaktır. Kare alan içinde kalan mekanların kullanıcıya olan uzaklığını bulabilmek için Haversine ve Vicenty formüllerini kullanarak mesafe hesaplayan bir program yazılmıştır. Formülleri denemek için öncelikle mobil cihazın GPS'i kullanılarak kare alan içinde kalacak şekilde dört lokasyonun enlem ve boylam bilgileri ölçülmüştür. İkinci adım olarak kare alanın köşegenlerinin kesişim noktasının enlem ve boylam bilgileri GPS kullanılarak 41.0084 enlem ve 28.9798 boylam olarak ölçülmüştür. Üçüncü adımda ölçülen kesişim noktasından diğer dört lokasyona olan uzaklıklar yazılan program yardımıyla ölçülmüştür. Çizelge 3.1'de yapılan ölçümler verilmiştir.

Çizelge 3.1 : Haversine ve Vicenty ölçümleri.

Haversine formülü uzaklık (metre)	Vicenty formülü uzaklık (metre)
775.854	775.185
1116.366	1118.041
1114.096	1114.874
1571.098	1573.345

Vicenty formülü ve Haversine formülü arasındaki fark iki metrenin üzerine çıkmamıştır. Ancak Haversine formülünün basit oluşu ve uygulama üzerinde daha hızlı çalışması sebebiyle ölçümlerde Haversine formülü kullanılmıştır.

3.2.3 Servis katmanı

Servis katmanı, iş mantığı katmanından aldığı işlenmiş veriyi Restful servislerle sağlayan bir haberleşme katmanıdır. Bu katman, JSON tipinde veri gönderip alır. Servis katmanı üç tane Restful servis sağlamaktadır.

3.2.3.1 Mekan getir RESTful servisi

Belirli bir alan içinde kalan mekanları getiren RESTful servistir. Servis giriş parametreleri olarak kullanıcının enlem, boylam ve uzaklık bilgilerini alır. Çıktı olarak kullacının belirlediği uzaklık bilgisi kadar uzaklıktaki tüm mekanları bir liste halinde getirir. Bu servisin çıktısı Şekil 3.8'de gösterilmiştir.

Şekil 3.8 : Mekan getir RESTful servisi JSON çıktısı.

3.2.3.2 Mekan detayı getir RESTful servisi

Bir mekanın istenen dilde detay bilgisini getiren RESTful servistir. Bu servisin giriş parametreleri olarak ilgili mekanın tanımlayıcı bilgisi ve dil türünü alır. Bu servisin çıktısı Şekil 3.9'da gösterilmiştir. Servis, çıktı olarak iş mantığı katmanı vasıtasıyla mekan detayı tablosundaki ilgili veriyi sorgular ve bunu JSON'a cevirip mobil uygulamaya sunar. Her özellik, özellik adı ve açıklaması alanlarından oluşur.


```
{
   "success": true,
   "description": "Success",
  -"returnData": {
      "id": null,
      "placeDetailName": "Topkapi Sarayi",
      "placeDetailDesc": "Topkapı Sarayı, İstanbul Sarayburnunda, Osmanlı İmparatorluğunun 600 yıllık tarihinin 400 yılı boyunca,
      devletin idare merkezi olarak kullanılan ve Osmanlı padişahlarının yaşadığı saraydır. Bir zamanlar içinde 4.000e yakın
      insan yaşamıştır.",
      "placeDetailLang": null,
      "placeDetailPhotoURL": "",
      "placeLat": null,
      "placeLong": null,
      "wktText": null
   }
}
```


Şekil 3.9 : Mekan detayı getir RESTful servisi JSON çıktıları.

3.2.3.3 Mekanlar ve detayları RESTful servisi

Belirli bir alan içinde kalan mekanları, seçilen dile göre detaylarıyla birlikte getiren Restful servistir. Servis giriş parametreleri olarak kullanıcının enlem, boylam uzaklık ve dil türü bilgisini alır. Bu bilgileri aldıktan sonra arka uç uygulamadan sorgular ve bir liste halinde mobil uygulamaya geri döner. Bu servis kullanıcının çevrimdışı moda geçmesi durumunda lokasyonlarla alakalı tüm bilgileri sorgulayabilmek için yazılmıştır. Bu servisin çıktısı Şekil 3.10'da gösterilmiştir.

```
-1: {
    "placeId": "6fc4704b-aecc-4479-9b35-64d1ef132c27",
    "wktText": "POINT(28.9804 41.0128)",
    "distance": 433.16552566734975,
    "latitude": 28.9804,
    "longitude": 41.0128,
    "name": "Gulhane Parki",
    "description": "Gulhane Parki",
    "image_url": "",
    "lang_short_name": "TR"
 },
-2: {
    "placeId": "d73e2edb-8cbc-49ca-9ac3-1d2751ac8805",
    "wktText": "POINT(28.9837 41.0131)",
    "distance": 630.1321805455607,
    "latitude": 28.9837,
    "longitude": 41.0131,
    "name": "Topkapi Sarayi",
    "description": "Topkapı Sarayı, İstanbul Sarayburnunda, Osmanlı İmparatorluğunun 600 yıllık tarihinin 400 yılı boyunca,
    devletin idare merkezi olarak kullanılan ve Osmanlı padişahlarının yaşadığı saraydır. Bir zamanlar içinde 4.000e yakın
    insan yaşamıştır.",
    "image_url": "",
    "lang_short_name": "TR"
 },
-3: {
    "placeId": "47f175ed-9761-4afe-b1f1-6718706c50ad",
    "wktText": "POINT(28.9768 41.015)",
    "distance": 723.4990784282295,
    "latitude": 28.9768,
    "longitude": 41.015,
    "name": "Sirkeci Gari",
    "description": "Sirkeci Gari",
```


Şekil 3.10 : Mekanlar ve detayları RESTful servisi.

Mekanlar ve detayları servisi mobil uygulamanın çevrimdışı modu için hazırlanmıştır. Kullanıcı çevrimdışı moda geçerken belirli bir alanı seçtiğinde mobi uygulama bu servisi çağırıp mekanları ve detaylarını elde eder.

3.2.4 Arka uç Uygulama erişim güvenliği

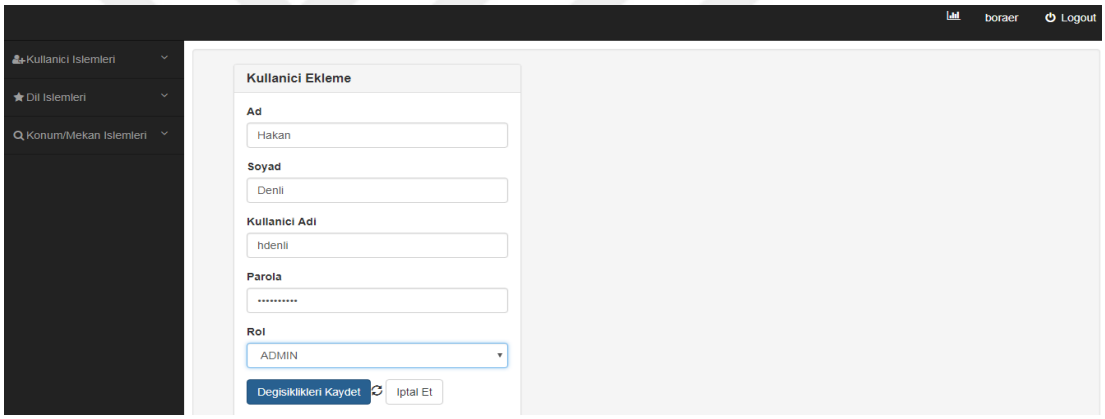
Arka uç uygulamaya dış dünyadan erişebilmek için bir kimlik doğrulama işleminin gerçeklenmesi gereklidir. Arka uç uygulamaya bağlanarak, uygulamanın sağladığı servisleri kullanmak isteyen diğer uygulamalar kullanıcı adı ve parola ile sisteme kayıtlı olup olmadıkları sorgulanarak kimlik doğrulama işleminden geçirilirler. Sisteme bağlanmak isteyen uygulamalar kimlik doğrulama işleminin ardından servis katmanına erişebilirler ve arka uç uygulamanın RESTful servislerini kullanabilirler.

3.3 Arka Uç Uygulama Yönetim Paneli

Arka uç uygulamada veri girişini ve yönetimini sağlayabilmek için bir yönetim paneline ihtiyaç duyulmuştur. Yönetim paneli; kullanıcı, dil ve mekan yönetimini sağlayacak altyapıyı sunacak şekilde hazırlanmıştır.Yönetim panelinin kodlanmasında Spring Boot çatısı ve ön yüz tasarımında Thymeleaf ve Bootstrap kullanılmıştır. Thymeleaf sunucu bazlı web sayfalarının tasarımını sağlayan bir kütüphanedir [29]. Bootstrap ise, web sayfalarının tasarımını kolayca yapmamızı sağlayan ve her cihazın web tarayıcısına göre sayfaların ekranda düzgünce gözükmesinie sağlayan bir kütüphanedir [30].

3.3.1 Yeni kullanıcı ekleme ekranı

Şekil 3.11'de yeni kullanıcı ekleme ekranı gösterilmiştir.

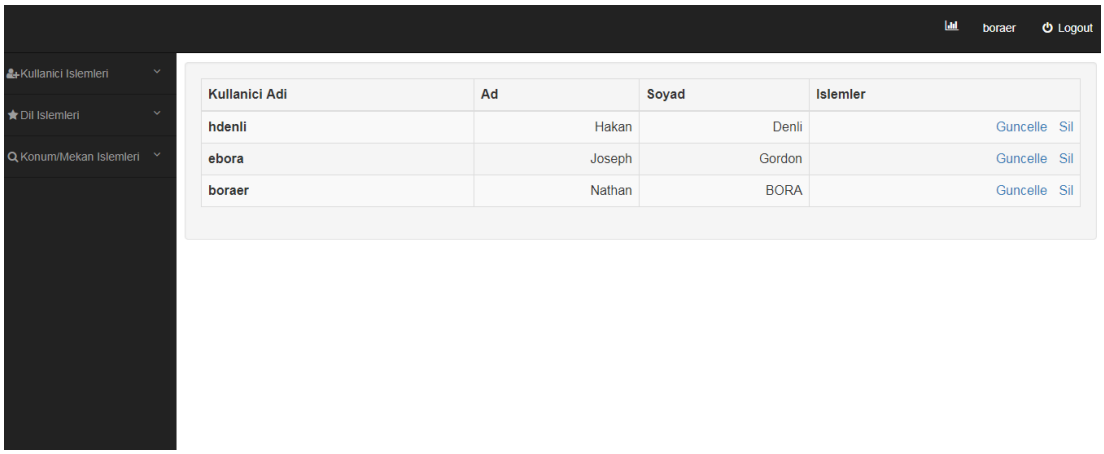


Şekil 3.11 : Yeni kullanıcı ekleme ekranı.

Yeni kullanıcı ekleme ekranından sisteme yeni bir kullanıcı eklenebilmektedir. Kullanıcının rolü normal veya yönetici olabilir.

3.3.2 Kullanıcı listeleme ekranı

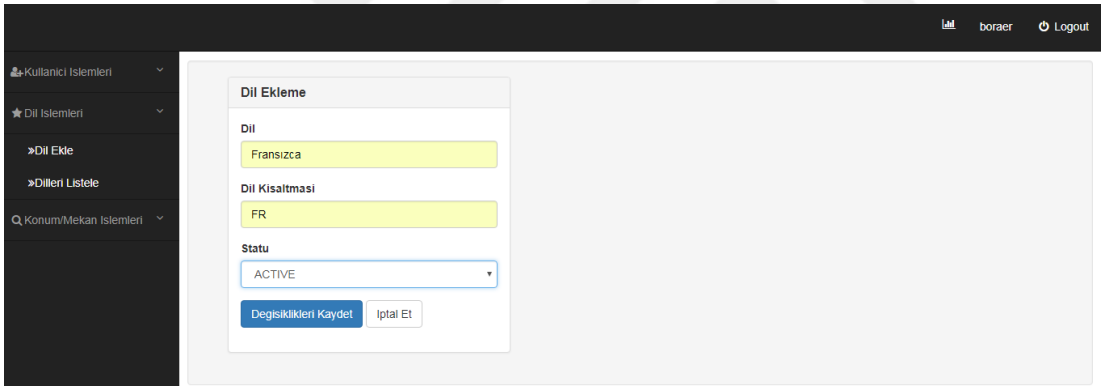
Bu ekran sistemde tanımlanan tüm kullanıcıların listelendiği ekrandır. Bu ekrandan istenen kullanıcı üzerinde güncellenme veya silme işlemi yapılabilir. Şekil 3.12'de kullanıcı listeleme ekranı gösterilmiştir.



Şekil 3.12 : Kullanıcı listeleme ekranı.

3.3.3 Yeni dil ekleme ekranı

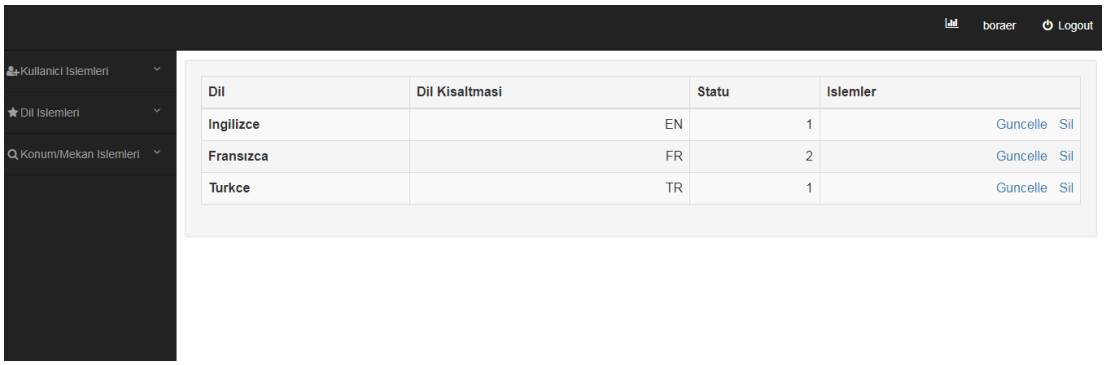
Dil ekleme ekranınından, mekanların detaylarının hangi dillerde olabileceği belirlenir ve bu doğrultuda istenen bir dil eklenir. Şekil 3.13'de yeni dil ekleme ekranı gösterilmiştir.



Şekil 3.13 : Yeni dil ekleme ekranı.

3.3.4 Dil listeleme ekranı

Bu ekran sistemde tanımlanan tüm dillerin listelendiği ekrandır. Bu ekrandan istenen dil üzerinde güncellenme veya silme işlemi yapılabilir. Şekil 3.14'de dil listeleme ekranı gösterilmiştir.



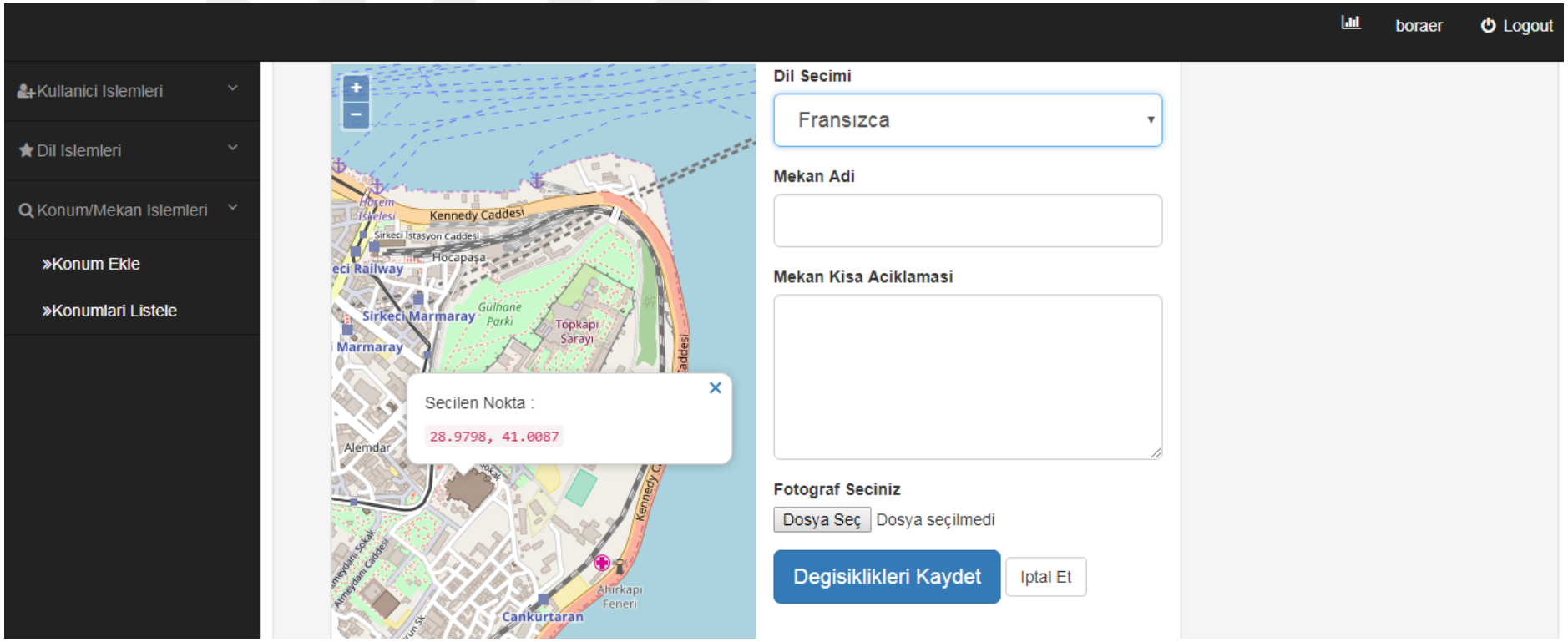
Şekil 3.14 : Dil listeleme ekranı.

3.3.5 Yeni mekan ekleme ekranı

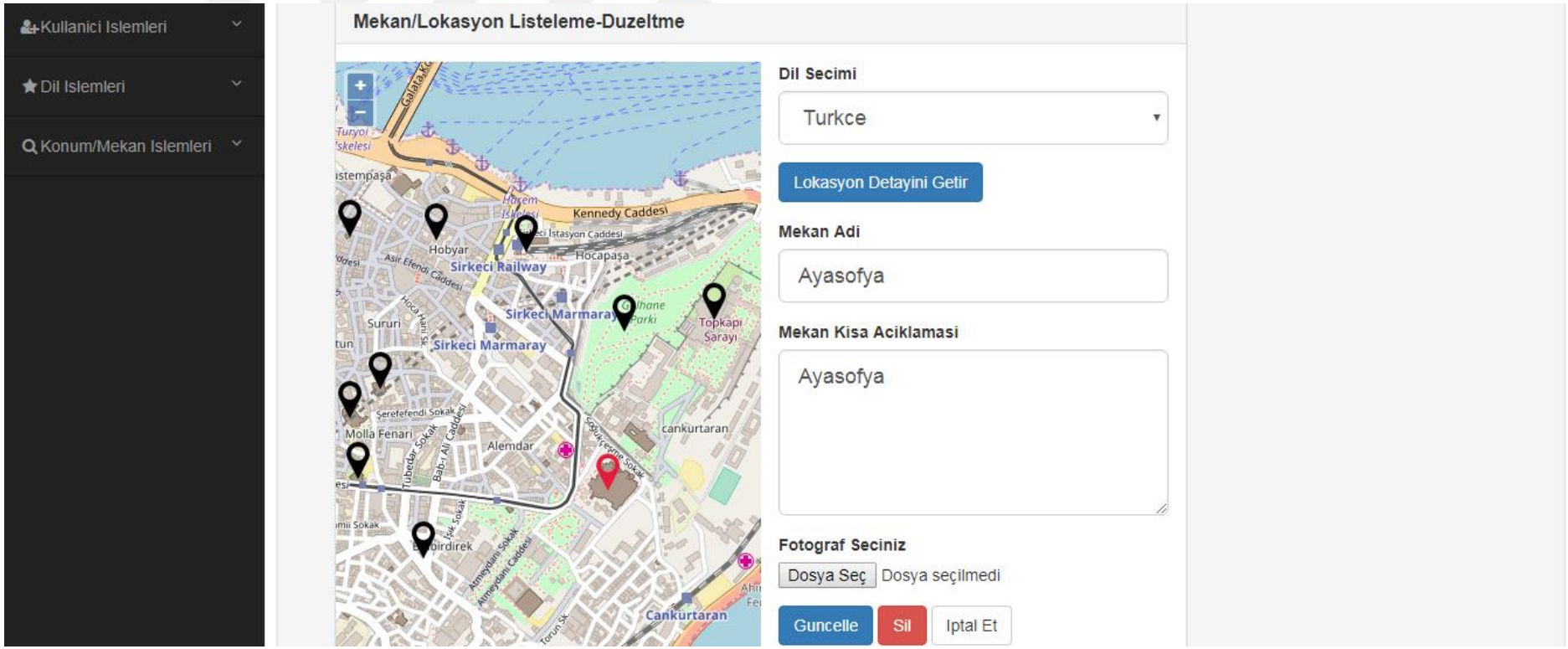
Yeni mekan ekleme ekranında sol bölmede OSM altlığı gelmektedir. Bu altlık kullanılarak eklenmek istenen mekanın koordinat bilgileri seçilir. Bu bilgiler OSM'de açılır menü olarak gösterilir. Sağ bölmedeyse mekanın detayı hangi dilde eklenecekse öncelikli dil olarak seçilir. Bir sonraki adımda mekan detayları girilir eğer isteniyorsa mekanın fotoğrafı eklenir ve mekan kaydedilir. Şekil 3.15'de mekan/konum ekleme ekranı gösterilmiştir.

3.3.6 Mekanları listeleme silme ve güncelleme ekranı

Mekanlar OSM üzerinde listelenir. Bir mekanın üzerine tıklanır. Tıklanan mekan kırmızı renge döner. Dil seçimi yapılıp detay bilgisini getir butonuna basıldığında o mekanla ilgili detay bilgili getirilir. Ön yüze gelen mekan bilgisi güncellenebilir veya detay bilgisi yoksa belirlenen dilde yeni detay bilgisi eklenebilir. Ayrıca mekan ve detaylarını silme işlemleri de silme komutuyla yerine getirilir. Şekil 3.16'da mekan listeleme silme ve güncelleme ekranı gösterilmiştir.



Şekil 3.15 : Yeni mekan/konum ekleme ekranı.



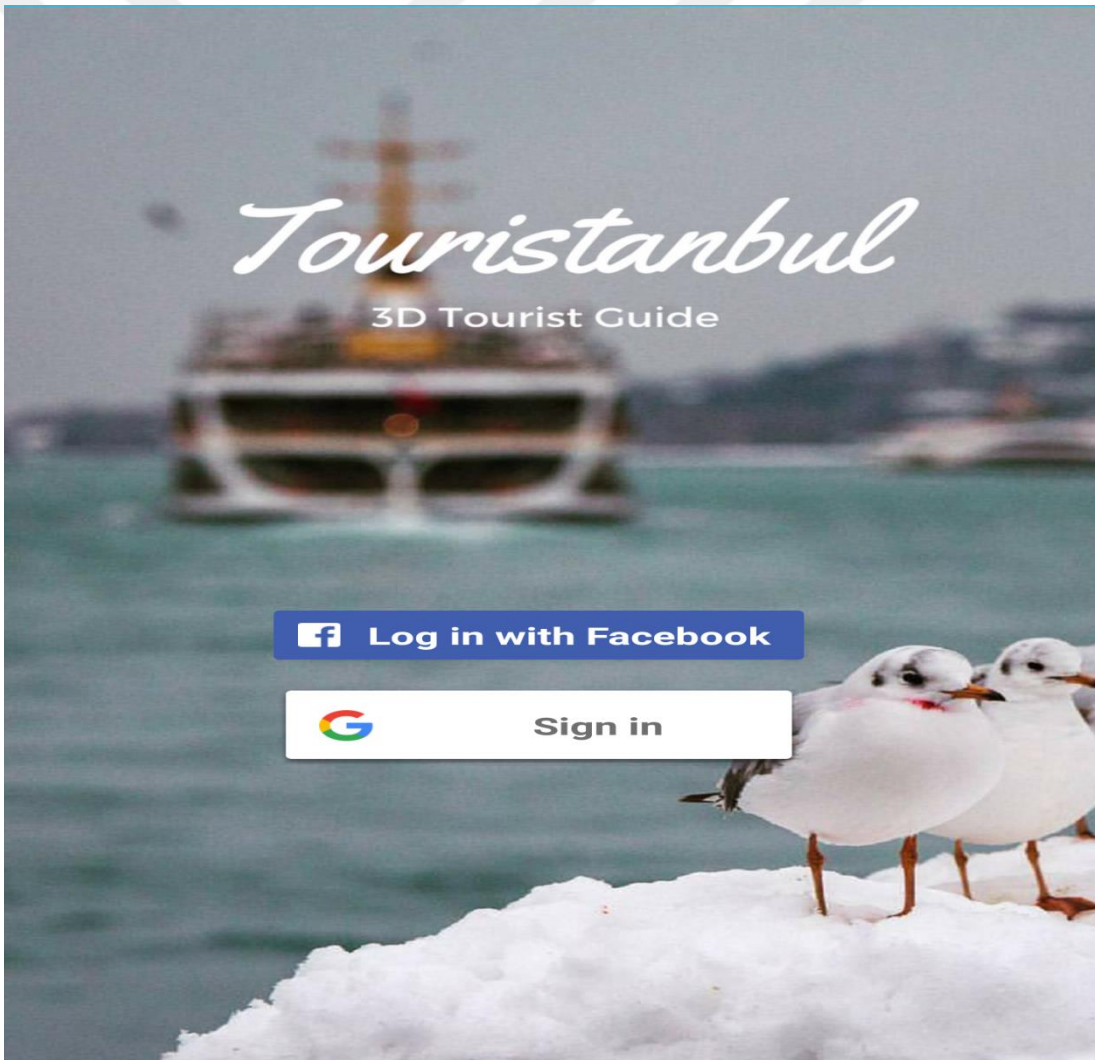
Şekil 3.16 : Mekan listeleme silme ve güncelleme ekranı.

4. TOURISTANBUL PROJESİ MOBİL UYGULAMA YAZILIM GELİŞTİRMESİ VE TESTİ

Bu bölümde SDLC adımları gereğince mobil uygulama geliştime esnasında yapılanlar ve sistemin testinin yapılışı hakkında bilgiler verilecektir.

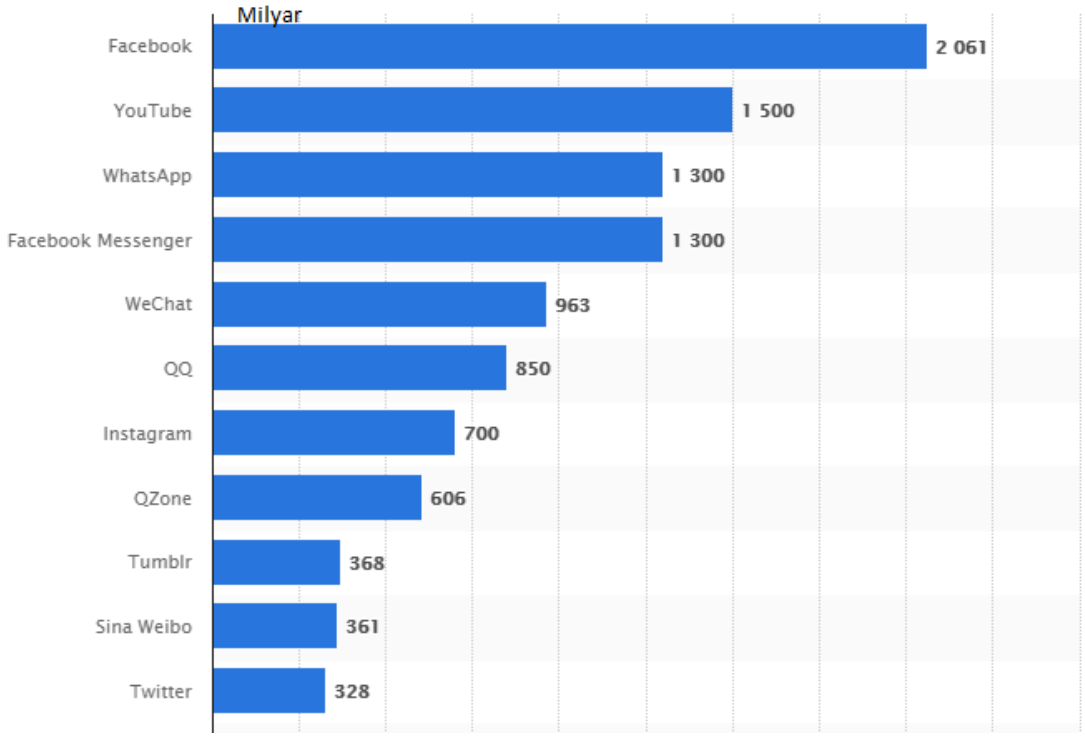
4.1 Giriş Ekranı

Şekil 4.1'de ana giriş ekranı gösterilmiştir.



Şekil 4.1 : Touristanbul giriş sayfası.

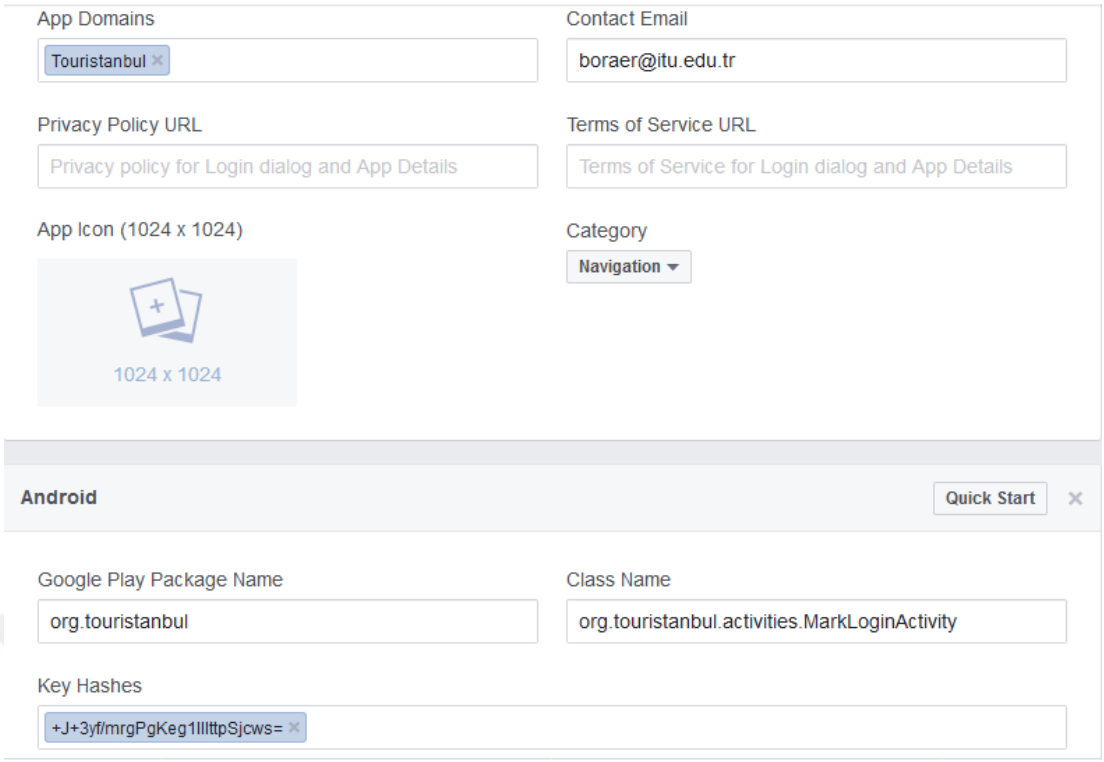
Şekil 4.1'de verilen giriş sayfasında uygulamanın adı ve kullanım amacı hakkında bilgi verildikten sonra, gereksinimlere uygun olarak iki adet sosyal medya girişi butonu konulmuştur. Kullanılacak sosyal medya hesapları, kullanıcıların genel yönelimleri göz önüne alınarak seçilmiştir. 2017 Eylül ayı itibariyle global düzeyde kullanılan sosyal medya uygulamalarının kullanıcı sayıları Şekil 4.2'de gösterilmiştir. Şekil 4.2'ye göre Facebook 2.061 milyar aktif kullanıcıyla birinci sıradadır [31].



Şekil 4.2 : Eylül 2017 itibariyle sosyal medya uygulamalarının kullanıcı sayıları.

Sosyal medya uygulamalarına ilaveten Google arama motorunun ortak çatısı altında topladığı Google hesaplarının kullanıcı sayısı 2017 Mayıs ayı itibariyle iki milyardır [32]. Genel yönelimin yapılan araştırma sonucunda, her kullanıcının bir Facebook veya Gmail hesabının olduğu yönündedir. Sonuç olarak kullanılacak sosyal medya hesapları olarak Facebook ve Google seçilmiştir.

Facebook ve Google giriş butonlarını uygulamaya ekleyebilmek için uygulamanın Facebook ve Google'a kayıt ettirilmesi gerekmektedir. Facebook için uygulama kayıt edilirken uygulama için bir doğrulama şifresi oluşturulur ve bu şifre kaydedilir. Şekil 4.3'de Facebook kaydı gösterilmiştir.



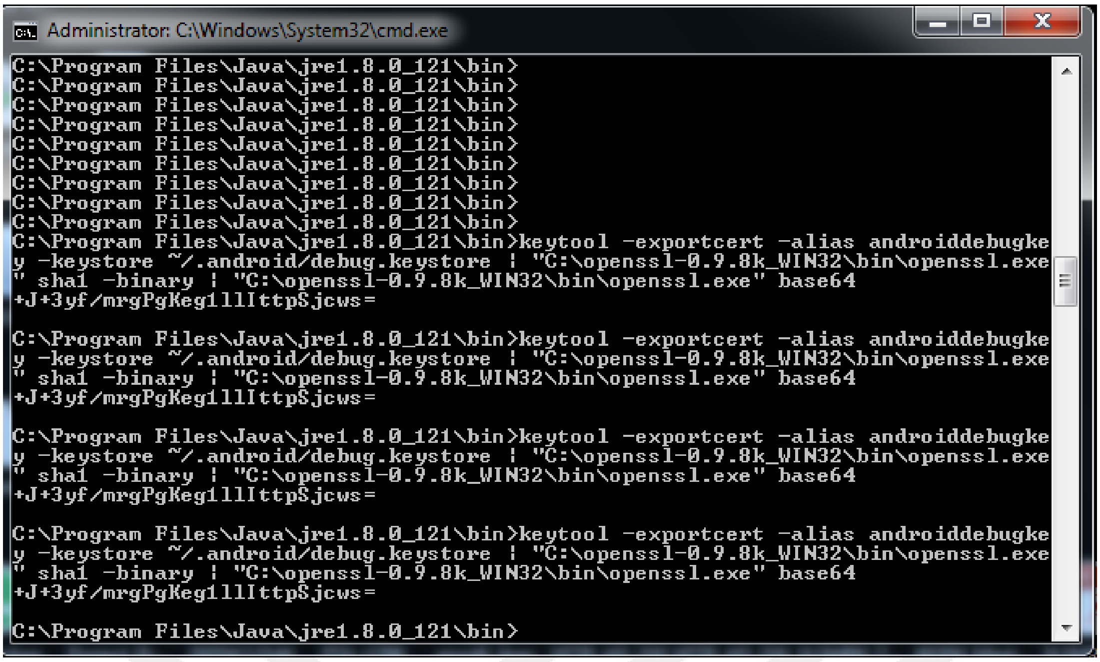
Şekil 4.3 : Facebook uygulama kaydı.

Uygulama kaydı oluşturulurken doğrulama şifresinin düzgün bir şekilde girilmesi önemlidir. Oluşturulan şifre sayesinde uygulama ve Facebook API haberleşmesi esnasında güvenlik sağlanmış olur. Uygulama tarafında bu şifreyi oluşturmak için Şekil 4.4'de verilen komut kullanılır.

```
keytool -exportcert -alias androiddebugkey -keystore  ~/.android/debug.keystore
| openssl sha1 -binary  | openssl base64
```

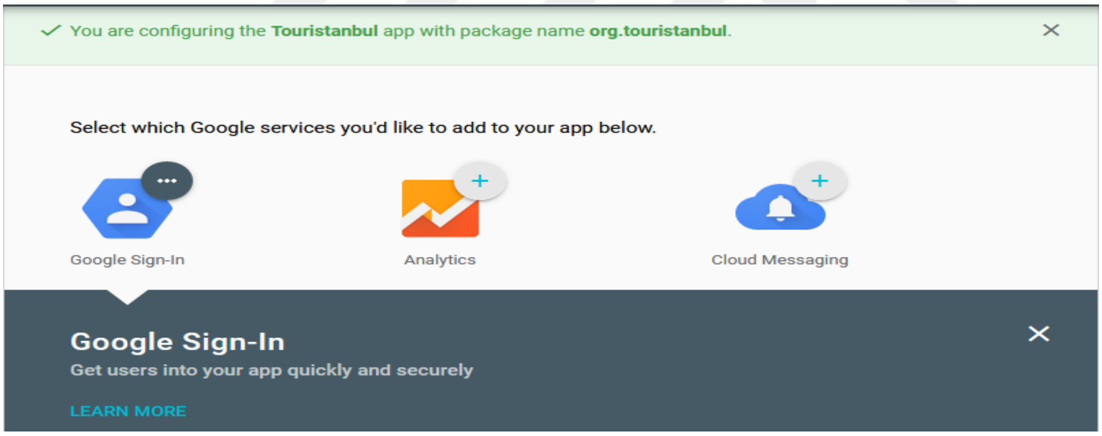
Şekil 4.4 : Şifre oluşturma komutu.

Şekil 4.5'de komut satırına Şekil 4.4'deki komutun yazılıp çalıştırılması sonucu oluşan ve uygulamaya işlenen şifre gösterilmiştir.



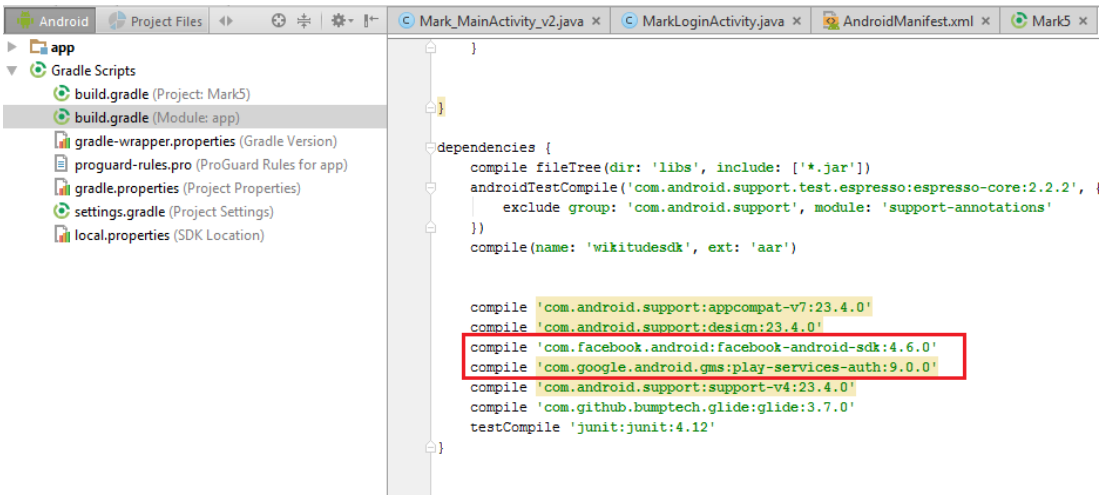
Şekil 4.5 : Şifrenin oluşturulması.

Uygulama Google’a da aynı şekilde kayıt edilir. Şekil 4.4’deki komut kullanılarak SHA1 anahtarı elde edilir. Şekil 4.6’da uygulamanın Google ’a kaydı gösterilmiştir.



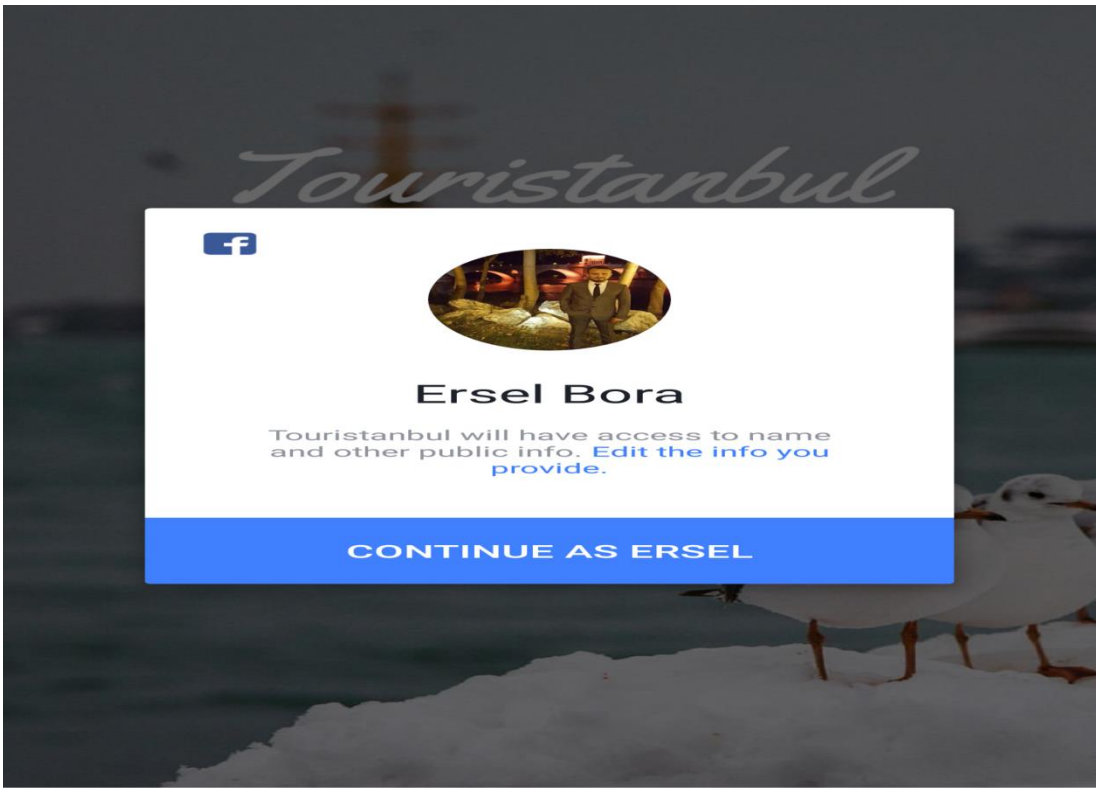
Şekil 4.6 : Uygulamanın Google’a kaydedilmesi.

Uygulamanın kayıt işlemlerinden sonra Facebook ve Google ile haberleşemesi için kodun derlendiği ana dosyaya bu iki dış uygulama bağımlılık olarak eklenmelidir. Bu sayede Facebook ve Google API’lerinin sağladığı kütüphaneler Touristanbul mobil uygulaması için de kullanılabilecektir. Şekil 4.7’de derleme ortamına bağımlılıkların eklenmesi gösterilmiştir.



Şekil 4.7 : Bağımlılıkların derleme ortamına eklenmesi.

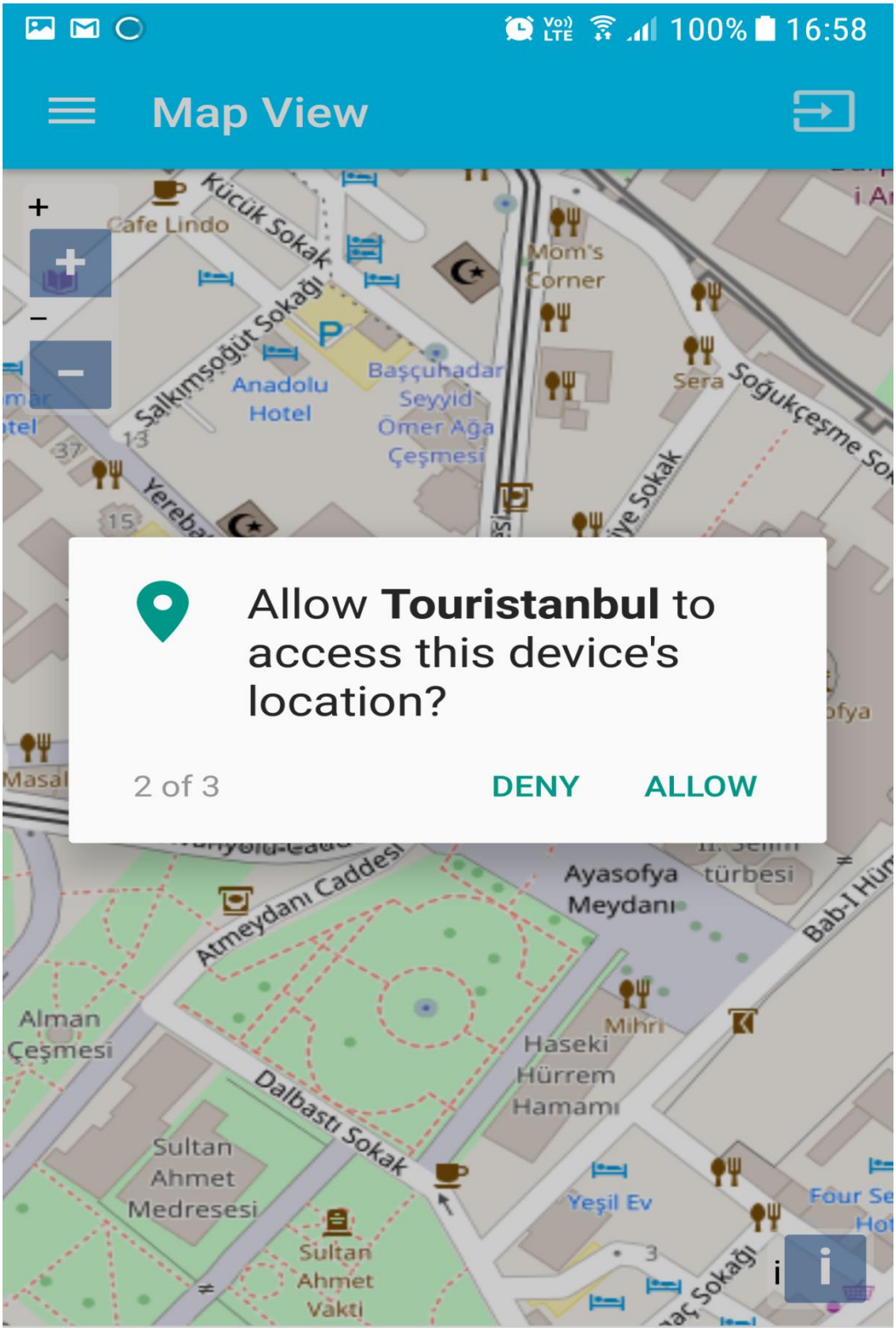
Kullanılan kütüphanelerle Facebook ve Google'dan kullanıcı bilgileri alınabilecek ve kullanıcının uygulamaya giriş yapabilmesi sağlanacaktır. Uygulamaya giriş yapabilmek için Facebook veya Google giriş butonlarından birine bastıktan sonra karşımıza Şekil 4.8'deki gibi bir ekran gelecektir. Bu ekranda kullanıcının hangi bilgilerine ulaşılacağı ve bunu isterse değiştirebileceği bilgisi verilir. Kullanıcı devam et tuşuna bastığında, kullanıcının uygulamaya girişi tamamlanır ve harita ekranına yönlendirilir.



Şekil 4.8 : Giriş geçiş ekranı.

4.2 Harita Ekranı

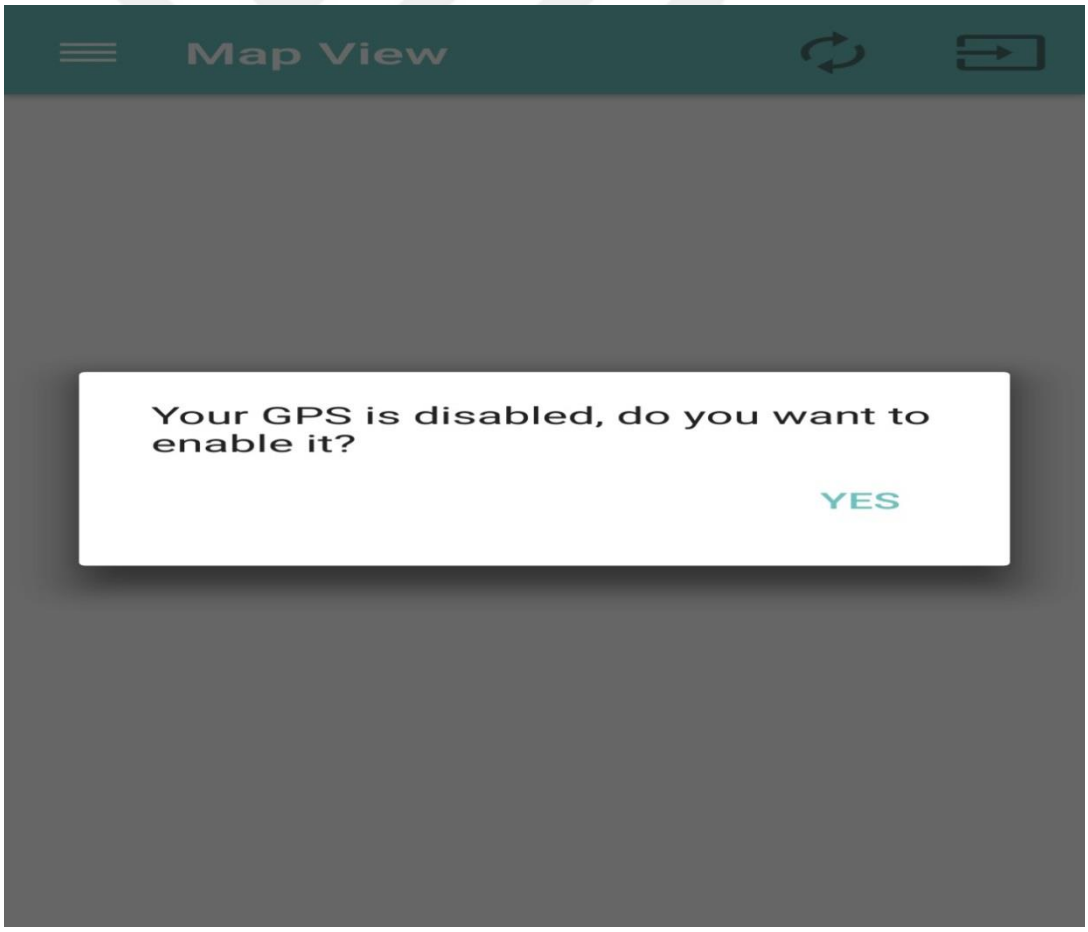
Şekil 4.9'da ilk giriş harita ekranı gösterilmiştir.



Şekil 4.9 : İlk giriş harita ekranı.

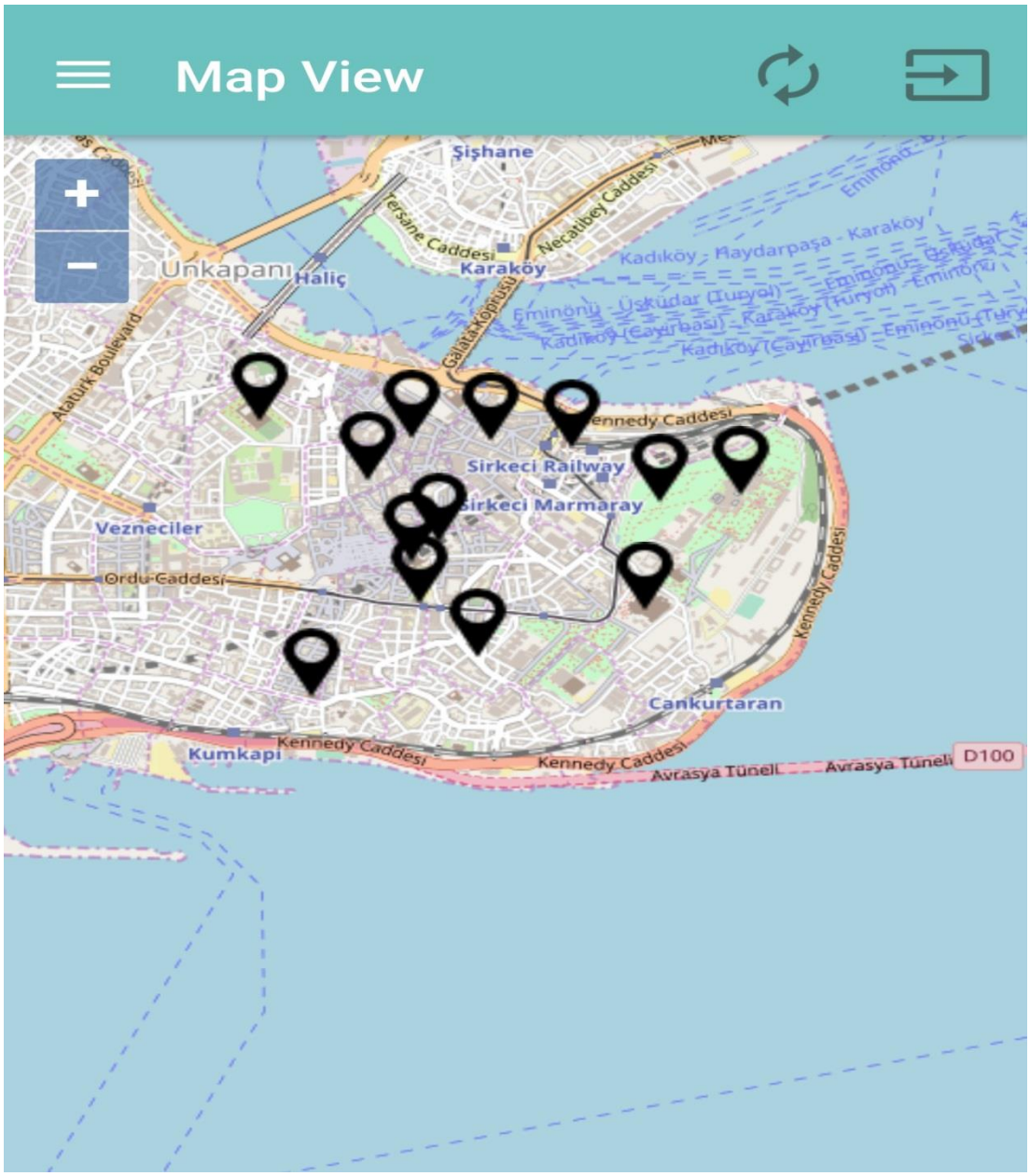
Uygulama kurulduktan sonra ilk giriş yapıldığında, uygulama zorunlu olarak cihazın lokasyon bilgilerine, dosya sistemine ve kamerasına erişim izni istemektedir. Lokasyon bilgilerine erişim sayesinde GPS'den gelen verileri okuyabilme, dosya sistemine erişim sayesinde kullanıcı bilgilerini ve harita bilgilerini cihazın sabit diskine kaydetme, kameraya erişim sayesinde ise artırılmış gerçekliğin kullanılması amacıyla kameranın açılmasını sağlayabilme özelliği kazanır. Eğer bu izinler verilmezse uygulama düzgün bir şekilde çalışamaz. Ayrıca yukarıda beliren uyarı ekranı gelecektir.

Erişim izinleri uygulamaya verildikten sonra uygulama mobil cihazın GPS'nin aktif olup olmadığını kontrol eder. Eğer aktif değilse kullanıcıya GPS'i aktif hale getirmesi için bir uyarı ekranı çıkartır ve konum ayarlarına yönlendirir. Şekil 4.10'da GPS uyarı ekranı gösterilmiştir.



Şekil 4.10 : GPS uyarı ekranı.

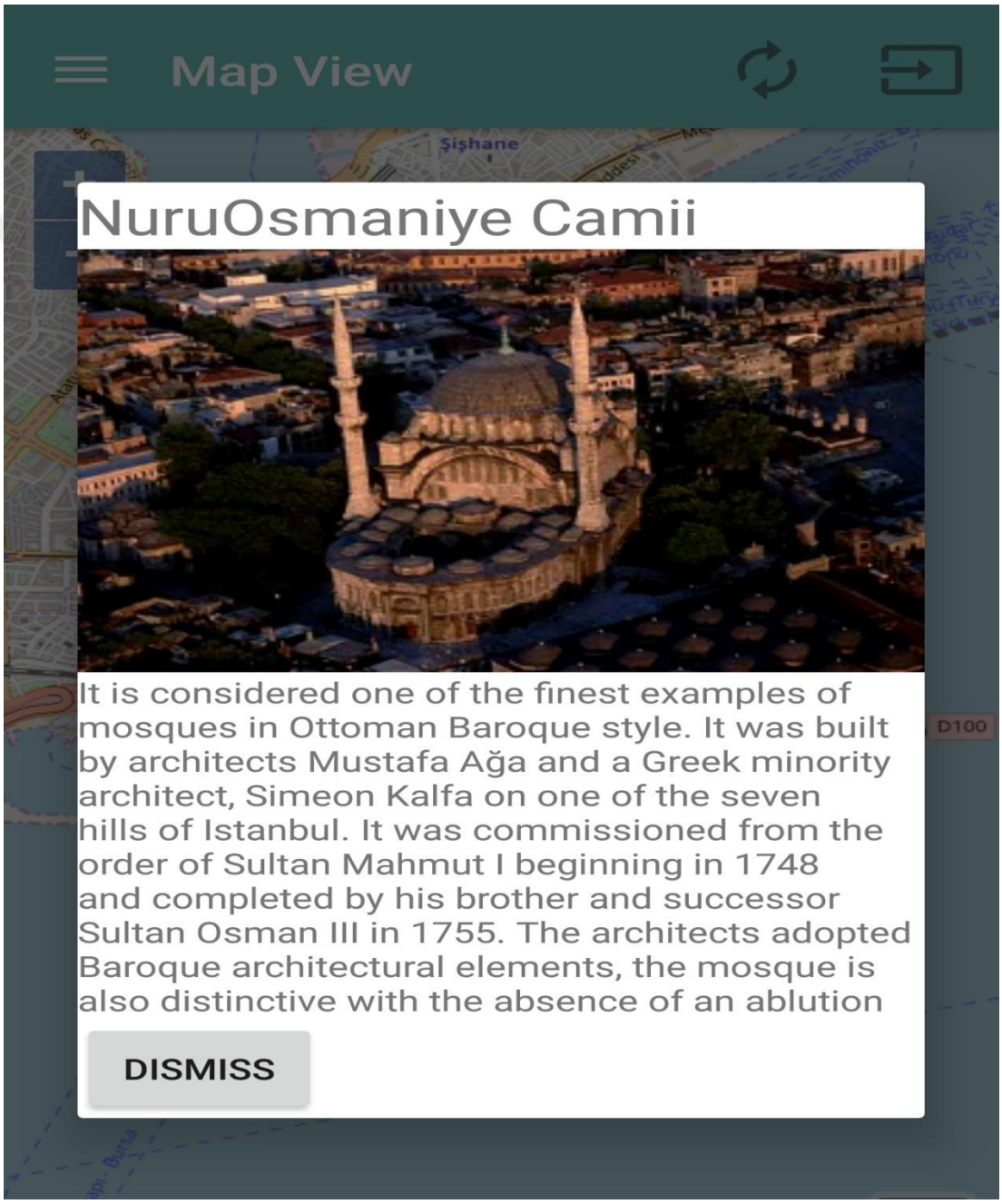
Mobil uygulama, GPS'in aktif hale gelmesiyle konum bilgisini GPS'den okur ve her bir dakikada veya kullanıcı bir önceki konumundan iki yüz elli metre yer değiştirmisse yeni konum bilgisini GPS üzerinden sorgulama işlemini tekrar eder. Uygulama konum bilgisini aldıktan sonra kullanıcının seçebildiği uzaklık parametresiyle birlikte mekankaların verilerini arka uç uygulamanın Restful servisini çağırarak alır. Mekan verileri harita üzerinde Şekil 4.11'de belirtilen şekilde gösterilir.



Şekil 4.11 : Harita ekranında mekanların gösterilmesi.

4.3 Mekan Detayı Ekranı

Harita ekranında veya artırılmış gerçeklik görünüm ekranında bir mekanı gösteren ikonun üzerine tıkladığımızda mobil uygulama bu işlemi algılar ve ilgili mekanın detay bilgisini arka uç uygulamadan Restful servisle sorgular. Arka uç uygulamadan dönen detay bilgilelerini ayrı bir çerçeve içinde gösterir. Şekil 4.12'de mekan detayı ekranı gösterilmiştir.



Şekil 4.12 : Mekan detayı ekranı.

4.4 Artırılmış Gerçeklikle Üç Yüz Altmış Derece Görünüm Ekranı

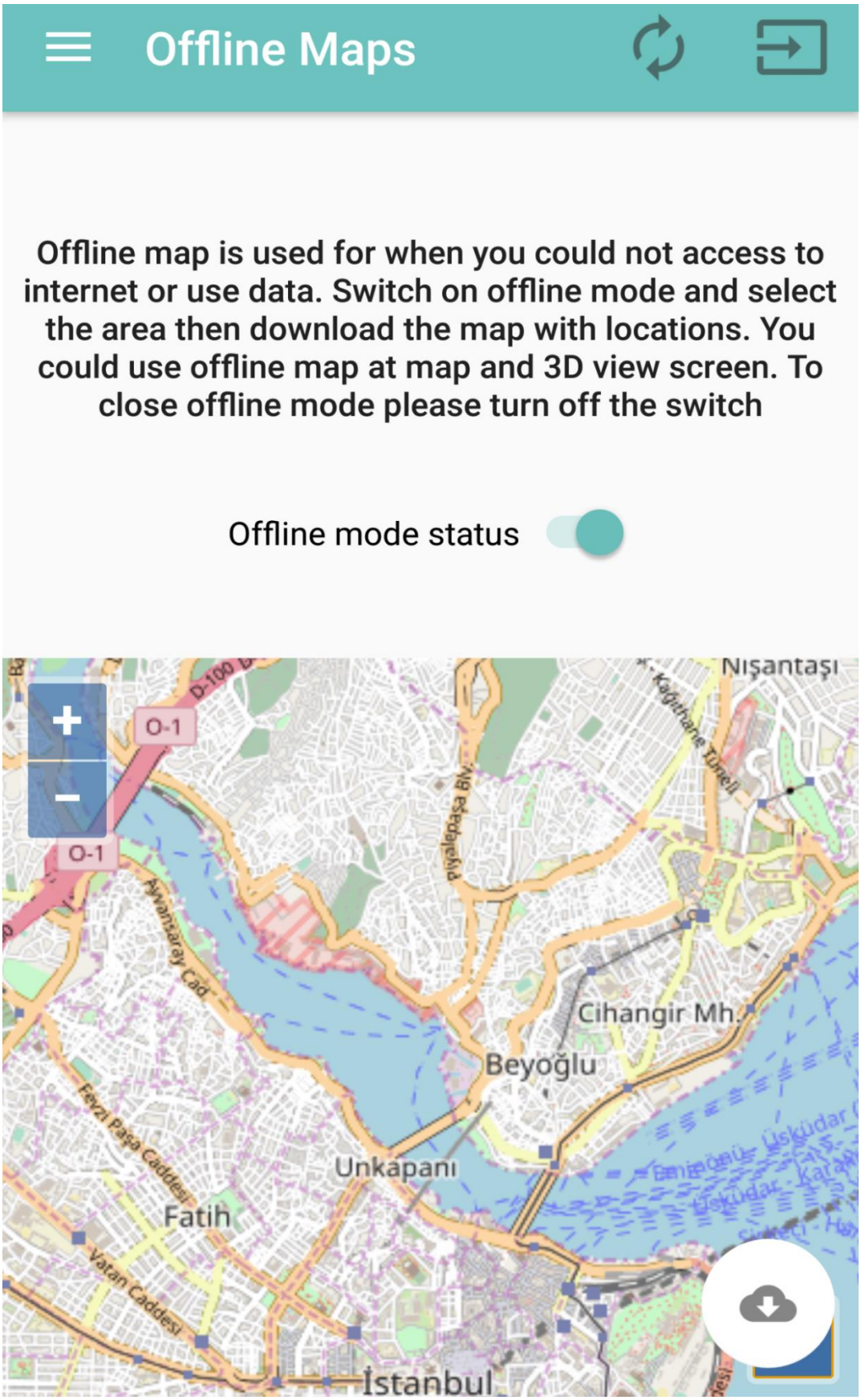
Şekil 4.13'de üç yüz altmış derece görünüm ekranı gösterilmiştir.

Şekil 4.13 : Artırılmış gerçeklikle üç yüz altmış derece görünüm ekranı.

Menüden üç ekranına geçiş yapıldığında uygulama otomatik olarak cihazın kamerasını açacaktır. Kullanıcı, GPS'in son olarak lokasyon aldığı konumundan iki yüz elli metre yer değiştirmediyse mekan verileri olarak önbellekte tutulan ve harita ekranı için kullanılan veriler kullanılır. Ancak kullanıcın lokasyonu iki yüz elli metreden fazla değiştiyse uygulama GPS üzerinden tekrar konum bilgisi alır ve arka uç uygulamadan mekanları sorgular. Mekanları göstermek için lokasyon ikonu kullanılır. Kamera bir yapıya tutulduğunda, kameranın gördüğü yapı veritabanında kayıtlıysa yapının üzerinde bir lokasyon ikonı belirir. Bu ikona tıkladığımızda mekan detayı ekranı açılarak yapıyla ilgili bilgiler ekrana getirilir. Artırılmış gerçeklik ekranının görselliğini arttırmak için pusula da eklenmiştir. Pusula üzerinde çevredeki mekanlar noktalar halinde gösterilir.

4.5 Çevrimdışı Mod ve Harita İndirme Ekranı

Kullanıcı internet erişimi olmayacağı veya kotasından harcamak istemediği durumlarda, uygulamayı çevrimdışı modda kullanabilir. Çevrimdışı moda geçiş yapmak için öncelikle menüden "Çevrimdışı haritalar" sekmesine giriş yapar. Bu ekranda çevrimdışı haritaların kullanımıyla ilgili kısa bir bilgi ve çevrimdışı özelliğini açma butonu bulunur. Kullanıcı çevrimdışı moduna geçiş yaptığında ekranda harita aktif hale gelir. Kullanıcı Ekranın sınırları içinde kalan harita alanını seçerek sağ alt köşede bulunan butonuna basar, belirli alan içindeki harita altlığını ve o bölgede bulunan mekanları mobil cihazının sabit diskine kaydeder. Şekil 4.14'de çevrimdışı modu açma ekranı gösterilmiştir. Belirli bir alan kaydedildikten sonra internete bağlı olmadan harita ekranından ve artırılmış gerçeklikle üç yüz altmış derece görünüm ekranından mekanlar gösterilir. Kullanıcı çevrimdışı modu kapatmak için ekrandaki çevrimdışı özelliğini açma butonunun yönünü sola kaydırmalıdır.



Şekil 4.14 : Çevrimdışı mod ekranı.

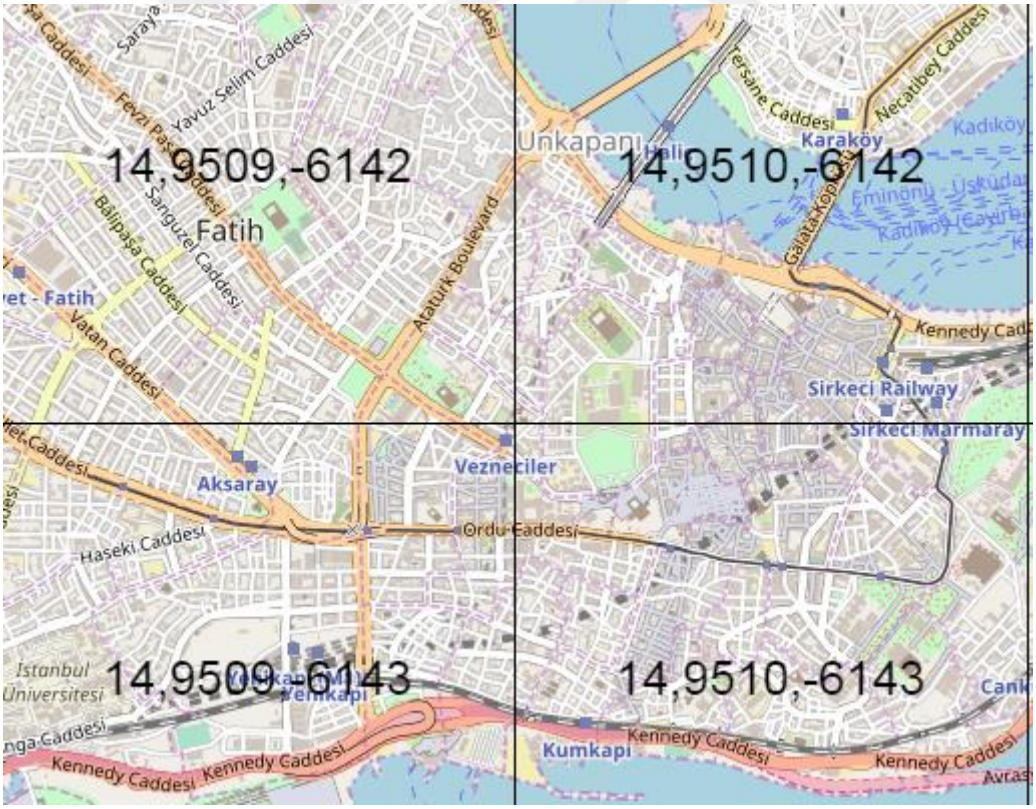
Harita altlık resimlerini kaydedebilmek için kullanıcının seçtiği alanın en üst, en alt enlem bilgileri ile en sol ve en sağ boylam bilgileri OpenLayers yardımıyla elde edilir. Elde edilen bilgilerden X-Y koordinat düzlemine denk gelecek şekilde harita altlıkları denklem 4.1 ve 4.2 ile hesaplanır [33].

n = 2 ^ zoom

xtile = n * ((lon_deg + 180) / 360) (4.1)

ytile = n * (1 - (log(tan(lat_rad) + sec(lat_rad)) / π)) / 2 (4.2)

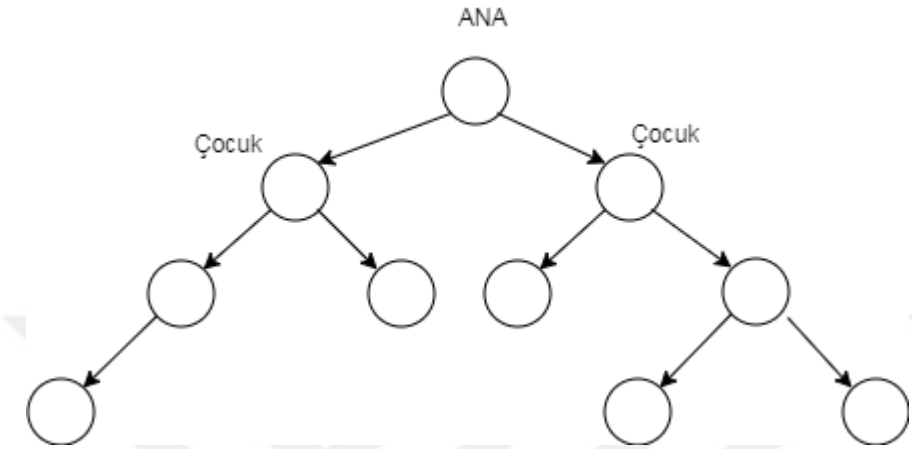
xtile parametresi sabit bir enlem değeri boyunca boylam değerinin değişimini belirtir. ytile parametresi sabit bir boylam boyunca enlem değerinin değişimini belirtir. Sol uç ve sağ uç boylarımlarıyla soldan sağa kaç tane harita altığı resmi, Üst uç ve alt uç enlemleriyle yukardan aşağıya kaç tane harita altlığı gerektiği hesaplanır. Şekil 4.15'de OSM üzerindeki harita altlıklarının hesaplanmış halleri belirli bir yakınlaştırma seviyesinde Tarihi yarımada için gösterilmiştir.



Şekil 4.15 : Harita altlığı X Y düzlemi.

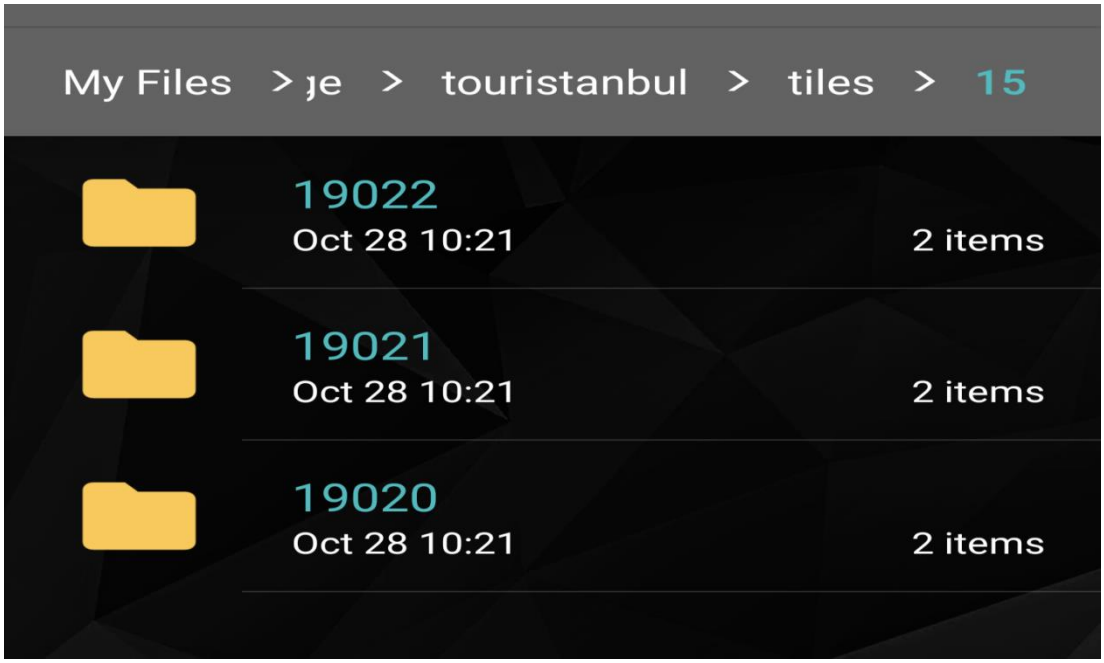
Şekil 4.16'ya göre on dördüncü yakınlaştırma seviyesi için xtile 9509 ve 9510'da bulunan 6142 ve 6143 numaralı ytile'ları indirilmelidir.

Harita altlıkları dosyalara kaydedilirken ağaç veriyapısında kaydedilir. Ağaç veri yapısı objeler arasında hiyerarşik bir yapı kurulmasını sağlar. Bu hiyerarşik yapıda düğüm "kök" veya "ana" olarak adlandırılır. Ana düğümün altında çoçuk düğümler katmanlar biçiminde yer alırlar [34]. Şekil 4.16'da örnek bir ağaç veriyapısı gösterilmiştir.



Şekil 4.16 : Ağaç veri yapısı.

Yakınlaştırma seviyesi kök, yakınlaştırma seviyesinin altında Xtile klasörleri ve her xtile klasörünün içinde Ytile harita altlığı resimleri yer alır. Haritaların bu mantık yapısında kaydedilmesinin sebebi, OSM'de harita altlıkları kullanılacağı zaman OSM'in çevrimdışı haritaları yakınlaştırma seviyesi/xtile/ytile olacak şekilde yüklemesidir. Şekil 4.17'de harita altlıklarının dosya yapısı gösterilmiştir.



Şekil 4.17 : Harita altlıkları dosya yapısı.

Şekil 4.17’de sağ üst taraftaki on beş sayısı yakınlaştırma seviyesini dosyalar ise harita altlıklarının numaralarını belirtir. Her dosyanın içinde belirli alanı kaplayacak şekilde harita altlığı resmi bulunur. Kaç tane resim olacağının sayısı yakınlaştırma seviyesine ve harita alanının büyüklüğüne göre değişiklik göstermektedir.

5. SONUÇLAR VE GELECEKTE YAPILABİLECEK ÇALIŞMALAR

5.1 Sonuçlar

Touristanbul projesini arka uç ve mobil uygulama olarak değerlendirğimizde: arka uç uygulamanın konumsal bir veritabanına sahip olması sayesinde veri yönetimini üzerinde toplamıştır. Konumsal verilerin yönetilmesi ilerleyen dönemlerde İstanbul'da yeni bölgelerin, İstanbul dışında yeni şehirlerin ve Türkiye dışında farklı şehirler eklenebilmesini ve projenin global olarak kullanılabilmesini sağlayacaktır. Açık kaynak olan OSM kullanılarak Google Maps, Yandex Maps gibi harita sağlayıcılara olan bağımlılığın kalkması sayesinde belirli sayıda konumsal sorgu sonrası ücret ödeme zorunluluğunu ve belirli bir platforma bağlı kalmayı ortadan kaldırmıştır. Mekan verilerinin REST servisler kullanılarak sunulması sayesinde farklı mobil uygulamalara ve IOS işletim sistemi için yazılabilecek Touristanbul uygulamasına veri sağlanacabilecektir.

Mobil uygulama tez kapsamında sadece Android işletim sistemine ait cihazlar için tasarlanmıştır. Kullanıcılar, en yaygın iki sosyal medya uygulamasının sağladığı API'ler aracılığıyla kayıt olma işlemleriyle uğraşmadan uygulamayı kullanabilirler. Mekanları iki boyutta ve kamera yardımıyla artırılmış gerçeklik kullanılarak üç yüz altmış derecede görüntüleyebilirler ve detayları hakkında bilgi edinebilirler. Tüm bu özelliklere ek olarak, çevrimdışı mod sayesinde mekanları internete erişimi olmadan görüntüleyebilirler ve bu özellik yabancı turistler için büyük avantaj sağlayacaktır.

5.2 Gelecekte Yapılacak Çalışmalar

Proje konusu belirlenip uygulama üzerinde çalışılmaya başlandığında mevcut açık kaynaklı AR modüllerinin dökümantasyonu ve kullanım kolaylıkları istenen seviyede değildi. Buna ilaveten belirtilen AR modülleri lokasyon bazlı çalışmamaktaydı. Geçen süre sonunda tekrar yapılan incelemelerde açık kaynaklı ARToolkit API'sinin dökümantasyonunun iyileştirildiği ve kullanımının

kolaylaştırıldığı görülmüştür. Ancak halen lokasyon bazlı AR'ı desteklememektedir. Lokasyon özelliğinin mevcut modüle eklenmesi üstünde çalışma yapılabilir.

Projenin ilerleyen fazlarında Emniyet Genel Müdürlüğü'yle veya yaygın olarak kullanılan otel ve turizm uygulamalarının sahibi şirketlerle ortaklaşa çalışılarak alınan geri bildirimlerden oluşturulacak bir suç haritası ile turistler için güvenli yolları belirleyen ve buna göre rota oluşturan bir modül üzerinde durulmaktadır. Projenin bankacılık, emlak, turizm endüstrilerine hizmet vermesi sağlanabilir. Örnek vermek gerekirse, internet emlak siteleri için arka uç uygulamayı kullanarak satılık veya kiralık emlaklara ait mekan verileri oluşturulup bu altyapıyı kullanan ve çevredeki satılık/kiralık durumdaki emlakları kamera yardımıyla artırılmış gerçeklikle işaretleyip gösteren bir mobil uygulama hazırlanabilir. İkinci örnek olarak, bankacılık sektörü için bir uygulama hazırlanabilir. Bu uygulama mevcut arka uç uygulamayı kullanarak bankaya özel indirimlerin bulunduğu mağazaları gösterebilir ve müşterileri bu mağazalara yönlendirebilir.

KAYNAKLAR

[1] **A brief computer history** (t.y.). Erişim tarihi 29 Mart 2017, http://people.bu. edu /baws/brief%20computer%20history.html

[2] **Goasduff, L.,Forni, A.**(2017). Fierce Battle Between Apple and Samsung to Hold the No.1 Global Smartphone Ranking.*Gartner*. Erişim tarihi Nisan 2 ,2017, https://www.gartner.com/newsroom/id/3609817

[3] ***Worldwide mobile app revenues in 2015, 2016 and 2020*** *(2007). Erişim tarihi Nisan 2, 2017, https://www.statista.com/statistics/269025/worldwide-mobile-app-revenue-forecast/*

[4] ***Most popular Apple App Store categories in October 2017, by share of available apps*** *(2007). Erişim tarihi Ekim 15, 2017, https://www.statista.com /statistics/270291/popular-categories-in-the-app-store/*

[5] **Smirnov, A., Kashevnik, A., & Shilov, N., & Teslya, N., & Shabaev,A.** (t.y.). Mobile Application for Guiding Tourist Activities: Tourist Assistant – TAIS, *Proceedıng of the 16th Conference of Fruct Assocıatıon*.

[6] **Carmigniani J., & Furth, B.** (2011).Augmented Reality: An Overview. Furth B.(Editör), *Handbook of Augmented Reality* (ss.3-7). New York:Springer

[7] **Katiyar, A., Kalra, K., Garg, C.** (2015). Marker Based Augmented Reality. *Advances in Computer Science and Information Technology,5(2)*.441-445

[8] **Java(programming language)**.(t.y.). *Encyclopædia Britannica*. Erişim tarihi Nisan 6,2017, https://www.britannica.com/technology/Java-computer-programming-language

[9] **Phillips, B., & Hardy, B.** (2013). *Android Programming The Big Nerd Ranch Guide*.Indianapolis.: Pearson Technology Group.

[10] **Abiteboul, S., Hull, R., & Vianu, V.** (1995). Database Systems. *Foundations of Databases(ss. 3)*. USA. : Addison-Wesley

[11] **Wagner, G.** (2012). Conceptual Modeling of Knowledge Bases. *Foundations of Knowledge Systems: with Applications to Databases and Agents*(ss.3-17)

[12] **Mock, L.** (2016). Entity Relationship Diagrams: Symbols & Meaning.*Gliffy*. Erişim tarihi Nisan 15, 2017, https://www.gliffy.com/blog/entity-relationship-diagrams-symbols-meaning

[13] **Kreger,H.** (2001). Web Services Conceptual Architecture (WSCA 1.0),*ResearchGate.* Erişimi tarihi Nisan 15, 2017, https://www.researchgate.net/publication/235720479_Web_Services_Conceptual_Architecture_WSCA_10

[14] **JSON'a Giriş.** (t.y.), Erişim tarihi 16 Nisan 2017, http://www.json.org/json-tr.html

[15] ***Understanding REST***. *(2017). Erişim tarihi 17 Nisan 2017, https://spring.io /understanding/REST*

[16] ***RESTful Web Services Tutorial***. *(2006). Erişim tarihi 15 Ekim 2017,https:// www.tutorialspoint.com/restful/*

[17] **API - application program interface**, (t.y.).*Webopedia.*Erişim tarihi Nisan 17, 2017, https://www.webopedia.com/TERM/A/API.html

[18] ***What is the Software Development Life Cycle (SDLC)***. *(2013).Erişim tarihi 20 Nisan 2017, https://airbrake.io/blog/sdlc/what-is-the-software-development-life-cycle*

[19] **Sinnot, R. W.** (1984). Virtues of the Haversine. *Sky and Telescope, vol. 68(2),* 159.

[20] **Vicenty, T.** (1975). Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations (Rapor No. 176). Wyoming : Survey Review Vol. XXIII.

[21] ***Spring Framework***. *(2017). Erişim tarihi 21 Nisan 2017,https://projects.spring. io/spring-framework/*

[22] ***Optimizing Spatial Query***. *(1995). Erişim tarihi 21 Nisan 2017,https://dev. mysql.com/doc/refman/5.7/en/optimizing-spatial- analysis.html/*

[23] ***Augmented Reality SDK Comparison***.*(2010). Erişim tarihi 24 Nisan 2017, http://socialcompare.com/en/comparison/augmented-reality-sdks*

[24] ***Introduction to the Wikitude SDK***. *(2009). Erişim tarihi 25 Nisan 2017, https: //www.wikitude.com/external/doc/documentation/latest/android/*

[25] ***13 JavaScript Libraries to Create Interactive & Customized Maps***. *(2017).Erişim tarihi 30 Nisan 2017, http://www.hongkiat.com/blog/ javascript-libraries-for-interactive-maps/*

[26] ***Layar***,*(2009). Erişim tarihi 25 Nisan 2017, https://www.layar.com/*

[27] ***Tuscany+: the first Augmented Reality tourism application***.*(2010). Erişim tarihi 1 Mayıs 2017, https://www.visittuscany.com/en/ideas/4aa26c34-5119-11e7-87d1-c25849f3fc0c/*

[28] ***Spatial Relation Functions That Use Minimum Bounding Rectangles***.*(2017). Erişim tarihi 11 Eylül 2017,https://dev.mysql.com/doc/refman/ 5.7/en/spatial-relation-functions-mbr.html#function_mbrcontains*

[29] **Thymeleaf**.(t.y.). Erişim tarihi 10 Ekim 2017, http://www.thymeleaf.org/

[30] ***Bootstrap***.*(2011).Erişim tarihi 10 Ekim 2017, http://getbootstrap.com/*

[31] ***Most famous social network sites worldwide as of September 2017***.*(2007). Erişim tarihi 12 Ekim 2017, https://www.statista.com/statistics/ 272014/global-social-networks-ranked-by-number-of-users/*

[32] **Popper, B.** (2017, Mayıs 17). Google announces over 2 billion monthly active devices on Android. *The Verge.* https://www.theverge.com/2017/5/ 17/15654454/android-reaches-2-billion-monthly-active-users

[33] ***Slippy map tilenames****.(2006).Erişim tarihi 5 Ağustos 2017, http://wiki .openstreetmap.org/wiki/Slippy_map_tilenames*

[34] **Shaffer, C. A.** (2011). Binary Trees, *Data Structures and Algorithm Analysis in Java,* (3. Baskı, ss 145-188). New York : Dover Publications.

ÖZGEÇMİŞ

Ad-Soyad **:** Ersel BORA

Doğum Tarihi ve Yeri **:** 18.07.1988 Ankara

E-posta **:** boraer17@gmail.com

ÖĞRENİM DURUMU:

- **Lisans** **:** 2012, İstanbul Teknik Üniversitesi, Elektrik Elektronik Fakültesi,Bilgisayar Mühendisliği

MESLEKİ DENEYİMLER

- Mayıs 2017- Kafein Danışmanlık, Türktelekom OIM projesi, Java yazılım danışmanı
- Temmız 2015-Mayıs 2017 Kafein Danışmanlık, Vodafone Faturalama ve Tahsilat sistemleri projesi, Java-PLSQL yazılım danışmanı
- Eylül 2014-Temmuz 2015 Kafein Danışmanlık, AVEA Anka projesi, Java yazılım uzmanı
- Mart 2012-Eylül 2014, Kafein Danışmanlık ,Vodafone Faturalama ve Tahsilat sistemleri projesi, Java-PLSQL yazılım uzmanı
- Temmuz 2011-Mart 2012 Vodafone Türkiye, Vodafone Faturalama ve Tahsilat sistemleri projesi, Java-PLSQL yazılım uzmanı