İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

ANDROID İŞLETİM SİSTEMLİ MOBİL CİHAZLARIN ANLIK KONUM BİLGİSİ KULLANILARAK PAFTA BUL UYGULAMASININ GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Salih YALÇIN

Bilişim Uygulamaları Anabilim Dalı

Coğrafi Bilgi Teknolojileri Programı

AĞUSTOS 2017

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ

ANDROID İŞLETİM SİSTEMLİ MOBİL CİHAZLARIN ANLIK KONUM BİLGİSİ KULLANILARAK PAFTA BUL UYGULAMASININ GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Salih YALÇIN
(706131020)

Bilişim Uygulamaları Anabilim Dalı

Coğrafi Bilgi Teknolojileri Programı

AĞUSTOS 2017

İTÜ, Bilişim Enstitüsü'nün 706131020 numaralı Yüksek Lisans Öğrencisi **Salih YALÇIN'ın**, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı **"ANDROID İŞLETİM SİSTEMLİ MOBİL CİHAZLARIN ANLIK KONUM BİLGİSİ KULLANILARAK PAFTA BUL UYGULAMASININ GELİŞTİRİLMESİ"** başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı :	**Doç. Dr. Himmet KARAMAN** İstanbul Teknik Üniversitesi	.....
Jüri Üyeleri :	**Doç. Dr. Turan ERDEN** İstanbul Teknik Üniversitesi	.....
	Yrd. Doç. Dr. Hüseyin Can ÜNEN Maltepe Üniversitesi	.....

Teslim Tarihi : **5 Mayıs 2017**
Savunma Tarihi : **9 Haziran 2017**

Nişanlım'a

ÖNSÖZ

Bu çalışmada akıllı cihazların GPS özelliği kullanarak, kullanıcıların konumundan anlık pafta bilgisini gösteren ve koordinat dönüşümleri yapan bir mobil uygulama geliştirilmiştir.

Projenin başlangıcından sonuna kadar her türlü desteğini esirgemeyen tez hocam Doç. Dr. Himmet Karaman'a;

Uygulamanın geliştirilmesi sürecinde bana sonsuz katkıda bulunan İstanbul Büyükşehir Belediyesi Coğrafi Bilgi Sistemi Müdürlüğü Mobil Yazılım Geliştirme Şefliği'nde bulunan değerli iş arkadaşlarıma;

Fikirleriyle, çalışmalarıyla ve yaptığı katkılarla hem üniversite hayatıma hem de iş hayatıma sayısız katkıda bulunan en yakın arkadaşım, iş arkadaşım Furkan Özoğlu'na

Beni bugünlere getiren ve karşılıksız desteklerini esirgemeyen aileme;

Teşekkür ederim.

Haziran 2017

Salih YALÇIN
(Harita Mühendisi)

İÇİNDEKİLER

Sayfa

KISALTMALAR

API **:** Application Programming Interface
ESRI **:** Enivromental Systems Research Institute
GPS **:** Global Positioning System
iOS **:** iPhone/iPad Operation System
JVM **:** Java Virtual Machine
SDK **:** Software Development Kit
UTM **:** Universal Transversal Mercator
WiFi **:** Wireless Fidelity
WGS **:** World Geodetic System

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

ANDROID İŞLETİM SİSTEMLİ MOBİL CİHAZLARIN ANLIK KONUM BİLGİSİ KULLANILARAK PAFTA BUL UYGULAMASININ GELİŞTİRİLMESİ

ÖZET

Gelişen teknolojiyle birlikte akıllı telefonlar hayatımızda daha fazla yer almaktadır. Kısıtlı özelliklere sahip telefonlar son yıllarda yerlerini daha iyi özellikli akıllı telefonlara bırakmaktadır. Akıllı telefonlar bünyelerinde GPS, WiFi, dokunulabilir ekran, kamera gibi özellikleri bulundurmaktadır. Bu özellikler sayesinde kullanıcılar kendilerine yardımcı birçok işlemi akıllı telefon üzerinden gerçekleştirilmektedir. Konum bulma özelliği ise akıllı telefonların en çok kullanılan özelliklerinin başında gelmektedir. Bu özellik ile navigasyon, acil yardım, lojistik vs. gibi alanlara katkı sağlanmaktadır.

Akıllı telefonlarda Android, iOS ve Windows Phone işletim sistemleri bulunmaktadır. Android Google tarafından Linux tabanlı olarak geliştirilmiş açık kaynak kodlu mobil işletim sistemidir. iOS, Apple tarafından geliştirilmiş ve iPhone, iPad gibi cihazlarda çalışan işletim sistemidir. Windows Phone ise Microsoft tarafından geliştirilmiştir. Tüm bu işletim sistemlerinin kendilerine has uygulamalarını yayınladıkları marketleri bulunmaktadır. Android için Google Play, iOS için AppStore ve Windows için ise Microsoft Store. Kullanıcılar telefonlarında kullandıkları işletim sistemine göre ihtiyaçları olan uygulamaları indirmektedir.

Konum bulma özelliği akıllı telefonlarda bulunan en önemli özelliklerden biridir. Üç farklı yöntem ya da bu yöntemlerin birleşimi ile doğru konum elde edilebilir. Bunlar; hücresel ağlar, WiFi ve telefonda bulunan GPS'tir. Hücresel ağlar ve Wifi, etrafta bulunan baz istasyonlarının konumunun kullanarak kestirme yöntemi yardımıyla konum belirlenmesine katkı sağlamaktadır. GPS vasıtasıyla belirlenen konumda ise GPS uyduları önemli bir rol oynamaktadır.

Bu tez çalışması kapsamında kullanıcının anlık konumu kullanılarak, yer tespiti yapılmıştır. Yapılan yer tespitinin anlamlı bilgiye dönüştürülmesi kapsamında ulusal pafta bölümlemesi sisteminde hangi dilim numarasına denk geldiği, dilim orta meridyeni bilgisi, 25 binlik, 50 binlik ve 100 binlik pafta bilgisinin gösterildiği, bu pafta bilgilerinin harita üzerinde çizdirildiği, harita üzerinde istenilen bir noktaya ait pafta bilgilerinin gösterilebildiği ve bu pafta bilgilerinin kullanıcı bazında favorilere

eklenebildiği, favorilere eklenen paftalar ile ilgili notlar alınabilen WGS84 sisteminde koordinat bilgisinin gösterildiği, belirlenen koordinat bilgisi ile UTM ve Gauss-Kruger sistemleri arasında dönüşüm yapılabilen Android işletim sistemli cihazlarda çalışan bir mobil uygulama geliştirilmesi amaçlanmıştır.

Uygulama geliştirilirken, Google tarafından yayınlanan Materyal tasarım ilkeleri göz önünde bulundurularak, kullanım kolaylığı ve kullanıcı deneyimi açısından zengin bir tasarım hazırlanmıştır. Hazırlanan bu tasarımların prototipleri oluşturulmuş ve uygulama geliştirilmeden önce kullanıcı deneyim ilkeleri göz önünde bulundurularak testler yapılmıştır. Yapılan testler sonucunda kullanıcı deneyimi açısından sorun teşkil edilebilecek yerler tekrardan tasarlanmıştır.

Tasarım aşamasının tamamlanması ile birlikte geliştirme aşamasına geçilmiştir. Uygulamanın geliştirme aşamasında Android Studio geliştirme ortamında Java dili kullanılmıştır. Harita tabanlı işlemler için Google Maps Android API kullanılmıştır. Geliştirme prensibi olarak nesneye dayalı programlama belirlenmiştir.

Çalışmanın sonucunda geliştirilen uygulamala ile kullanıcılar Android platformunda anlık konumlarından pafta bilgilerini ve koordinat bilgilerini kolaylıkla öğrenebilmiş koordinat dönüşümlerini hızlı bir şekilde yapabilmiştir.

DEVELOPMENT OF MAP SECTION FINDING APPLICATIONS BY USING INSTANT LOCATION INFORMATION OF ANDROID DEVICES

SUMMARY

Along with developing technology, smartphones are taking more place in our lives. In recent years, phones with limited features have left their place to better-equipped smartphones. Smart phones have features such as GPS, WiFi, touch screen, camera thanks to these features, many operations are carried out on the smartphone to assist the users themselves. Location detection is one of the most used features of smartphones. With this feature location information could contirbute areas like navigation, emergency assistance, logistics etc.

Smartphones have Android, iOS and Windows Phone operating systems. Android is an open source mobile operating system developed by Google based on Linux. IOS is an operating system developed by Apple and running on devices such as iPhone and iPad. Windows Phone is developed by Microsoft. All these operating systems have their own markets where they publish their own applications. Google Play for Android, AppStore for iOS and Microsoft Store for Windows. Users download the applications that they need on their phones according to the operating system they use.

Android operational system is Linux based open source system that developed by Google. Android OS is distributed for free of charge. Astro, the first version of the Android operating system, was launched on 9 February 2009. The latest version still on the launch is the 8.0 version with the Oreo name.

The programming language used while developing Android application is the Java programming language. The Java programming language is also an open source programming language, as well as the Android operating system. Android apps are not only developed with Java, but are also developed with the Kotlin. Kotlin is an open source programming language that published by the JetBrains company for the JVM in 2011. Development with Kotlin is easier than Java language. The Google IO 2017 event has announced that the Kotlin language will now be officially supported.

Location detection is one of the most important features found on smartphones. The correct position can be obtained by combining three different methods or described methods. These are; Cellular networks, WiFi and GPS on the phone. Cellular networks

and Wifi contribute to location determination with the aid of the location method of the nearby base stations. GPS satellites play an important role in the determining position using device's GPS.

Maps are one of the best feature that help users to determine their position via their smartphone Google Maps API is Android operation systems' primary map application development libary. The great advantage of this library is every Android phone using Google Maps as a default map application. Many operations such as downloading, map viewing and gesture movements on the map are done automatically via this API. At the same time, operations such as adding points to the map, adding polygons, adding various analysis resultant data, or focusing the user's opinion on a certain part of the map are also done through the API.

The API was developed according to the object oriented programming princip. This makes application development phase uncomplicated for the developer. Developers are using API keys with the help of the side. For each application developed, an API overview is needed. This key information is generated in some application and managed entirely according to the user.

In this thesis study, A mobile application developed that place detection have been done using the instant location of the user. In the context of the meaningful information of the location determination, detected position also gives which slice number in the national layout system, the information of the middle meridian of the slice, 25K, 50K and 100K map sheet information is displayed, this map information is also plotted on the map and the coordinate information is displayed in the WGS84. Application could also transform coordinates between UTM and Gauss-Kruger systems. User could save map sheet to favorites and could take notes about map sheet.

When the application was being developed, a rich design was created in terms of ease of use and user experience which taking into account Google's material design principles. Prototypes of these designs have been created and tests have been carried out before the application is developed taking user experience principles into consideration. As a result of the tests made, the places where problems can be posed in terms of the user experience are redesigned.

With the completion of the design phase, the development phase was developed. During the development phase of the application, the Java language was used with

Android Studio development environment. For map-based operations, the Google Maps Android API is used on the Android side. Object oriented programming has been defined as a development principle.

As a result of developed application, map sheet informations has been determined with precisely defined instant location informations.This informations also determined with any location that user taped on the map. In addition of this capabilites map sheet informations could be saved for furter use. User could convert the map coordinates into required coordinate. Slice numbers that determined from instant location is also presented to app users.

1. GİRİŞ

Bu bölümde, tez çalışmasının amaçları ve kapsamından bahsedilmiştir. Literatür araştırmalarıyla birlikte yurt içi ve yurt dışı olmak üzere Android platformunda geliştirilen uygulamalar ele alınmıştır.

1.1 Tezin Amacı

Hazırlanan tezde, kullanıcıların anlık konumlarından 25 binlik, 50 binlik ve 100 binlik pafta bilgilerini görebileceği, 3 derecelik ve 6 derecelik koordinat sisteminde dilim orta meridyenini, dilim numarasını görebileceği, UTM ve Gauss-Kruger koordinat sistemleri arasında koordinat dönüşümü yapılabilecek bir uygulama geliştirilmesi amaçlanmıştır.

Yapılan görüşmeler neticesinde keşif işlemi yapan uzmanların keşif işlemlerini kolaylaştırmak amacıyla, aynı zamanda TİGEM ve Orman Bakanlığı, Madencilik işlemleri ile ilginenen kişiler iş yapımında ihtiyaç duyulan noktaların hangi paftaya denk geldiğini görmek istemektedir. İlgili uygulamaya nokta üzerinden pafta sorgulaması özelliği eklenmesi amaçlanmıştır.

Tez kapsamanıda akıllı cihazlarda kullanım oranı olarak önde gelen işletim sistemi Android seçilmiş ve Google Maps Android API harita kütüphanesi ile uygulama geliştirilmiştir.

1.2 Literatür Araştırması

Yapılan literatür çalışması neticesinde çeşitli amaçlara hizmet eden Android uygulamaları geliştirdliği görülmektedir. Geliştirilen bu uygulamalardan bazıları kullanıcının konumunu kullanmakta olup, bazıları kullanmamaktadır.

Sakarya Üniersitesi Fen Bilimleri Enstitüsünde yapılan ’Karekod Tabanlı Gıda İçerik Kontrolüne Yönelik Android Uygulaması’ tezinde tüketicilerin satın alacakların gıdaların son kullanma tarihi, üretim içeriği gibi bilgilere erişebileceği bir mobil uygulama geliştirilmiştir. (Elmalı, 2015)

Ege Üniversitesi Fen Bilimleri Enstitüsü'nde yapılan başka bir çalışmada ise Ege Üniversitesi mezunlarının bilgilerinin gösterildiği bir mobil uygulama geliştirilmiştir. (İğci, 2014)

Konumsal olmayan uygulamaların yanında konumsal amaca hizmet eden uygulamalar da geliştirilmiştir. Amerika'nın San Diego kentinde bulunan San Diego üniversitesinde yapılan bir çalışmada üniversite öğrencilerinin güvenliklerine katkıda bulunmak amacıyla acil durum anında konumlarını paylaşabilecekleri bir uygulama geliştirilmiştir. (Choundhury, 2016) Geliştirilen bu uygulamada Google Maps Android API kullanılmış ve konumunu paylaşan kullanıcının bilgileri uygulama içerisinde bulunan Google Maps yardımıyla gösterilmiştir. (Choudhury, 2016)

San Diego Üniversitesi'nde yapılan başka bir çalışmada, kullanıcıların suç analizleri yapabileceği bir mobil uygulama geliştirilmiştir. (Gonchikara, 2014) Geliştirilen uygulamada harita altyapısı olarak ESRI Android SDK kullanılmış olup kullanıcılara sorgu, adres arama, ölçüm gibi özellikler sunmaktadır.

Yine San Dieogo Üniversitesi'nde yapılmış bir çalışmada kullanıcılara konumsal olarak haber sunan bir mobil uygulama "NewsMap On The Go" geliştirilmiştir. Bu uygulama da haber servislerinden haberleri web servisi vasıtasıyla getirip konumsal bilgi ile ilişkilendirmektedir. (Mokashi, 2012)

California State Üniversitesi'nde yapılan bir çalışmada ise afet yönetimi için gönüllü coğrafi bilgi sistemi kullanıcılarının kullanabileceği bir mobil uygulama geliştirilmiştir. Bu uygulama, afet ile ilgili bilgi giren kullanıcıların girdikleri verileri harita üzerinde göstermekte ve erişilebilirlik açısından afet yönetiminde kurtarma ekiplerine fayda sağlamaktadır. (Ulaganatan, 2016)

Google Play'de geliştirilecek uygulamalara benzer uygulamalar araştırılmıştır. Bunun için "pafta" (Ek A.1) , "pafta bul" (Ek A.2), "koordinat dönüşümü" (Ek A.3) anahtar kelimeleri ile uygulama araştırılmıştır. İlgili aramalar sonucu çıkan sonuçlar belirli bir benzerlik süzgecinden geçip üst sıralara çıktığı için sonuçlar arasındaki konu ile ilgili uygulamalar incelenmiştir.

Kocaman, Hakan Kocaman & Ümit Tacettin Akgöl yayıncı kimliğiyle sunulmuş, en son güncellemesini 7 haziran 2017'de almış ve 50.000+ indirmeye sahip, kullanıcı değerlendirmelerine bakıldığında 5 üzerinden 4.6 puan almış bir mobil uygulamadır.

Uygulamanın ilk açılış ekranında hesap, gps, hakkında ve yardım menüleri bulunmaktadır. Hesap menüsünde Semt Mesafe, Açı Mesafe, Kutupsal Aplikasyon, Yan Nokta Hesabı, Ara Nokta Hesapla, Dik Ayak - Dik Boy, Basit Kurp Hesabıü 2B Koordinat Dönüşümü, Coğrafi ve UTM koordinatları arasında dönüşüm, Tecviz Hesabı ve Derece Dönüşüm işlemlerini yapan özellikleri bulunmaktadır. GPS menüsünde GPS ile aplikasyon, Gps ile Alım ve GPS ile alan hesabı bulunmaktadır. Hakkında menüsünde uygulamaya ait iletişim bilgileri ve sürüm notları bulunmaktadır. Yardım menüsünde de uygulamanın kullanımına ait bilgiler bulunmaktadır. Uygulama pafta bilgileri ile ilgili bir özellik sunmamaktadır. Ayrıca harita üzerinden kullanıcı konumuna dayanarak pafta bilgilerinin sunulmasına dair bir özellik de sunmamaktadır. Uygulamanın tasarımsal özelliklerine bakıldığında materyal tasarıma uygun tasarlanmadığı, eski Android versiyonlarına göre tasarlandığı görülmektedir. Ayrıca uygulamanın ücretli versiyonu olan PRO sürümü bulunmaktadır. (URL - 1)

KorTrans Pafta, Kerem KAT yayıncı kimliğiyle sunulmuş, en son güncellemesini 24 Aralık 2011'de almış ve 10.000 - 50.000 indirmeye sahip, kullanıcı değerlendirmelerine bakıldığında 5 üzerinden 4.2 puan almış bir mobil uygulamadır. Uygulamanın ilk açılış ekranında Koordinat Dönüşümü, Haritadan Seç, Kayıtlı Noktalar, Ayarlar, Yardım Videoları ve Hakkında menüleri bulunmaktadır. Kooordinat dönüşümü özelliğinde Ondalık Derece, Derece Dakika Saniye ve UTM koordinatları arasında dönüşüm yapma imkanı bulunmaktadır. Haritadan seç özelliği ile haritadan seçilen bir noktaya ait pafta ve koordinat bilgileri kullanıcıya sunulmaktadır. Kayıtlı noktalar özelliği ile kullanıcıların kaydetmek istediği bilgileri saklamalarına imkan vermektedir. Ayarlar özelliği elipsoid türünü göstermek, açılış ekranının gösterilmesi, pafta sınırlarının kullanıcıya gösterilmesi gibi özelliklere dair ayarlamalar yapmaya imkan sağlamaktadır. Yardım videoları uygulamanın kullanımına ait bilgiler içermektedir. Hakkında özelliği de uygulama ait iletişim bilgileri ve diğer bilgileri içermektedir. Uygulama en son 2011'de güncelleme aldığı için yeni trendlere uygun tasarımı bulunmamaktadır. Uygulamada bulunan harita bileşenleri ve harita güncel değildir. (URL - 2)

TKGM Parsel Sorgu, Tapu ve Kadastro Genel Müdürlüğü yayıncı kimliğiyle sunulmuş, en son güncellemesini 26 Ocak 2017'de almış ve 500.000 - 1.000.000 indirmeye sahip, kullanıcı değerlendirmelerine bakıldığında 5 üzerinden 3.9 puan

almış bir mobil uygulamadır. Uygulamanın ilk açılış ekranında kişinin konumu belirlenerek ilgili altlık harita üzerinde sunulmaktadır ve konumun sahip olduğu hassasiyet kullanıcıya gösterilmektedir. Uygulama native olarak hazırlanmamış olup, cross platform olarak hazırlanmıştır. Uygulama ana işlev olarak konuma git, istenilen noktadan bilgi al, komşu parselleri göster, ilgili parseli kaydet ve yazdır gibi özellikleri barındırmaktadır. Yapılan sorgulama neticesinde kullanıcıya gösterilen parsel bilgisinde il, ilçe, mahalle/köy, ada, parsel, tapu alanı, nitelik, mevkii ve pafta bilgileri bulunamtadır. (URL - 3)

Coordinator – Koordinat Topla, Süleyman Er yayıncı kimliğiyle sunulmuş, en son güncellemesini 31 Mayıs 2017'de almış ve 10.000 - 50.000 indirmeye sahip, kullanıcı değerlendirmelerine bakıldığında 5 üzerinden 4.3 puan almış bir mobil uygulamadır. Uygulama genel işlevi olarak istenilen projeksiyon bilgisinde kooordinat toplama amacı gütmektedir. Ayrıca alan hesabı, mesafe hesabı gibi özellikler de sunuluş durumdadır. (URL - 4)

2. ANDROİD İŞLETİM SİSTEMİ

Android işletim sistemi, açık kaynak kodlu olarak linux tabanlı geliştirilmiş bir işletim sistemidir. Google tarafından geliştirilmiş bir işletim sistemi olup ücretsizdir. Android işletim sisteminde yayınlanan uygulamalar kullanıcılara Google Play aracılığı ile sunulmaktadır. Android işletim sisteminin ilk sürümü olan Astro 9 şubat 2009'da piyasaya sürülmüştür. Hala yayında olan son sürümü Oreo ismine sahip 8.0 versiyonlu sürümüdür. (URL - 5) Bu sürüm geliştirme aşamasında olup telefonlara dağıtımı Nexus ve Google Pixel cihazlara gerçekleştirilmiştir. Kullanım yüzdesi %0.1'in altında olduğu için ilgili tabloda gösterilmemiştir. (URL - 6)

Çizelge 2.1: Android versiyonları ve kullanım yüzdeleri.

Versiyon	Adı	Geliştirici Seviyesi	Kullanım Yüzdesi
2.3.3 - 2.3.7	Gingerbread	10	0.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.7%
4.1.x	Jelly Bean	16	2.7%
4.2.x		17	3.8%
4.3		18	1.1%
4.4	KitKat	19	16.0%
5.0	Lollipop	21	7.4%
5.1		22	21.8%
6.0	Marshmallow	23	32.3%
7.0	Nougat	24	12.3%
7.1		25	1.2%
8.0	Oreo	26	-

Android işletim sistemi çekirdek, Android Runtime, kütüphaneler, uygulama çatısı ve uygulama katmanı olmak üzere beş bölümden oluşur. Çekirdek kısmı Android'in güvenlik özelliklerinin cihaz üreticileri tarafından kendi cihazlarına uygulamasına yardımcı olur. Android Runtime; sanal makinedir ve geliştirilen kodları Android platformunda çalıştırmak üzere Android işletim sisteminin anlayacağı dile dönüştürür. Kütüphaneler, uygulamalarda bulunan veritabanı, grafik, web tarayıcıları bileşenlerini

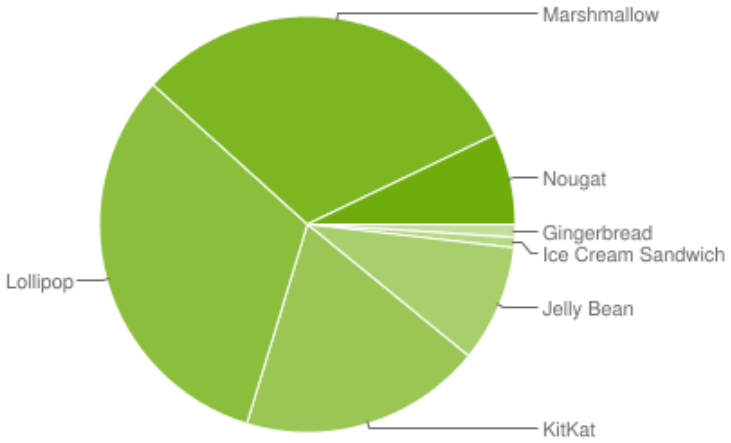
içermektedir. Uygulama çatısı geliştiricilerin kullandığı kısımdır. Uygulama katmanı ise Java dilinde geliştirilmiş uygulamaları temsil etmektedir. (URL - 5)

Andıroid uygulama geliştirilirken kullanılan programmala dili Java programlama dilidir. Java programlama dili de Android işletim sistemi gibi açık kaynak kodlu geliştirilmekte olan bir programlama dilidir. Android uygulamaları sadece Java ile geliştirilmemekte olup Kotlin dili ile de geliştirilmektedir. Kotlin dili JetBrains firmasının 2011 yılında JVM için yayınladığı bir dildir. Java diline kıyasla geliştirme işlemleri daha kolay yapılmaktadır. Google'ın düzenlediği Google IO 2017 etkinliğinde Kotlin dilinin artık resmi olarak destekleneceği duyurulmuştur. (URL - 7) Bu gelişme de gelecekte Android uygulama geliştirmenin eskiye nazaran daha modern bir şekilde olacağını göstermektedir.

3. ANDROİD SDK

Android yazılım geliştirme kiti olarak adlandırılan Android SDK, uygulama geliştiricilerin android uygulama geliştirmesine yardımcı olan bileşenler topluluğundur. İçinde örnek projeler ve bu projelere ait kaynak kodları barındırır. Aynı zamanda uygulamaları çalıştırmak için tüm Android versiyonlarının test edilebileceği Android emülatorleri içermektedir.

Her yeni Android versiyonu yayınlandığında Android SDK da en güncel versiyona güncellenmektedir. Uygulama geliştiricileri uygulamalarını son versiyona uyumlu hale getirmek istediğinde bu SDK'yı indirmektedir. Sadece en güncel versiyonları kapsamaktan ziyade henüz geliştirilme aşamasında olan Android versiyonlarını da beta sürümü olarak geliştiricilere sunmaktadır. (URL – 8)



Şekil 3.1: Android SDK versiyonları ve kullanım oranı.

4. MOBİL HARİTA KÜTÜPHANELERİ

Mobil harita kütüphaneleri Android uygulamalarda harita tabanlı işlemler yapabilmek için kullanılan kütüphanelerdir. Bu kütüphanelerin en önemli özelliği kullanıcının konumuna göre işlem yapmasıdır. Bu işlemi gerçekleştirirken akıllı telefonların konumuna ihtiyaç duymaktadır.

4.1 MapBox Android SDK

Açık kaynak kodlu harita uygulamaları geliştiren MapBox şirketinin geliştiricilere sunduğu harita tabanlı Android uygulama geliştirme kütüphanesidir. 5.0.2 güncel versiyonu ile yayında bulunmaktadır. İlgili versiyon kullanıcılara anlık trafik bilgisi, çevrimdışı harita çalıştırabilme, çalışma zamanında stil düzenlenmesi ve 3 boyutlu haritaların gösterilmesi gibi özellikler sunmaktadır. (URL – 9)

4.2 OSMDroid

Altyapısında OpenStreetMap haritalarını kullanan açık kaynak kodlu harita kütüphanesidir. En önemli özelliği ücretsiz olarak kullanıcılara sunulmasıdır. Kullandığı altlık haritalar ise Coğrafi Bilgi Sistemi gönüllüleri tarafından oluşturulmuş hairtalardır. (URL – 10)

4.3 ESRI Android SDK

ESRI bünyesinde bulunan web servisleri ile iletişim için geliştirilmiş Android kütüphanesidir. ArcGIS Server vasıtasıyla yayınlanan harita servisleri, sorgu servisleri ve rota servisleri bu SDK ile geliştirilen Android uygulamalarında kullanılabilmektedir. (URL – 11) İki farklı versiyon olarak yayında bulunmaktadır. Bunlardan ilki 100.0.0 versiyonudur. Diğeri ise 10.2.9 versiyondur. 100.0.0 versiyonu güncel geliştirme prensiplerine uyularak geliştirilmiş SDK'dır. 10.2.9 versiyonu ise SDK'nın başında oluşturulmuş ve çeşitli iyileştirmelerle bu güne gelmiş versiyonudur.

ESRI yeni uygulama geliştirmek istendiğinde 100.0.0 versiyonu ile geliştirme yapılmasını önermektedir. Öte yandan eski versiyon ile uygulama geliştirildiyse de bu versiyonda devam edilmesini önermektedir. (URL – 12)

4.4 Google Maps Android API

Google tarafından geliştirilmiş, Android uygulamalara özelleştirilebilir harita yetenekleri eklenebilen kütüphanedir. Bu kütüphane ile uygulama geliştirmenin en büyük avantajı piyasada olan Android telefonlarda varsayılan harita uygulaması olarak Google Maps kullanılmasıdır. Uygulama bazında veri indirme, harita görüntüleme ve haritada yapılan jest hareketleri gibi işlemler API vasıtasıyla otomatik olarak yapılmaktadır. Aynı zamanda haritaya nokta eklenmesi, polygon eklenmesi, çeşitli analiz sonucu oluşmuş verilerin eklenmesi ya da kullanıcnın görüşünün haritanın belirli bölümüne odaklanması gibi işlemler de API vasıtasıyla yapılmaktadır.

API nesneye yönelik programlama prensiplerine göre geliştirilmiştir. Böylelikle geliştirici açısından uygulama geliştirme işlemi kolaylaşmaktadır. Bu uygulama kütüphanesi 10.2.6 versiyonu ile yayında bulunmaktadır.

Geliştiriciler tarafında kullanılması API anahtarı yardımıyla olmaktadır. Geliştirilen her bir uygulama için bir API anahatarına ihtiyaç duyulmaktadır. Bu anahtar bilgisi uygulama bazı oluşturulmaktadır ve tamamen kullanıcıya bağlı olarak yönetilmektedir. (URL – 13)

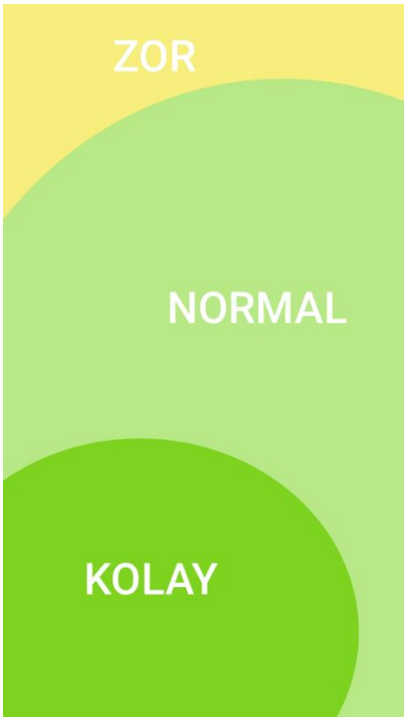
5. UYGULAMA

Uygulama geliştirme aşaması 4 evreden oluşmaktadır. tasarım, geliştirme, test ve tayın. Geliştirme platform olarak Android işletim sistemi seçilmiştir. Bu seçime karar verilirken kullanım yaygınlığı göz önünde bulundurulmuştur.

5.1 Tasarım

Uygulamanın tasarımı yapılırken kullanıcı deneyimi dikkate alınmış ve Google tarafından belirlenmiş material tasarım ilkeleri göz önünde bulundurularak tasarım yapılmıştır. Bu ilkelere göre,

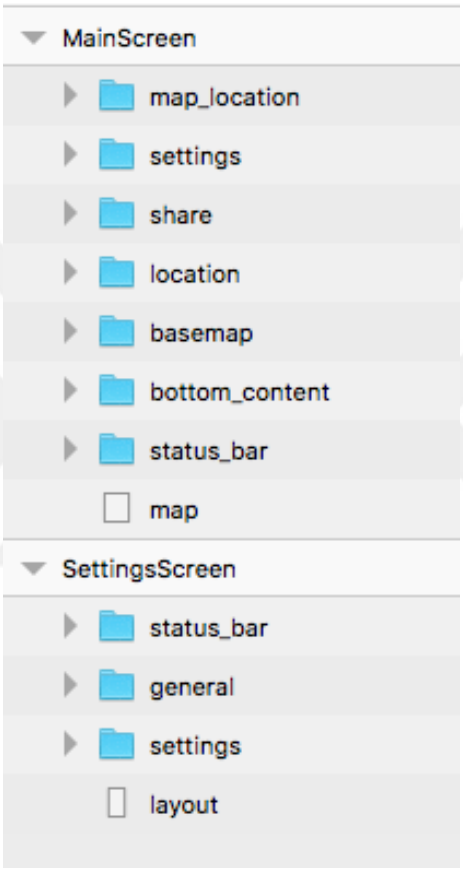
Mobil uygulama kullanımlarında kullanıcının erişebildiği alanlar yukarıya doğru gidildikçe azalmaktadır. Bu nedenle uygulamanın ana işlevleri aşağıda toplanacak şekilde bir tasarım yapılmıştır. (URL – 14)



Şekil 5.1: Akıllı telelefon ekranlarının kullanım durumuna göre kısımları.

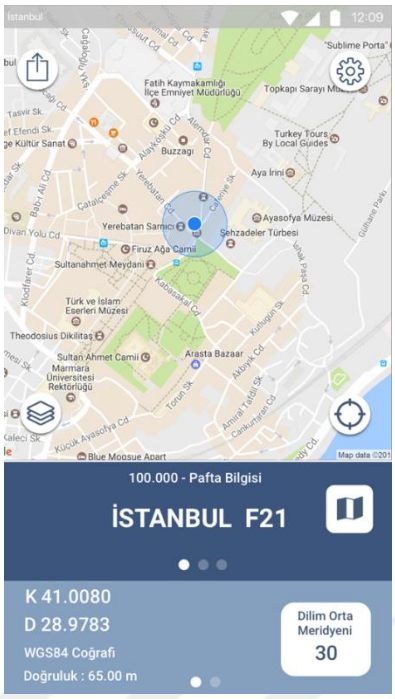
Yine kullanıcı deneyimi ilkeleri göz önüne alındığında mümkün olduğunca az tıklama ile işlem yapılacak bir tasarım yapılmıştır. Farklı amaçlara hizmet eden işlemler aynı ekranda yapılmaktan ziyade farklı ekranda gerçekleştirilmiştir.

Uygulama tasarımları Sketch adlı tasarım programı ile yapılmıştır. Tasarım aşamasında grafik öğelerin katmanlanarak çalışılmasına dikkat edilmiştir. Grafik öğeleri gruplayarak çalışmak geliştiriciye daha sonra kullanmak üzere dışa aktarma işleminde kolaylık sağlamaktadır.



Şekil 5.2: Katmanlanarak çalışılan grafik öğelerinin dosya yapısı.

Uygulama kullanım olarak iki ekran olarak tasarlanmıştır. Bu ekranlar kullanıcının harita ile ilgili işlemlerini yapabileceği ana ekran ve uygulama ile ilgili birtakım ayarlamalar yapabileceği ayarlar ekranı. Ana ekranda belirlenen anlık konuma göre 100.000'lik, 50.000'lik ve 25.000'lik pafta bilgileri gösterilmektedir. Aynı zamanda belirlenen koordinat bilgisi bulunmaktadır. Dilim orta meridyeni bilgisi ve belirlenen pafta bilgisinin haritada çizdirilme özelliği de uygulamanın ana ekranında kullanılmak üzere tasarlanmıştır.

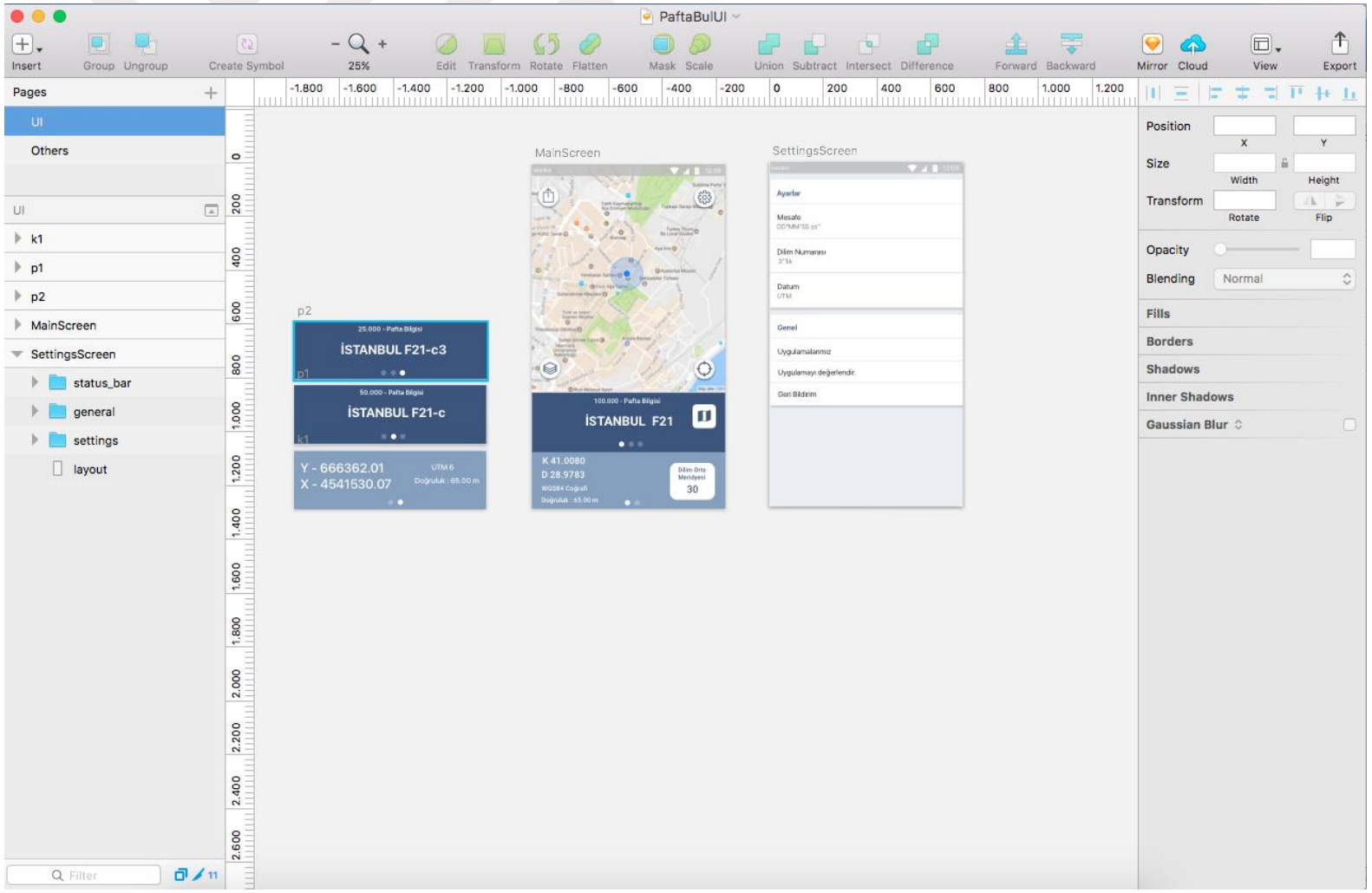


Şekil 5.3: Ana ekran tasarımı.

Ana ekranın yanısıra ayarlar ekranında da kullanıcı koordinat gösterim şeklini, dilim numarasını ve UTM tipini değiştirebilmektedir. Aynı zamanda uygulama geliştiricisinin diğer uygulamalarını görüntüleyebilmekte ve uygulamayı değerlendirebilmektedir.

Şekil 5.4: Ayarlar ekran tasarımı.

Tasarım yapılırken sadece sayfaların tasarımı yapılmamış, aynı zamanda ekran geçişlerinde gösterilecek olan diğer bilgilerin de eklendiği bir tasarım yapılmıştır.



Şekil 5.5: Sketch uygulaması ile yapılan tasarım.

5.2 Geliştirme

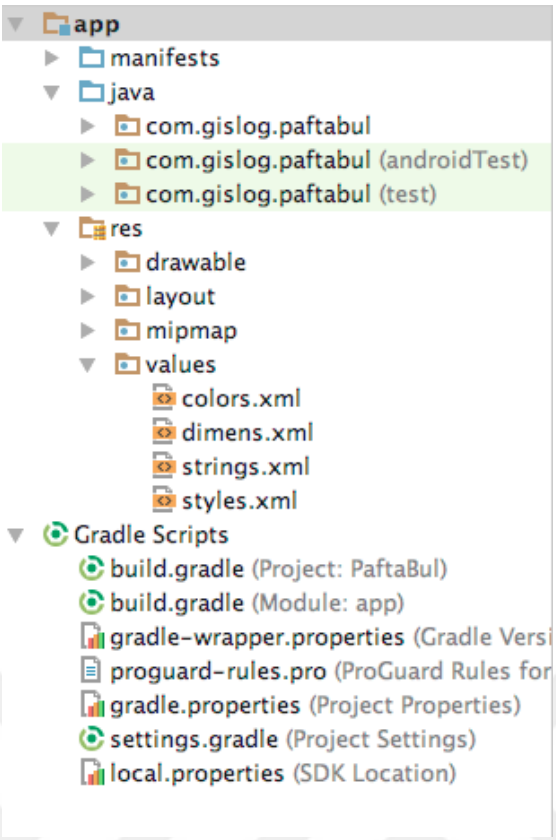
Geliştirme aşamasında Android Studio programı ve Java dili kulanılmıştır. Proje açılırken desteklenecek en alt sürüm olarak Android 4.1 seçilmiştir. Bu sürüm seçildiğinde geliştirilen uygulama Android kullanıcıların %95.2'sinin kullanımına sunulmuş olmaktadır.

Android Studio ile geliştirilen Android uygulamalarında gradle adı verilen build sistemi bulunmaktadır. Bu sistem ile uygulamaya eklenecek kütüphanelerin yönetimi otomatik olarak sağlanmaktadır.

Uygulama önceki Android versiyonlarını desteklediği için ana kütüphane olarak Google tarafından geliştirilen AppCompat kütüphanesi kullanılmıştır. Tasarım aşamasında kullanılmak üzere yine Google tarafından geçiştirilmiş CardView bileşeni kütüphanesi kullanılmıştır. Veri transferinden kullanılmak üzere GSON kütüphanesi, yazı stillerinin değiştirilmesi için Calligraphy kütüphanesi ve konum izni alınması için PermissionDispatcher kütüphanesi kullanulmıştır.

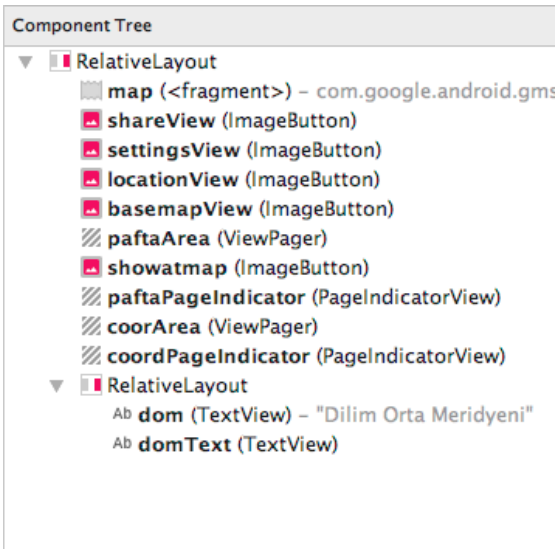
Android 6.0 ile birlikte kullanıcılardan istenen kritik izinler (konum, sms, rehber erişimi vb.) uygulamanın çalışması anında alınması gerekmektedir. Android 6.0 öncesi versiyonlarda ise bu izinler kullanıcı uygulamayı yüklediği zaman alınmaktadır. Bu izinler manifest adını verdiğimiz dosyaya tanımlanarak Android 6.0 öncesi sürümler için alınmış olmaktadır.

Uygulama yapı olarak bakıldığında iki kısımdan oluşmaktadır. App kısmı ve Gradle kısmı. App kısımında uygulamanın temel işlevlerini meydana getiren manifest, java ve res klasörleri bulunmaktadır. Gradle kısmında ise kütüphaneler ve uygulamayı yayınlamak ile ilgili bilgiler bulunmaktadır. Manifest bölümü uygulama adı, ikonu, kullanılan google api anahtar bilgisini ve hangi ekranın ilk açılışta kullanıcıları karşılayacağını belirleyen kısımdır. Res kısmı uygulamanın tasarımsal bileşenlerinin bulunduğu kısımdır. Burada ikonlar, ekranlar, renkler ve sözel ifadelerin tutulduğu bölümler bulunmaktadır. (Şekil 5.6)



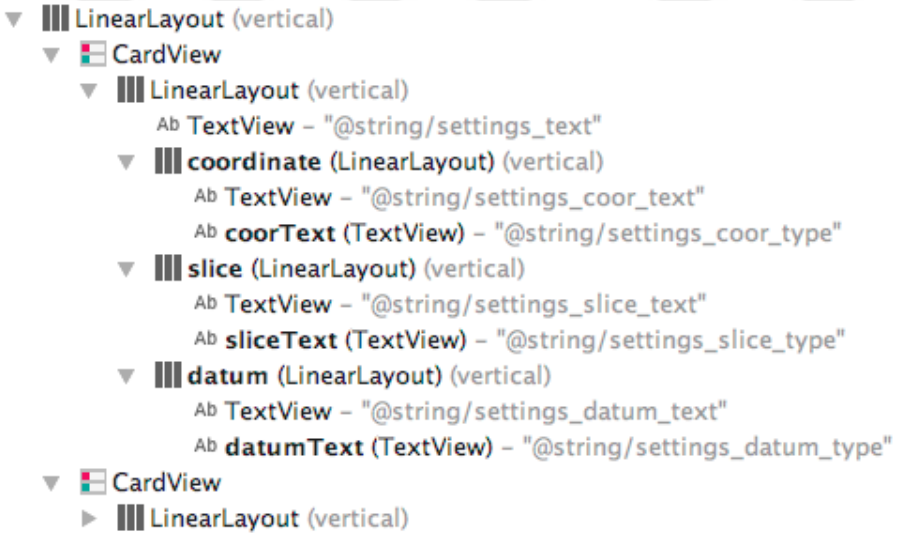
Şekil 5.6: Oluşturulan projenin dosya yapısı.

Tasarlanan uygulama ekranları layout kısmından XML ile geliştirilmiştir. (URL Ana ekran tasarlanırken Sketch dosyasında çizilmiş tasarımı yakalamak için konuma git, altlık harita değiştir, paylaş ve ayarlar özellikleri için ImageButton bileşeni kullanılmıştır. Bu bileşenin kullanılmasındaki amaç butona ekstradan bir görsel de ekleme imkanı sağlamasıdır. Koordinat ve pafta bilgilerinin kaydırılarak gösterilmesi için ViewPager ve PageIndicatorView bileşenleri kullanılmıştır. (URL – 15) PageIndicatorView bileşeni kullanıcı sayfalar arasında geçiş yaptığında kullanıcının hangi sayfada olduğunu göstermektedir. Harita gösterimi için de Google'a ait SupportMapFragment bileşeni kullanılmıştır. (Şekil 5.7) Uygulama tasarımı yapılırken en küçük ekran boyutu olarak 4 inç seçilmiştir ve diğer ekranlara da uyumlu olacak bir tasarım yapılmıştır.



Şekil 5.7: Ana ekran bileşenleri.

Ayarlar ekranı tasarlanırken ana kısımlar için CardView bileşeni kullanılmıştır. Bu kısımda hiyerarşik bir düzen olduğu için ana bileşen olarak dikey LinearLayout kullanılmıştır. Ayrıca her CardView'in içine yine aynı hiyerarşik yapıyı oluşturmak için bir dikey LinearLayout kullanılmıştır. Bu layoutların içine de TextView yerleştirilerek tasarımda belirlenen ekranlar oluşturulmuştur. (Şekil 5.8)

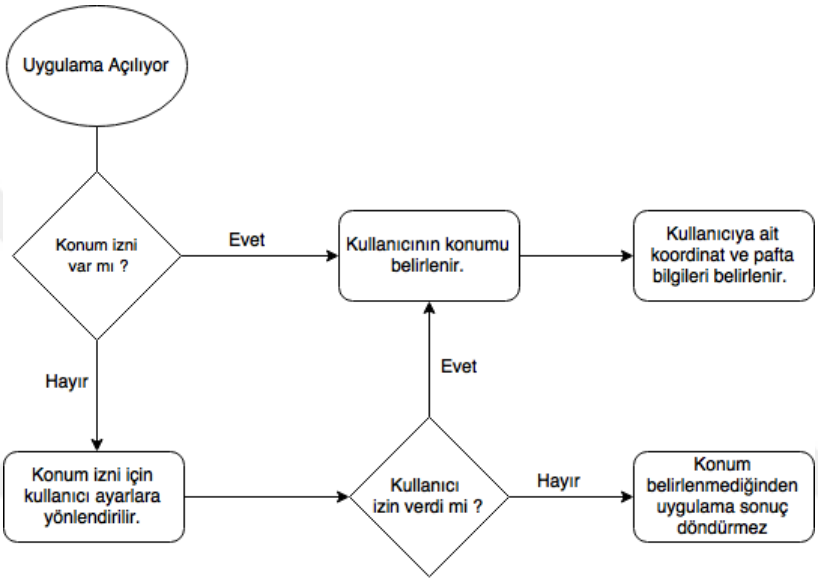


Şekil 5.8: Ayarlar ekranı bileşenleri.

İlgili ekranları Java kodlarında temsil eden sınıfları Activity adlı Android bileşenleridir. Bu bileşenlerin onCreate metodu bulunmaktadır ve bu metod uygulama ilk açıldığı zaman çalışan metoddur. XML ile oluşturulan tasarımların Java kodları ile kullanulması için View nesnelerini Java tarafına bağlamak gerekir. Bu da findViewById metodu ile gerçekleştirilmektedir. Fakat uygulamadaki View sayısı arttıkça bu metodu kullanmak da kaliteli kod geliştirme açısından

birbirini tekrar eden kodların türemesine neden olacaktır. Bu nedenle projeye ButterKnife kütüphanesi eklenmiştir. Bu kütüphane ile XML kısmı ve Java kısmı arasındaki bağlantı kolay yoldan halledilmektedir ve uygulama geliştirme süresine olumlu etkide bulunmaktadır.

Uygulama çalışma prensibi ise şu şekilde ilerlemektedir. Kullanıcı uygulamayı açtığı zaman konum izni var mı yok mu diye control edilmektedir. Eğer konum izni verilmişse uygulama çalışmaya devam eder. Konum izni verilmemişse kullanıcı konum izninin alınması için ayarlar ekranına yönlendirilir. Ayarlar ekranında da konum izni veren kullanıcı tekrar uygulamayı çalıştırabilmektedir.



Şekil 5.9: Uygulamanın çalışma prensibi.

Uygulamanın çalışması ile birlikte kullanıcının konumunun belirlenmesi için konum güncellemeleri istenmektedir. Bu güncellemelere Android sisteminin LocationManager yardımı ile ulaşılmaktadır. Birinci güncelleme Network Provier adı verilen konum bilgisidir. Bu güncelleme ile kullanıcının konumu etrafta bulunan baz istasyonları ve WiFi ağları yardımıyla belirlenmektedir. Belirlenen konum hassas değildir. Bir diğer yöntem olan GPS Provider yöntemi ile de telefonda bulunan GPS özelliği vasıtasıyla GPS uyduları kullanılarak konum belirlenmektedir. Bu yöntem Network Provider yöntemine göre daha hassas bir yöntem olup, konum belirlenmesi zaman olarak Network Provider'a göre daha uzun sürmektedir ve hava şartlarından olumsuz etkilenmektedir. Aynı zamanda kapalı mekanlarda GPS ile konum belirlenmesi Network Provider'a göre daha zor olabilmektedir. Uygulama bazında Network Provider ile işlem yapmak için ACCESS FINE LOCATION iznine ihtiyaç duyulmaktadır. GPS Provider ile yapılan işlemler hassas sonuçlar getirdiği için bu özelliği kullanmak için de ACCESS COARSE LOCATION iznine ihtiyaç duyulmaktadır. (URL - 16) Geliştirilen

uygulamada hassaslık bilgisi önemli olduğu için hem GPS hem de Network Provider'dan gelen konum bilgisi kullanılmıştır. Konum güncellemeleri 100 milisaniyelik periyotlarla ve 3 metrelik değişim olduğunda değişecek şekilde ayarlanmıştır.

Konum belirlenirken farklı kaynaklardan gelen bilgiler ile işlem yapıldığından kullanıcıya en doğru konum sunulmuştur. Bu amaçla bir metod yazılmış ve bu metoda göre işlem yapılmıştır. Bu metoda göre, parametre olarak değerlendirilecek yeni konumu ve önceki konumu almaktadır. Eğer hiç konum bilgisi gelmediyse metod yeni gelen konum bilgisini en iyi konum bilgisi olarak almaktadır. Bununla birlikte eğer metoda parametre olarak hem yeni gelen konum hem de değerlendirilecek konum alındığında ise en iyi konumu belirlemek için şu yöntem izlenmektedir.

İlk olarak yeni konum bilgisinin eski mi yeni mi olduğunun anlaşılması için değerlendirilen iki konumun zaman bilgisi karşılaştırılır. Belirlenen iki konum arasındaki zaman farkı 2 dakikadan küçükse ve çıkan fark 0'dan da büyükse elde edilen konum bilgisinin yeni bir konum olduğu anlaşılır ve bu konum ile işlem yapılır. Eğer istenilen değerler çıkmazsa karşılaştırılacak iki konum arasındaki doğruluk değerleri karşılaştırılır. Bu doğruluk değeri 0'dan küçük ise gelen verinin doğru veri olduğu anlaşılır. Bununla birlikte elde edilen konum bilgilerinin aynı sağlayıcıdan mı yoksa farklı sağlayıcıdan mı elde edildiği karşılaştırılır. Tüm bu parametreler ışığında yeniden karşılaştırma yapılır. Eğer konum verilerinin karşılaştırılmasıya elde edilen mantıksal değer doğru sonucu veriyorsa işleme devam edilir. Yine bir sonuç alınmadıysa elde edilen konum bilgisinin yeni mi elde edildiğine veya doğruluğunun az olup olmadığına bakılır. Bu iki karşılaştırmadan biri sağlanıyorsa o konum ile işlem yapılır. Bu parametrelerle de işlem yapılamıyorsa konum sağlayıcıları eklenerek yeni bir değerlendirme yapılır ve en iyi konum bilgisi kullanılmak üzere sunulur. (Ek – A.4)

Belirlenen konum WGS84 koordinat sisteminde Coğrafi Koordinatlar olarak gösterilmektedir. İkincil koordinat olarak ise bu koordinatlardan yapılan dönüşüm ile kartezyen koordinatlar gösterilmektedir. Kartezyen koordinatların datum bilgisi ve dilim numarası uygulama içinden değiştirilebilir şekilde geliştirilmiştir.

Atlık haritalar olarak Google tarafından sağlanan vektör harita ve uydu görüntüsü kullanılmıştır. Kullanıcı bu altlık haritalar arasında geçiş yapabilmektedir.

Paylaşma özelliği geliştirilirken, uygulama açık iken ekranda bulunan pafta bilgileri, koordinat bilgisi, datum bilgisi ve dilim orta meridyeni bilgisinin başka kullanıcılarla paylaşılması olanağı sağlanmıştır.

Konuma git özelliği ile kullanıcının kendi konumuna gitmesi imkanı sağlanmıştır. Böylelikle haritanın farklı yerlerine giden kullanıcı kendi konumuna kolaylıkla ulaşabilmektedir.

Google tarafından yayınlanmış bulunan Android tabanlı harita uygulaması geliştirme kütüphanesi Google Maps Android API, kullanıcının konumunun gösterilmesi için setMyLocationEnabled metodunu bullanmaktadır. Bu metodun aktif edilmesiyle birlikte kullanıcının konumuna gitmesi için sağ üst köşede bir konuma git butonu otomatik olarak oluşmaktadır. Pafta bul uygulamasında bu konum butonunun uygulama tasarımına uygun olması istendiği için değiştirilmiştir. Bu değişim uygulamala kodlarını yazdığımız .java uzantılı dosyaların bulunduğu geliştirme kısmında yapılmıştır. Buna göre;

Google kütüphanesinde haritalar kullanıcıya MapFragment adı verilen bir bileşen ile sunulmaktadır. Bu bileşenin 2. elemanı konum butonu olarak görülmektedir. findViewById metodu ile ilgili konum butonuna erişim sağlanmaktadır. Erişim sağlandıktan sonra konumu temsil eden butonun ImageView denilen bileşen olduğu görülmektedir. Böylelikle bu ImageView'e ilgili tasarımsal düzenlemeler yapılarak konuma git butonunun tasarıma uygun hale getirilmesi sağlanmıştır. (Ek A.5)

Kullanıcılara bilgilendirme yapılması amacıyla anlık bildirim gönderme özelliği eklenmiştir. Bu özellik ile geliştirici kullanıcılara belirli bir zaman diliminde gönderilmek üzere ya da anlık göndermek üzere bildirim gönderebilmektedir. Bu da kullanıcıların uygulama ile etkileşimlerini arttırmaktadır.

Kullanıcıların uygulama içindeki davranışlarının incelenmesi için uygulama analiz yapılabilecek bir özellik eklenmiştir. Bunun amaçla Google tarafından geliştirilen Firebase Analytics aracı kullanılmıştır. Bu araç yardımı ile uygulamayı kullanan kullanıcıların yaş durumu, telefon modelleri, işletim sistemi sürümleri, uygulamayı ortalama kullanma süresi gibi metrikler hakkında bilgi edinilebilmektedir. Bu bilgiler çerçevesinde kullanıcı hareketleri incelenip çeşitli güncelleme ile kulanıcıyı daha fazla mutlu edecek iyileştirmeler yapılmaktadır.

5.2.1 Bulunan konuma ait dilim orta meridyeni gösterme

Belirlenen konuma ait enlem ve boylam bilgileri kullanılarak 3 derecelik dilim orta meridyeni hesaplanırken boylam bilgisi 1.5 ile toplanıp 3 e bölündükten sonra çıkan sayının tam sayı kısmı alınmış ve bu 3 ile çarpılarak 3 derecelik koordinat sistemine ait dilim orta meridyeni bulunmuştur. (Ek A.6)

$$\lambda_0 = 3\ int\left(\frac{\lambda + 1.5}{3}\right) \quad \textbf{(5.1)}$$

6 derecelik koordinat sistemine ait dilim orta meridyeni hesaplanırken boylam bilgisi 6'ya bölünmüş ve elde edilen sayının tam kısmı alınarak 6 ile çarpılıp 3 eklenmiştir. (Ek. A.6)

$$\lambda_0 = 6\ \mathrm{int}\left(\frac{\lambda}{6}\right) + 3 \quad \textbf{(5.2)}$$

Uygulamanın ilk açılışında 3 derecelik sisteme ait dilim numarası gösterilmektedir. 6 derecelik koordinat sistemine ait dilim orta meridyenini görmek isteyen kullanıcı ayarlar ekranından dilim tipini değiştirebilmektedir.

5.2.2 Üç derecelik ve altı derecelik koordinat sistemleri arasında dönüşüm

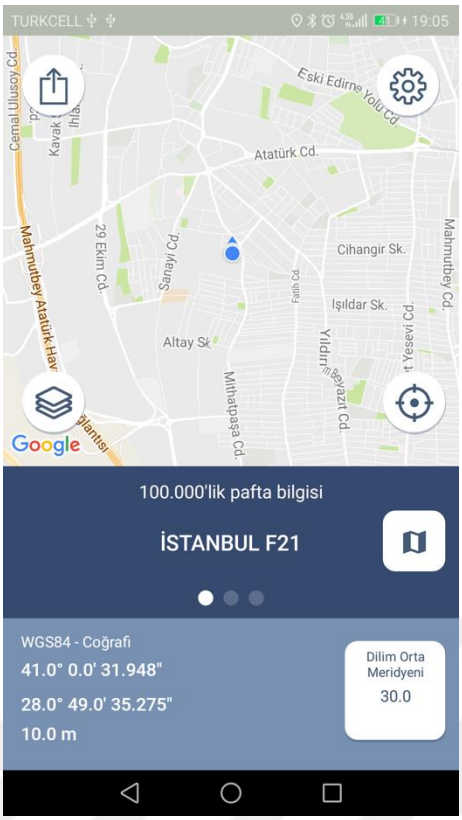
Ülkemizde haritalar üretilirken 1/25.000'den küçük ölçekler için 6 derecelik koordinat sistemi, 1/5000 ve daha büyük ölçekler için 3 derecelik koordinat sistemi kullanılmaktadır. Geliştirilen uygulama ile coğrafi koordinatlardan Gauss-Kruger ve UTM sistemlerinde koordinatlar hesaplanmıştır. Kullanıcının konumundan elde edilen koordinat bilgisi 3 derecelik ve 6 derecelik koordinat sistemleri arasında dönüşüm yapma imkanı sağlamaktadır. Böylelikle kullanıcı hem coğrafi koordinatlara hem de dönüştürülmüş koordinatlara aynı ekran üzerinden hızlı bir şekilde ulaşabilmetedir. (URL – 17)

Şekil 5.10: Koordinatların gösterildiği ekran görüntüsü.

5.2.3 Derece dakika saniye ve ondalık koordinat gösterimi

Uygulama ile elde edilen koordinatlar ondalıklı olarak gelmektedir. Ondalıklı koordinat bilgisinin yanında kullanıcı derece dakika saniye cinsinden de koordinatları görmek isteyebilir. Bu nedenle gelen koordinat bilgisi derece dakika saniye şekline de dönüştürülmüştür. Bu sayede isteğe bağlı olarak istenilen formatta koordinat bilgisi görüntülenebilmektedir.

Bu işlem yapılırken ondalık koordinatın tam kısmı dereceyi ifade etmektedir. Kalan değerin 60 ile çarpılmasıyla elde edilen yeni değerin tam kısmı dakikayı ve en son kalan kısmın 3600 ile çarpılmasıyla elde edilen değerin tam kısmı da saniyeyi ifade etmektedir.



Şekil 5.11: Derece dakika saniye şeklinde gösterilen koordinat.

5.2.4 Pafta bilgilerinin haritada gösterilmesi

Pafta belirleme işlemi uygulama içinde iki türlü yapılmaktadır. Kullanıcının konumundan anlık pafta bilgisi ve harita üzerinde tıklanan herhangi bir noktaya ait pafta bilgisi.

Her iki durumda da pafta bilgilerinin belirlenebilmesi için konum bilgisi kullanılmıştır. Tek bir meted ile enlem ve boylam bilgisinden pafta bilgileri getirilmektedir. Bunun için Türkiye Pafta bölümlemesine ait olan 250.000'lik pafta bilgilerinin bulunduğu iki boyutlu bir dizi oluşturulmuştur. (Ek A.7) Enlem ve boylam bilgisinden ilk olarak 250.000'lik pafta bilgilerine ulaşılmaktadır. Buna göre, 42 derece kuzey paraleli ve 25.5 derece doğu meridyeni referans olarak alınmaktadır. İlgili enlem boylam ile referans alınan enlem boylam arasındaki farkı bulunmaktadır. Bu fark neticesinde 250.000'lik paftanın enlemleri arası 1 derecelik fark ve boylamları arasında 1 derece 30 dakikalık fark bulunduğundan enlem ve boylam farklarında kaç adet fark bulunduğu tespit edildikten sonra noktanın 250.000'lik paftada yeri belirlenmektedir. (Ek A.8)

250.000’lik paftada yeri belirlenen pafta bilgisinden sonra uygulamanın ana işlevi olan 100.000’lik, 50.000’lik ve 25.000’lik pafta bilgilerinin yerlerinin belirlenmesi amacıyla ilgili paftaların köşe uzunluklarına göre işlem yapılmakta va pafta bilgileri belirlenmektedir.

Harita üzerinde bir noktada pafta bilgisinin belirlenmesi için de haritaya tıklanan noktanın enlem ve boylam bilgisi gösterilerek farklı ekrana geçmeden aynı ekranda kullanıcıya sonuç gösterilmektedir.

Şekil 5.12: Haritadan herhangi bir noktadan alınan pafta bilgisi.

Kullanıcı konumundan belirlenen pafta bilgileri daha sonradan kullanılmak üzere kaydedilmektedir. Bu kayıt işlemi SharedPreferences adı verilen uygulama ait dosyaların bulunduğu kısımda lokal olarak tutulmaktadır. Ayrıca kullanıcı istediği paftaya ait notlar ekleyebilmekte ve bu notları görüntüleyebilmektedir.

Şekil 5.13: Kaydedilen paftalar ve paftalardan birine ait not.

5.2.5 Parsel bilglerinin haritada gösterilmesi

Parsel bilgilerinin Google Harita kütüphanesi üzerinde vektörel olarak gösterilmesi amacıyla İstanbul iline ait Zeytinburnu ilçesine ait .shape uzantılı dosyalar kullanılmıştır. GeoJSON sözel verilerle birlikte geometrik verileri içeren JSON formatlı dosyaya verilen isimdir. Shape uzantlı dosyaların Google Haritalar kütüphanesinde gösterilmesi amacıyla shape dosyasından GeoJSON'a dönüşüm işlemi gerçekleştirilmiştir. (URL – 18) GeoJSON'a dönüştürülen veri seti harita üzerinde gösterilmiştir.

Şekil 5.14: GeoJSON formatındaki parsel bilgilerinin haritada gösterimi.

5.3 Analizlerin Bilimsel Temelleri

Uygulamanın ana işlevlerinden biri olan coğrafi koordinatlardan Kartezyen koordinatlara geçiş işlemi aşağıdaki formüller yardımıyla yapılmıştır. 1:25000 ve küçük ölçekli harita üretiminde UTM, 1:5000 ve daha büyük ölçeklerde ise Gauss-Krüger projeksiyonları kullanılmaktadır. Coğrafi koordinatlardan kartezyen koordinatlara geçiş sağlanması için öncelikle meridyen yay uzunluğu hesaplanmaktadır. (Bildirici, 2012)

Referans elipsoidinin a ve b eksenleri ya da a ekseni ve basıklık verilmesi durumunda birinci ve ikinci eksantirisite parametreleri hesaplanır.

Basıklık eksen ilişkisi:

$$f = \frac{a-b}{a}$$

Birinci Eksantrisite:

$$e = \sqrt{\frac{a^2-b^2}{a^2}}$$

İkinci Eksantrisite:

$$e' = \sqrt{\frac{a^2-b^2}{b^2}}$$

a,b ve e' parametrelerine göre (ikinci eksantrisite kullanılarak) meridyen yay uzunluğu hesabı için ilk olarak aşağıdaki katsayılar hesaplanır.

$$\alpha = \frac{a^2}{b}\left(1-\frac{3}{4}e'^2+\frac{45}{64}e'^4-\frac{175}{256}e'^6+\frac{11025}{16384}e'^8\right)$$

$$\beta = -\frac{a^2}{2b}\left(\frac{3}{4}e'^2-\frac{15}{16}e'^4+\frac{525}{512}e'^6-\frac{2205}{2048}e'^8\right)$$

$$\gamma = \frac{a^2}{4b}\left(\frac{15}{64}e'^4-\frac{105}{256}e'^6+\frac{2205}{4096}e'^8\right)$$

$$\delta = \frac{a^2}{6b}\left(\frac{35}{512}e'^6-\frac{215}{2048}e'^8\right)$$

Meridyen yay uzunluğu hesaplanan katsayılarla aşağıdaki gibi bulunur:

$$B = \alpha\varphi+\beta\sin(2\varphi)+\gamma\sin(4\varphi)+\delta\sin(6\varphi)$$

Şekil 5.15: Meridyen yay uzunluğu hesabı (Bildirici, 2012).

Meridyen yayı hesaplandıktan sonra Kartezyen koordinatların hesaplanması.(Ek A.9)

Gauss-Krüger koordinatları,

$$\eta^2 = e'^2\cos^2\varphi \qquad N = \frac{a^2}{b\sqrt{1+\eta^2}} \qquad \Delta\lambda = \lambda-\lambda_0$$

olmak üzere aşağıdaki gibi hesaplanır. λ_0 dilim orta meridyenini göstermektedir.

$$x_g = B+\frac{N}{2}\cos^2\varphi\tan\varphi\Delta\lambda^2+\frac{N}{24}\cos^4\varphi\tan\varphi(5-\tan^2\varphi+9\eta^2)\Delta\lambda^4$$

$$y_g = N\cos\varphi\Delta\lambda+\frac{N}{6}\cos^3\varphi(1-\tan^2\varphi+\eta^2)\Delta\lambda^3+\frac{N}{120}\cos^5\varphi(5-18\tan^2\varphi+\tan^4\varphi)\Delta\lambda^5$$

Üç derecelik dilimde koordinatlar:

$$X_{3drc} = x_g \qquad Y_{3drc} = y_g+500000m$$

Altı derecelik dilimde koordinatlar (UTM koordinatları):

$$X_{6drc} = x_g * m_0 \qquad Y_{6drc} = y_g * m_0+500000m$$

Burada m_0 küçültme faktörü olup, 0.9996 olarak alınır.

Şekil 5.16: Coğrafi koordinatlardan Gauss-Krüger koordinatlarının hesabı (Bildirici, 2012)

Hesaplamalarda kullanılan referans elipsoidlerinin boyutları da aşağıda gösterilmiştir

Elipsoit	***a***	***b***	***f***
Hayford 1924 (ED50)	6378388	6356911.9461	1/297.0
WGS84	6378137	6356752.314	1/298.257223563
GRS80	6378137	6356752.298	1/298.257
Bessel 1841	6377397.1550	6356078.9632	1/299.1528
Clarke 1880	3678249.145	6356514.990	1/293.466

Şekil 5.17: Yaygın kullanılan bazı referans elipsoitlerinin boyutları (Bildirici, 2012)

6. SONUÇ VE ÖNERİLER

Tezin sonucunda akıllı telefonlarında Android işletim sistemi kullanan kullanıcıların %95.2'sine hitap eden bir mobil uygulama geliştirilmiştir. Geliştirilen bu mobil uygulama ile gps uydularından gelen konum ve telefondaki hücresel bilgilerden gelen konum karşılaştırılmış ve en doğuru konum elde edilmiştir. Elde edilen konum bilgisinden anlık gösterilmek üzere 25 binlik, 50 binlik ve 100 binlik pafta bilgilerinin gösterilmesi sağlanmıştır. Anlık konumdan bağımsız olarak kullanıcının haritada tıkladığı noktaya ait pafta bilgileri de gösterilmiştir. Ayrıca gösterilen pafta bilgilerinin kaydedilmesi ve kaydedilen paftaya ait not eklenmesi sağlanmıştır.

Belirlenen pafta bilgilerinin solt alt ve sağ üst köşe koordinatları bilgisi kullanılarak harita üzerinde pafta çizdirilmiştir. Böylece kullanıcının paftayı görsel olarak görmesine olanak sağlanmıştır.

GeoJSON tipinde bulunan parsel verilerinin haritaya eklenmesi ile ilgili Zeytinburnu ilçesine ait bilgiler kullanılmıştır fakat ilgili veri seti parsel bilgisi içerdiğinden ayrıntı düzeyi artmaktadır. Ayrıntı düzeyi artan verinin GeoJSON'a dönüştürülmesi esnasında koordinat verilerinin fazlalığından dolayı parsel bilgilerinin haritada gösterilmesi uzun sürmektedir. Zeytinburnu ilçesinin bir bölümüne ait çalışma yapılığında ise verinin boyutu azaldığından haritada gösterilmesi kısa sürmektedir. Bu nedenle uygulamadan performans alınması için GeoJSON verilerinin optimize edilerek haritaya eklenmesi gerekmektedir.

3 derecelik ve 6 derecelik koordinat sistemleri arasında dönüşüm yapılarak kullanıcının istediği tipte bilgi gösterilmiştir. Aynı zamanda altlık haritaların değiştirilmesi ve belirlenen pafta bilgilerinin sosyal platformlarda paylaşılmasına olanak sağlanmıştır.

Gelişen teknoloji ve akıllı telefon kapasiteleri ile konumsal anlamda daha hassas verilerle işlem yapılmasıyla birlikte kullanıcılara daha doğru bilgi sunulacaktır. Geliştirilen mobil uygulama ile alanında uzman kişilerin nokta keşifleri esnasında anlık konumdan pafta bilgilerine erişim işlemi kolaylaştırılmıştır.

Tasarımsal anlamda kullanıcı deneyimine uygun geliştirilen mobil uygulama ile dönüşüm işlemi çok kısa sürede ve kolaylık ile yapabilmektedir.

7. KAYNAKLAR

Elmalı, Ş. **(**2015). *Karekod Tabanlı Gıda İçerik Kontrolüne Yönelik Android Uygulaması*, Sakarya Üniversitesi Fen Bilimleri Enstütüsü Yüksek Lisans Tezi.

İğci, E **(**2014). *Android İşletim Sistemi ile Çalışan Mobil Uygulama Geliştirilmesi*, Ege Üniversitesi Fen Bilimleri Enstütüsü Yüksek Lisans Tezi

Uçar, D; İpbüker C.; Bildirici, Ö **(**2012). *Matematiksel Kartografya*

Ulaghanatan, M. N., (2016). *Building a Volunteered Geographic Information System* (VGIS): A Mobile Application for Disaster Management., California State University

Gonchikara, L (2014). *An Android Application for Crime Analysis in San Diego*, San Diego State University ., Master of Science Thesis

Choudhury, M (2016). *Android Application for SDSU Faculty/Student Safety*, San Diego State University, Master of Scicence Thesis.

Mokashi, P., N. (2012). *NewsMap On The Go, San Diego State University*, Master of Science Thesis.

Url-1 <*https://play.google.com/store/apps/details?id=com.kocaman*>,alındığı tarih: 04.08.2017.

Url-2 <*https://play.google.com/store/apps/details?id=com.dissipatedheat.kortrans*>, alındığı tarih: 04.08.2017.

Url-3 <*https://play.google.com/store/apps/details?id=com.tkgm.parselsorgu*>, alındığı tarih: 04.08.2017.

Url-4 <*https://play.google.com/store/apps/developer?id=Süleyman+ER*>,alındığı tarih: 04.08.2017.

Url-5 <*https://tr.wikipedia.org/wiki/Android_(i%C5%9Fletim_sistemi)*>,alındığı tarih: 20.04.2017.

Url-6 <*https://developer.android.com/about/dashboards/index.html*>, alındığı tarih: 29.08.2017.

Url-7 <*https://blog.jetbrains.com/kotlin/2017/05/kotlin-on-android-now-official/*>, alındığı tarih: 05.07.2017.

Url-8 <*https://developer.android.com/guide/index.html*>, alındığı tarih: 20.04.2017.

Url-9 <*https://www.mapbox.com/android-docs/map-sdk/overview/*>, alındığı tarih: 05.07.2017.

Url-10 < *https://github.com/osmdroid/osmdroid*>, alındığı tarih: 05.07.2017.

Url-11 <*https://developers.arcgis.com/android/*>, alındığı tarih: 05.07.2017.

Url-12 <*https://developers.arcgis.com/arcgis-runtime/choosing-the-right-version/*>, alındığı tarih: 05.07.2017.

Url-13 <*https://developers.google.com/maps/documentation/android-api/*>, alındığı tarih: 05.07.2017.

Url-14 <*https://uxplanet.org/perfect-bottom-navigation-for-mobile-app-ffabbb98c0f*>, alındığı tarih: 05.07.2017.

Url-15 <*https://developer.android.com/guide/topics/ui/declaring-layout.html*>, alındığı tarih: 05.07.2017.

Url-16 <*https://developer.android.com/guide/topics/location/strategies.html*>, alındığı tarih: 05.07.2017.
Url-17 <*http://www.yildiz.edu.tr/~hekim/JEODEZI12.pdf*>, alındığı tarih: 05.07.2017.
Url-18 <*https://ogre.adc4gis.com*>, *shape to geojson* alındığı tarih: 04.08.2017.
Url-19 <*https://play.google.com/store/search?q=koordinat%20dönüşümü&c=apps*>, alındığı tarih: 04.08.2017.
Url-20 <*https://play.google.com/store/search?q=pafta&c=apps*>,alındığı tarih: 04.08.2017.
Url-21 <*https://play.google.com/store/search?q=pafta%20bul&c=apps*>,alındığı tarih: 04.08.2017.

8. EKLER

EK A.1 : isBetterLocation metodu

```
/**
 * Yeni belirlenen konumun bir öncekine göre daha iyi mi olup olmadığını anlayan kod.
 *
 * @param location          Değerlendirilmek istenen yeni konum
 * @param currentBestLocation Karşılaştırılacak konum
 */
protected boolean isBetterLocation(Location location, Location currentBestLocation) {
    if (currentBestLocation == null) {
        // Yeni gelen konum bilgisi eğer hiç konum bilgisi yoksa en iyi konumdur.
        return true;
    }

    // Konum düzeltmesinin yeni mi eski mi olduğunun kontrolü.
    long timeDelta = location.getTime() - currentBestLocation.getTime();
    boolean isSignificantlyNewer = timeDelta > TWO_MINUTES;
    boolean isSignificantlyOlder = timeDelta < -TWO_MINUTES;
    boolean isNewer = timeDelta > 0;

    //Geçerli konumdan bu yana iki dakikadan fazla bir süre geçtiyse, yeni konumu kullanılır
    if (isSignificantlyNewer) {
        return true;
        // Yeni yer iki dakikadan daha eski ise, konum belirlemesi daha kötü olmalı
    } else if (isSignificantlyOlder) {
        return false;
    }

    //Yeni konum düzeltmenin daha fazla veya daha az doğru olup olmadığı kontrol edilir
    int accuracyDelta = (int) (location.getAccuracy() - currentBestLocation.getAccuracy());
    boolean isLessAccurate = accuracyDelta > 0;
    boolean isMoreAccurate = accuracyDelta < 0;
    boolean isSignificantlyLessAccurate = accuracyDelta > 200;

    //Eski ve yeni konum bilgisinin aynı sapğlayıcından mı olduğu belirlenir.
    boolean isFromSameProvider = isSameProvider(location.getProvider(),
            currentBestLocation.getProvider());

    // Geçerlilik ve doğruluk kombinasyonu kullanarak yer kalitesini belirlenir
    if (isMoreAccurate) {
        return true;
    } else if (isNewer && !isLessAccurate) {
        return true;
    } else if (isNewer && !isSignificantlyLessAccurate && isFromSameProvider) {
        return true;
    }
    return false;
}

/**
```

```
 * İki sağlayıcının aynı olup olmadığı kontrol edilir.
 */
private boolean isSameProvider(String provider1, String provider2) {
    if (provider1 == null) {
        return provider2 == null;
    }
    return provider1.equals(provider2);
}
```

EK A.2 : Google kütüphanesinin konum ikonunu değiştirmek için yazılmış metod.

```
public static void changeLocationButton(SupportMapFragment mapFragment) {
    ImageView locationButton = (ImageView)
mapFragment.getView().findViewById(Integer.parseInt("2"));
    int dp10 = dpToPx(10);
    int dp25 = dpToPx(25);
    int dp50 = dpToPx(50);
    RelativeLayout.LayoutParams rlp = (RelativeLayout.LayoutParams)
locationButton.getLayoutParams();
    rlp.addRule(RelativeLayout.ALIGN_PARENT_TOP, 0);
    rlp.addRule(RelativeLayout.ALIGN_PARENT_BOTTOM, RelativeLayout.TRUE);
    rlp.height = dp50;
    rlp.width = dp50;
    locationButton.requestLayout();
    rlp.setMargins(dp25, dp25, dp25, dp25);
    locationButton.setBackgroundResource(R.drawable.drw_circle);
    locationButton.setImageResource(R.drawable.ic_location);
    locationButton.setScaleType(ImageView.ScaleType.FIT_CENTER);
    locationButton.setPadding(dp10, dp10, dp10, dp10);
    locationButton.setLayoutParams(rlp);
}
```

EK A.3 : 3 ve 6 derecelik dilimlerde meridyen bilgisinden dilim orta meridyenine ulaşmayı sağlayan metod.

```
/**
 * Dilim orta meridyeni bulma fonksiyonu
 *
 * @param slice : 3 veya 6 derecelik olarak tip verilir
 * @param lonn : Meridyen bilgisi
 * @return : dilim orta meridyenini döndürür.
 */
private int calcMiddleMeridian(String slice) {
    int midleMeridian = 0;
    switch (slice) {
        case "3":
            midleMeridian = (3 * ((int) ((lonn + 1.5) / 3)));
            break;
        case "6":
            midleMeridian = (6 * ((int) (lonn / 6)) + 3);
            break;
    }
    return midleMeridian;
```

EK A.4 : Pafta bilgilerinin 2 boyutlu dizide tutlmasını sağlayan metod

```
private static String[][] pafta250;
static String[][] paftainit() {
   pafta250 = new String[7][];
   int i, j;
   for (i = 0; i < 7; i++) {
      pafta250[i] = new String[13];
   }
   for (i = 0; i < 7; i++) {
      for (j = 0; j < 13; j++) {
         pafta250[i][j] = "_";
         pafta250[0][0] = "EDİRNE";
         pafta250[0][1] = "KIRKLARELİ";
         pafta250[0][2] = "İSTANBUL";
         pafta250[0][3] = "EREĞLİ";
         pafta250[0][4] = "ZONGULDAK";
         pafta250[0][5] = "KASTAMONU";
         pafta250[0][6] = "SİNOP";
         pafta250[0][7] = "SAMSUN";
         pafta250[0][8] = "PERŞEMBE";
         pafta250[0][9] = "AKÇAABAT";
         pafta250[0][10] = "ARTVİN";
         pafta250[0][11] = "ARDAHAN";
         pafta250[0][12] = "TİFLİS";
         pafta250[1][0] = "ÇANAKKALE";
         pafta250[1][1] = "BANDIRMA";
         pafta250[1][2] = "BURSA";
         pafta250[1][3] = "ADAPAZARI";
         pafta250[1][4] = "BOLU";
         pafta250[1][5] = "ÇANKIRI";
         pafta250[1][6] = "ÇORUM";
         pafta250[1][7] = "TOKAT";
         pafta250[1][8] = "GİRESUN";
         pafta250[1][9] = "TRABZON";
         pafta250[1][10] = "TORTUM";
         pafta250[1][11] = "KARS";
         pafta250[1][12] = "ERİVAN";
         pafta250[2][0] = "AYVALIK";
         pafta250[2][1] = "BALIKESİR";
         pafta250[2][2] = "KÜTAHYA";
         pafta250[2][3] = "ESKİŞEHİR";
         pafta250[2][4] = "ANKARA";
         pafta250[2][5] = "KIRŞEHİR";
         pafta250[2][6] = "YOZGAT";
         pafta250[2][7] = "SİVAS";
         pafta250[2][8] = "DİVRİĞİ";
         pafta250[2][9] = "ERZİNCAN";
         pafta250[2][10] = "ERZURUM";
         pafta250[2][11] = "AĞRI";
         pafta250[2][12] = "DOĞUBEYAZIT";
         pafta250[3][0] = "URLA";
         pafta250[3][1] = "İZMİR";
         pafta250[3][2] = "UŞAK";
         pafta250[3][3] = "AFYON";
         pafta250[3][4] = "ILGIN";
         pafta250[3][5] = "AKSARAY";
         pafta250[3][6] = "KAYSERİ";
         pafta250[3][7] = "ELBİSTAN";
         pafta250[3][8] = "MALATYA";
```

```
            pafta250[3][9] = "ELAZIĞ";
            pafta250[3][10] = "MUŞ";
            pafta250[3][11] = "VAN";
            pafta250[3][12] = "BAŞKALE";
            pafta250[4][0] = "SAMOS";
            pafta250[4][1] = "AYDIN";
            pafta250[4][2] = "DENİZLİ";
            pafta250[4][3] = "ISPARTA";
            pafta250[4][4] = "KONYA";
            pafta250[4][5] = "KARAMAN";
            pafta250[4][6] = "ADANA";
            pafta250[4][7] = "GAZİANTEP";
            pafta250[4][8] = "ŞANLI URFA";
            pafta250[4][9] = "DİYARBAKIR";
            pafta250[4][10] = "MARDİN";
            pafta250[4][11] = "CİZRE";
            pafta250[4][12] = "HAKKARİ";
            pafta250[5][0] = "KALİMONOS";
            pafta250[5][1] = "MARMARİS";
            pafta250[5][2] = "FETHİYE";
            pafta250[5][3] = "ANTALYA";
            pafta250[5][4] = "ALANYA";
            pafta250[5][5] = "SİLİFKE";
            pafta250[5][6] = "MERSİN";
            pafta250[5][7] = "ANTAKYA";
            pafta250[5][8] = "SURUÇ";
            pafta250[5][9] = "CEYLANPINAR";
            pafta250[5][10] = "HASEC";
            pafta250[5][11] = "MUSUL";
            pafta250[5][12] = "ERBİL";
            pafta250[6][6] = "LAZKIYE";
            pafta250[6][7] = "HAMA";
        }
    }

    return pafta250;
}

}
```

EK A.5 : Enlem ve boylam bilgilerinden 250.000'lik pafta bilgilerine erişim.

```
private static String paftaFrom250k(Double ph, Double lm) {
    Double ph0 = 42.0, lm0 = 25.5, isa, isu;
    //İstisnai durumlar...
    if (ph > 42. && ph <= 42.125)
        if ((lm >= 27 && lm <= 28.5) || (lm >= 33. && lm <= 36.))
            ph = 41.999;
    if (ph >= 35. && ph <= 42. && lm >= 25.5 && lm <= 45.) {
        isa = Math.floor(Math.abs(ph - ph0));
        isu = Math.floor(2 * Math.abs(lm - lm0) / 3);
        if (isu < 13 && isa < 7)
            return pafta250[isa.intValue()][isu.intValue()];
        else
            return "_";

    } else
        return "_";
}
```

EK A.6 : Koordinat dönüşümü hesaplamaları

```
private Parameter param;
//Enlem ve boylam degiskenleri
private double latt;
private double lonn;
private double m0 = 0.9996;
//parametrelere gore ikinci eksantirisite
private double f;
private double e2; // Birinci dış merkezlilik
private double t;
private double n;
//Meridyen yay uzunlugu icin gerekli olan parametreler
private double a0;
private double a2;
private double a4;
private double a6;
private double a8;
//Meridyen uzunlugu
private double G;
//Egiklik yarıçapı
private double n2;
private double N;
//Dilim orta meridyeni
public double smm;
//Dilim orta meridyeni ile Meridyen farkı
private double w;

/**
 * @param lat   : Enlem
 * @param lon   : Boylam
 * @param ctype : Koordinat sistemi tipi
 * @param slice : Dilim tipi (3 veya 6 diye verilebilir)
 */

public Calculation(Double lat, Double lon, String ctype, String slice) {
    this.latt = Math.toRadians(lat);
    this.lonn = lon;
    this.param = getParameter(ctype);
    this.f = calcf();
    this.e2 = calce2();
    this.N = calcN();
    this.n2 = calcNuSquare();
    this.t = Math.tan(latt);
    this.n = calcn();


    this.a0 = calcA0();
    this.a2 = calcA2();
    this.a4 = calcA4();
    this.a6 = calcA6();
    this.a8 = calcA8();

    this.G = calcMeridianLength();
    this.smm = calcMiddleMeridian(slice);
    this.w = calcMeridDiff();


}
```

```
//verilen elipsoidin adina gore parametrelerini donduren metod
private Parameter getParameter(String ctype) {
    Parameter p = new Parameter();
    switch (ctype) {

        case "ed50":
            p.setA(6378388.0);
            p.setB(6356911.9461);
            p.setF(1.0 / 297.0);
            break;
        case "wgs84":
            p.setA(6378137.0);
            p.setB(6356752.3142);
            p.setF(1.0 / 298.257223563);
            break;
        case "grs80":
            p.setA(6378137.0);
            p.setB(6356752.298);
            p.setF(1.0 / 298.257);
            break;
        case "bessel1841":
            p.setA(6377397.1550);
            p.setB(63576078.9632);
            p.setF(1.0 / 299.1528);
            break;
        case "clarke1880":
            p.setA(3678249.145);
            p.setB(6356514.990);
            p.setF(1.0 / 293.466);
            break;
    }
    return p;
}


//Elipsoidal basıklık
private double calcf() {
    double firstValue = ((param.getA() - param.getB()) / param.getA());
    return firstValue;
}

//birinci eksantrisite - (e2) hesabi
private double calce2() {
    double firstValue = ((Math.pow(param.getA(), 2) - Math.pow(param.getB(), 2)) /
(Math.pow(param.getA(), 2)));
    return firstValue;
}


//hesaplama icin buyukluk N : capraz egrilik yarıçapıu
private double calcN() {
    double c1 = (Math.pow(Math.sin(latt), 2));
    double c2 = e2 * c1;
    double c3 = Math.sqrt(1.0 - c2);
    return param.getA() / c3;
}


//egiklik yaricapi nukare
```

```
private double calcNuSquare() {
    double c1 = (Math.pow(Math.cos(latt), 2));
    double c2 = 1 - e2;
    double c3 = e2 / c2;
    return c1 * c3;
}


//Yardımcı büyüklük
private double calcn() {
    double c1 = 2 - f;
    return f / c1;
}



 /*
* a, b ve eksantrisite - (Eu) değerlerini kullanarak meridyen yay uzunlugu hesaplamasi
* Öncelikle katsayilar hesaplanacak
* */

//a0 değeri
private double calcA0() {
    double x1 = Math.pow(n, 2) / 4.0;
    double x2 = Math.pow(n, 4) / 64.0;
    return 1 + x1 + x2;
}

//a2 degeri
private double calcA2() {
    double x1 = Math.pow(n, 3) / 8.0;
    double x2 = n - x1;
    double x3 = 3.0 / 2.0;
    return x2 * x3;
}

//a4 degeri
private double calcA4() {
    double x1 = Math.pow(n, 4) / 4.0;
    double x2 = Math.pow(n, 2) - x1;
    double x3 = 15.0 / 16.0;
    return x2 * x3;
}

//a6 degeri
private double calcA6() {
    double x1 = 35.0 / 48.0;
    double x2 = Math.pow(n, 3);
    return x1 * x2;
}


//a8 degeri
private double calcA8() {
    double x1 = 315.0 / 512.0;
    double x2 = Math.pow(n, 4);
    return x1 * x2;
}
```

```
//ikinci eksantrisite - (Eu) hesabi
private double calcEu() {
    double firstValue = ((Math.pow(param.getA(), 2) - Math.pow(param.getB(), 2)) /
(Math.pow(param.getB(), 2)));
    Log.d(TAG, "Calculation ikinci eksntrisite -  e' - " + Math.sqrt(firstValue));
    return Math.sqrt(firstValue);
}


/**
 * enlem bilgisinden meridyen yayi uzunlugu
 *
 * @return : meridyen yayi uzunlugu
 */
private double calcMeridianLength() {
    double x1 = param.getA() / (1 + n);
    double x2 = a0 * latt;
    double x3 = a2 * Math.sin(2 * latt);
    double x4 = a4 * Math.sin(4 * latt);
    double x5 = a6 * Math.sin(6 * latt);
    double x6 = a8 * Math.sin(8 * latt);
    return x1 * (x2 - x3 + x4 - x5 + x6);
}
```

EK A.7 : Koordinat dönüşümü anahtar kelimesi ile yapılan arama neticesinde çıkan uygulama sonuçları (URL – 19)

Uygulama Adı	Yayıncı	Son Güncelleme Tarihi	İndirilme	Mevcut Sürüm	Değerlendirme Oyu (5 üzerinden)
Coordinator - Koordinat Topla	Süleyman Er	31 Mayıs 2017	10.000 - 50.000	3.9	4.3
Kocaman	Hakan KOCAMAN & Ümit Tacettin AKGÖL	8 Haziran 2017	50.000 - 100.000	2.5	4.6
KorTrans Pafta	Kerem KAT	24 Aralık 2011	10.000 - 50.000	0.7	4.2
Coordinate Converter+	TennyApps	20 Şubat 2017	10.000 - 50.000	2.18.31	4.4
Coordinate converter: WGS84 and local systems	PillaBlu	23 Mayıs 2017	1.000 - 5.000	1.0.1.1	3.7
Coordinate Converter	Innovative Concepts, Inc.	1 Ekim 2014	50.000 - 100.000	1.5	4.2
AliCo	Ali Güvenaltın	12 Haziran 2017	50 - 100	1.4	4.8
iConv Ads: Converti Coordinate	GraphMouse	5 Temmuz 2015	10.000 - 50.000	3.10	4.3
Mobile Topographer Free	S.F. Applicality Ltd.	18 Mayıs 2017	100.000 - 500.000	8.0.6	4.3
iConv Ads: Converti Coordinate	GraphMouse	5 Temmuz 2015	500 - 1.000	4.10	4.1

EK A.8 : Pafta anahtar kelimesi ile yapılan arama neticesinde çıkan uygulama sonuçları – (URL - 20)

Uygulama Adı	Yayıncı	Son Güncelleme Tarihi	İndirilme	Mevcut Sürüm	Değerlendirme Oyu (5 üzerinden)
TKGM Parsel Sorgu	Tapu ve Kadastro Genel Müdürlüğü	26 Ocak 2017	500.000 - 1.000.000	2.1.7	3.9
KorTrans Pafta	Kerem KAT	24 Aralık 2011	10.000 - 50.000	0.7	4.2
İmar ve Parsel Sorgulama	imarbox	5 Temmuz 2017	100.000 - 500.000	New.Borders	3.9
Emlak Ajandam	Emlak Ajandam	5 Mart 2017	10.000 - 50.000	-	4.4
Gps Area Calculator	KBK INFOSOFT	18 Ekim 2016	500.000 - 1.000.000	1.2	4.3
Canlı Haritalar Sokak Görünümü	Nermo	18 Aralık 2016	1.000.000 - 5.000.000	3.0	4.0
Bilgimap Pro	Bilgimap Ltd. Sti.	18 Kasım 2016	10.000 - 50.000	3.0	3.8

EK A.9 : Pafta bul anahtar kelimesi ile yapılan arama neticesinde çıkan uygulama sonuçları (URL – 21)

Uygulama Adı	Yayıncı	Son Güncelleme Tarihi	İndirilme	Mevcut Sürüm	Değerlendirme Oyu (5 üzerinden)
KorTrans Pafta	Kerem KAT	24 Aralık 2011	10.000 - 50.000	0.7	4.2

ÖZGEÇMİŞ

Ad Soyad: Salih YALÇIN

Doğum Yeri ve Tarihi: Altındağ/ANKARA – 03.03.1989

Adres: Çobançeşme Mah. Badem Sok. No:9 D:10 Bahçelievler/İSTANBUL

E-Posta: yalcins@itu.edu.tr

Lisans: Yıldız Teknik Üniversitesi – Harita Mühendisliği

Mesleki Deneyim ve Ödüller:

2014 – … - İstanbul Büyükşehir Belediyesi Android Uygulama Geliştiricisi

Yayın ve Patent Listesi: