T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MOBİL CİHAZLAR ÜZERİNE BİR UYGULAMA: EBEVEYN ÇOCUK TAKİP PROGRAMI

YÜKSEK LİSANS TEZİ

Hacer BAYIROĞLU

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ

Tez Danışmanı : Yrd. Doç. Dr. Kürşat AYAN

Ağustos 2013

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

MOBİL CİHAZLAR ÜZERİNE BİR UYGULAMA: EBEVEYN ÇOCUK TAKİP PROGRAMI

YÜKSEK LİSANS TEZİ

Hacer BAYIROĞLU

Enstitü Anabilim Dalı : BİLGİSAYAR VE BİLİŞİM MÜHENDİSLİĞİ

Bu tez 16/ 08 /2013 tarihinde aşağıdaki jüri tarafından Oybirliği ile kabul edilmiştir.

Yrd. Doç. Dr. Kürşat AYAN	Doç. Dr. Bayram TOPAL	Yrd. Doç. Dr. Murat İSKEFİYELİ
Jüri Başkanı	Üye	Üye

TEŞEKKÜR

Bu konu üzerinde ileriki zamanlarda, çalışma ve araştırma yapmak isteyecek arkadaşlara büyük yardımı olacağına inandığım bu projeyi hazırlarken bana her konuda yardım eden, desteğini hiç esirgemeyen, değerli fikirleriyle yol gösteren değerli danışman hocam Yrd. Doç Dr. Kürşat AYAN' a ve değerli hocam Yrd. Doç. Dr. Murat İSKEFİYELİ, çalışmaya doğrudan ya da dolaylı olarak katkıda bulunmuş tüm arkadaşlarıma ve hayatımın her döneminde desteklerini esirgemeyen aileme teşekkürlerimi sunarım.

İÇİNDEKİLER

SİMGELER VE KISALTMALAR LİSTESİ

ADT	: Android Development Tool (Android Geliştirme Araçları)
AWT	:Abstract Window Toolkit (Kullanıcı Arayüzü Araç Kütüphanesi)
API	:Application Programming Interface (Uygulama Programlama Arayüzü)
ASP	: Active Server Pages (Aktif Sunucu Sayfaları)
AVD	: Android Virtual Device (Android Sanal Makine)
C/A	: Coarse/Acquisition (İşlenmemiş/Veri İzleme)
DGPS	:Diferantial Global Positioning System (Diferansiyel Yer Belirleme Sistemi)
DVM	: Dalvik Virtual Machine (Dalvik Sanal MAkine)
GPRS	: General Packet Radio Service (Genel Paket Radyo Servisi)
GPS	: Global Positioning System (Küresel Konumlama Sistemi)
GPL	: General Public License (Genel Kamu Lisansı)
HTTP	: Hypertext Transfer Protocol (Hipermetin Aktarım Protokolü)
IBM	: International Business Machines (Uluslararası İş Makineleri)
ICCRD	: 2011 Computer Research and Development
ID	: Identity (Kimlik)
IDE	: Integrated Development Environment (Bütünleşmiş Geliştirme Ortamı)
IMEI	:International Mobile Equipment Identity (Uluslararası Mobil Cihaz Kimliği)
IPC	: Interprocess Communication (İşlemler Arası İletişim)
J2ME	: Java 2 Micro Edition (Java 2 Mikro Basım)
JDK	: Java Development Kit (Java Gelistirme Teçhizati)
JPO	: Joint Program Office (Ortak Program Ofisi)

JSON	: Java Script Object Notation (Java Betik Nesne Gösterimi)
JSP	: Java Server Page (Java Sunucu Sayfası)
JVM	: Java Virtual Machine (Java Sanal Makinası)
MAC	: Mandatory Access Control (Zorunlu Erişim Denetimi)
MYSQL	: My Structured Query Language (My Yapılı Sorgulama Dili)
NMEA	: National Marine Electronics Association (Ulusal Denizcilik Elektroniği Birliği)
NNSS	: Navy Navigational Satellite System (Deniz Kuvvetleri Seyir Uydu Sistemi)
NAVSTAR	: Navgation Satellite Timing And Ranging (Yöngüdüm Uydu Zamanlama ve Konumlayıcı)
OHA	: Open Handset Alliance (Açık Mobil Cihazlar Platformu)
PHP	: Hypertext Preprocessor (Hipermetin Önislemcisi)
REST	: Representational State Transfer - Temsili Durum Transferi
RPC	: Remote Procedure Call (Uzaktan Yordam Çağrısı)
SOAP	: Simple Object Access Protocol (Basit Nesne Erişim Protokolü)
SQL	: Structured Query Language (Yapılı Sorgulama Dili)
UTC	: Universal Time Coordinated (Koordine Edilmiş Evrensel Zaman)
UHF	: Ultra High Frequency (Oldukça Yüksek Frekans)
XML	: Extensible Markup Language (Genişletilebilir İşaretleme Dili)
WSDL	: Web Service Description Language (Web Servis Tanımlama Dili)
W3C	: World Wide Web Consortium (Dünya Çapında Ağ Birliği)

ŞEKİLLER LİSTESİ

ÖZET

Anahtar kelimeler: Konum Belirleme Hizmeti, GPS(Global Positioning System), Android.

GPS sisteminin ilk kuruluş hedefi tamamen askeri amaçlar içindi. GPS alıcıları yön bulmakta, askeri çıkartmalarda ve roket atışlarında kullanılmak üzere tasarlanmıştır. Ancak, 1980'lerde GPS sistemi sivil kullanıma da açılmıştır. Artık birçok alanda hayati önem taşıyan bir araç olarak kullanıma girmiştir. Ve günümüz de bir çok mobil telefonlarda GPS modülü bulunmaktadır. GPS size bulunduğunuz yerleri işaretleme ve belirlediğiniz noktaya geri dönme imkanı sağlar.

Android İşletim Sistemi Google firması tarafından yayınlanan yeni nesil bir mobil telefon platformudur. Android İşletim Sistemi bir uygulamaya, konum bilgisini sağlamak için farklı metodlar sunar. Bu çalışmada Android İşletim Sisteminin sunduğu konum hizmetleri kullanılarak bir çocuk takip yazılımı geliştirilmiştir.

AN APPLICATION ON MOBILE DEVICES: PARENT CHILD TRACKING PROGRAM

SUMMARY

Key words: Location Based Services, GPS (Global Positioning System), Android

At first the GPS system was made solely for military purposes. GPS recievers were designed to find direction, to be used in military extractions and rocket fire. But in the 1980's GPS was opened to the use of civilians. It's now in use as a very important tool in many fields. And nowadays many mobile phones have a GPS module. GPS enables you to mark your current location and return to the marked location.

The Android Operating System is a new generation mobile phone platform released by Google. Android provides different methods to provide location data to an application. In this study using the location services Andoid provides, a child tracking program has been developed.

BÖLÜM 1. GİRİŞ

Günlük yaşamda karşılaşılan olumsuz sebepler ve çevresel şartlar ebeveynlerin çocuklarını sürekli gözetim altında tutmak istemelerine neden olmaktadır. Türkiye İstatistik Kurumu'nun 81 ilde yaptığı araştırmaya göre 2008-2011 yılları arasında kaybolan çocuk sayısı 27 bini geçmiş bulunmaktadır.

Çocuk güvenliği gittikçe artan bir sorun olmakla birlikte, bu sorun teknolojiyle aşılabilir. Uygun bir yazılımla Android konum bilgisi hizmetleri kullanarak ebeveynler hemen her an neredeyse her yerden çocuklarının bulundukları yeri öğrenebilirler.

Teknoloji geliştikçe insanoğlunun hayatı da kolaylaşmaya devam ediyor. Ama aynı hızda çocuklar üzerinde kontrol sahibi olmak da bir o kadar zorlaşıyor.

Şu an birçok aile gönüllü olarak olmasa bile çocuklarına cep telefonu almak durumunda kalıyor. Piyasada Android İşletim sistemine sahip birçok firmanın her bütçeye uygun fiyatlardan başlayarak geniş ürün yelpazesine sahip telefonları bulunmaktadır. Ve bu telefonların çoğunda GPS modülü bulunmaktadır. GPS yani Küresel Yer Belirleme Sistemi sadece sürücülerin, araçlarını bilmedikleri yollarda daha rahat kullanmalarını sağlamaktan ibaret değil güvenlik aracı olarak kullanılabilecek son derece faydalı bir teknolojidir. Gerçekleştirilen bu uygulamada aileler GPS yardımıyla çocuklarını an ve an takip edebilecekler ve çocuğa ulaşamadıkları acil bir durum olursa yerini belirleyebileceklerdir.

Çocuk takip konusu birçok özel firmanın ilgi alanına girmesinin yanı sıra problemleri birçok akademik çalışmaya da konu olmuştur. Aşağıda bu çalışmalarla alakalı bilgiler verilmiştir.

[1] numaralı akademik çalışmada, kullanıcının geçerli konumu veren bir konum farkındalık sistemi anlatılmıştır. Java'yı destekleyen telefonlar için yapılmış bir çalışmadır. Bunun için mobil uygulama kısmı J2ME platformunda yazılmıştır. Çalışmada konum verisini alabilmek için Küresel Konumlandırma Sistemi teknolojisi olan GPS kullanılmıştır. Kullanıcının konumu kısa mesaj servisiyle daha önceden belirtilen numaralara gönderilmektedir ve kullanıcılar konumlarını bir web sayfası aracılığıyla Google Map üzerinden ailesiyle ve arkadaşlarıyla paylaşabilmektedir.

[2] numaralı akademik çalışmada, GPS sistemi kullanılarak acil durum senaryosu oluşturulmuştur. Bu çalışmada kullanıcı konumunu enlem ve boylam bilgileriyle birlikte harita üzerinde görebiliyor ve acil bir durum olduğunda önceden belirlediği kişilere isterse ekran görüntüsünü çekip veya sadece konum bilgilerini mesaj veya e-mail atabiliyor.

[3] numaralı akademik çalışmada, yine Android İşletim sistemine sahip bir mobil telefona entegre olmuş GPS sistemi ve veri iletiminde GPRS kullanılarak bir takip sistemi oluşturulmuştur. Bu çalışmada kullanıcılara ait konum verileri GPS sisteminden elde edilip ve o telefona ait IMEI numarası eklenerek MYSQL Server da oluşturulan bir veritabanına GPRS sistemi kullanılarak gönderilmektedir. Java Script Page 'de hazırlanan web sayfasında kullanıcıya ait konum Google MAP aracılığıyla harita üzerinde gösterilmektedir.

[4] numaralı akademik çalışmada, herhangi bir kronik rahatsızlığı olan çocuklar için 24 saat takip edilebilecekleri bir mobil acil durum sistemi oluşturulmuştur. Bu acil durum sisteminde hasta çocuk yardıma ihtiyaç duyduğunda bir butona basarak web servera mesaj atmaktadır. Mesajın içeriğine çocuğun ID' si ve konum verisi otomatik eklenmektedir. C# programlamayla oluşturulmuş olan web sayfasında çocuğun konumu harita üzerinde gösterilmektedir. Ayrıca önceden veri tabanında kayıtlı olan çocuğun ailesine, hastaneye, doktoruna mesaj gitmektedir. Bu çalışmada mobil telefon olarak Windows Phone tercih edilmiştir.

Bu tez çalışmasında ise, GPS sistemi, kablosuz ağ erişim noktaları ve CELL ID (baz istasyonu)' ler aracılığıyla en az hata payıyla konum bilgisi elde edilmeye çalışılmıştır. Çalışmanın web ara yüzünde ebeveyn gerçek zamanlı ve geçmişe dönük çocuğun konumunu görebilmekte, istediği gün ve saate bölge sınırlaması belirleyebilmekte ve sınırları ihlal ettiğinde maille bilgilendirilmektedir. Ayrıca ebeveyn çocuğun telefonuna (nerdesin) mesajı attığında çocuğun o anki konumu adres olarak elde edilebilmektedir.

Bu çalışmada; yapılan çalışmalardan farklı olarak, konum verisini alırken GPS ve Wi-fi kapalı olduğunda, baz istasyonlarından almasıdır. Ayrıca web ara yüzünde ebeveyn çocuğu için sınırlamaları esnek girebilmektedir.

Bu tez çalışması 5 bölümden oluşmaktadır. 2. bölümde GPS (Küresel Konum Belirleme) kavramı ayrıntılı olarak anlatılmıştır. GPS sistemini ne olduğu, bölümleri, uygulama alanları ve çalışma yapısı hakkında detaylı bir bilgi sunulmaktadır. 3. bölümde tezde kullanılan teknoloji ve programlar ayrıntılı olarak ele alınıp, geliştirme ortamlarının kurulumları anlatılmıştır. Tezin 4. bölümünde sistemin gerçekleştirilmesi ayrıntılı olarak ele alınmıştır. Sistemin bölümlerinden ve ayrıca bu bölümleri oluştururken gerçekleştirilen sınıflar anlatılmıştır. 5. bölüm ise tezin sonuç kısmını kapsamaktadır. Bu kısımda uygulamanın bize neler kattığı ve uygulamadan çıkarılan sonuçların neler olduğu anlatılmıştır.

BÖLÜM 2. GPS TEKNOLOJİSİ

Hesaplama tekniği ve uzay çalışmalarındaki hızlı gelişmeler 1980'li yılların en önemli ürünlerinden biri olan GPS (Global Positioning System-Küresel Konumlama Sistemi)' in günlük yaşama girmesine sebep olmuştur. Bu bölümde gerçeğe en yakın konum bilgisinin elde edilmesini sağlayarak çalışmanın temelini oluşturan GPS sistemi hakkında genel bilgiler verilmiştir.

2.1. GPS Nedir?

GPS uydu teknolojisine dayanan Küresel Konumlama Sitemi'dir. GPS 'in temel tekniği; alıcı ile aynı anda gözlemlenen birkaç uydu arasındaki mesafeleri ölçmektir. Uyduların konumları tahmin edilir ve GPS sinyali ile kullanıcıya yayınlanır. Uyduların birkaç bilinen konumlarıyla ve alıcı ve uydular arasında ölçülen mesafelerle, alıcının konumu belirlenebilir. Bu durumda konum değişikliği alıcının hızıdır. GPS' in en önemli uygulamaları konum belirleme ve yön güdümdür [5].

Bu sistem, ABD savunma bölümüne ait, yörüngede sürekli olarak dönen 24 uydudan oluşur. Bu uydular çok düşük güçlü radyo sinyalleri yayarlar. Yeryüzündeki GPS alıcısı, bu sinyalleri alır. Böylece konum belirlenmesi mümkün olur [6].

Sistem, temel olarak jeodezideki en eski tekniklerden biri olan "geriden kestirme" esasına dayanır. Geriden kestirme, konumu bilinmeyen bir noktadan konumu bilinen noktalara yapılan gözlem ve hesapları kapsar [7]. Konumu bilinen noktalar GPS uydularıdır. Bilinmeyenler, bulunulan noktanın yer merkezli Kartezyen koordinatlarıdır (X,Y,Z). Matematik kuralı olarak bu 3 bilinmeyenin çözümü için 3 ölçü değeri yetiyor gibi gözükse de, saat hatalarını ortadan kaldırmak için en az 4

tane konumu bilinen uyduya ihtiyaç vardır. GPS, 4 boyutlu bir sistemdir (3D+zaman).

2.2. Tarihçe

İlk yapay uydu olan SPUTNIK-1'in 04 EKİM 1957 tarihinde uzaya fırlatılmasıyla uzay jeodezi bilimi içerisinde önemli bir yer edinmiştir. Başka bir deyişle, SPUTNIK-1 ile uzay jeodezisinin fiili gelişimi başlamıştır. Diğer taraftan, günümüzün modern konum belirleme teknolojisi 1960'lı yıllara dayanmakta olup TRANSIT (DOPPLER veya Navy Navigational Satellite System; NNSS) olarak bilinmektedir. Bu sistem yeryüzünden yaklaşık 1100 km uzaklıkta olan 6 uydudan oluşmaktaydı. TRANSIT sistemi A.B.D. Silahlı Kuvvetleri tarafından geliştirilmiş olup ana amaç uçak ya da diğer askeri araçların koordinatlarının belirlenmesiydi. Daha sonraları sistem sivil sektörün kullanımına açılmış ve jeodezik konum belirleme amacıyla 1967 yılından bu yana yaygın olarak kullanılmıştır [8].

GPS, TRANSIT sistemin bazı zayıf yanlarını ortadan kaldırmak için geliştirilmiştir. Örneğin, TRANSIT sisteminde bir uydunun aynı enlemden iki geçişi arasında yaklaşık 90 dakikalık zaman farkı vardı. Dolayısıyla, ölçücü uydunun iki geçişi arasındaki zamanlar için enterpole yapmak zorundaydı. Diğer sorun ise, TRANSIT sistemden elde edilen doğruluklar oldukça düşüktü. Bilindiği gibi, yön güdümde ana amaç anlık (real-time) konum ve hız belirlemek olmasına karşın TRANSIT bunu karşılayamamakta, birkaç günlük Doppler ölçüsü yapılarak ancak desimetre mertebesinde doğruluk elde edilebilmekteydi. Transit'ten elde edilen deneyimler sonucu, hava şartlarından etkilenmeden sürekli gözlem yapabilen ve yeryüzünde tek anlamlı, süratli ve doğru konum belirlemeye olanak veren bir sistem gereksinimi ortaya çıkmıştır [9].

Transit sisteminin gelişmiş bir biçimi olan "NAVSTAR/GPS" (Navigation Satellite Timing And Ranging/Global Positioning System) ABD Savunma Dairesi (Deparment of Defence) tarafından geliştirilen, elinde GPS alıcısı olan herhangi bir kullanıcının, uydu sinyalleri yardımıyla; herhangi bir yer ve zamanda, anında ve

sürekli konum, hız ve zaman belirlenmesinde olanak veren bir radyo yön güdüm sistemidir.

Sistemle ilgili çalışmalar 1973 yılında ABD Deniz Kuvvetleri "TIMATION" programı ile Hava Kuvvetlerinin "621B" projesini birleştirilmesiyle başlamış ve Los Angeles Hava Üssünde kurulmuş olan Ortak Program Bölümünün (JPO, Joint Program Office) sorumluluğuna verilmiştir [8].

2.3. Uygulama Alanları

Bu sistemin ilk kuruluş hedefi tamamen askeri amaçlar içindi. GPS alıcıları yön bulmakta, askeri çıkartmalarda ve roket atışlarında kullanılmak üzere tasarlanmıştır. Ancak, 1980'lerde GPS sistemi sivil kullanıma da açılmıştır. Artık birçok alanda hayati önem taşıyan bir araç olarak kullanıma girmiştir. GPS' in karada, havada ve denizde birçok kullanım alanı vardır. GPS size bulunduğunuz yerleri işaretleme ve belirlediğiniz noktaya geri dönme imkanı sağlar. GPS, kapalı alanlar ve su altı gibi sinyallerin alınmasının güçleştiği yerler dışında dünya üzerinde her yerde çalışır.

Gps sistemin özellikle askeri alanda kullanımı fazladır. GPS kıtalar arası füzelerde ve hassas güdümlü füzelerde kullanılmaktadır. Balistik füzelerde de fırlatma pozisyonunun daha doğru olarak hesaplanması için kullanılmaktadır. Ayrıca Amerikan Nükleer Patlama Gözlemleme Sisteminin büyük bir parçası olarak GPS uyduları nükleer patlama detektörleri içerir [10].

Türk Silahlı Kuvvetleri de izlediği savunma politikasına paralel olarak birçok alanda GPS uygulamalarından yararlanmaktadır. Örnek olarak komando birlikleri intikal, arazide yön bulma gibi birçok alanda GPS 'ten faydalanmaktadır [11].

GPS sistemi askeri kullanımların dışında özellikle araştırmalarda kullanılmak üzere geniş bir dünyaya sahip. Bir ekibin haftalarını alabilecek bir işi, bir araştırmacı çok daha kısa sürede ve daha net sonuçlarla elde edebilmektedir. Örneğin haritalamada kullanılan vazgeçilmez bir sistemdir. GPS sistemi sayesinde bütün dünya haritalanabiliyor. Sadece dağlar, nehirler değil şehirler, şehirlerin sokakları, nesli

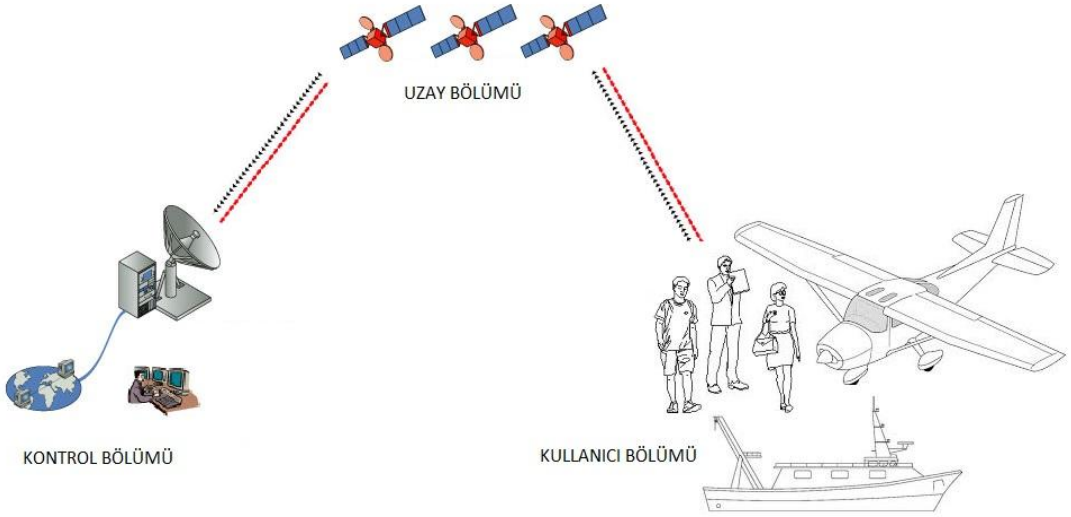
tükenmekte olan hayvanlar, değerli mineraller ve kaynakların her türlüsü, hasar ve afetler [12].

1980'lerin sonlarına doğru uygulamaya giren GPS ile birlikte görme engelliler için "MoBIC, Drishti, Brunel Navigation System for the Blind, NOPPA, BrailleNote GPS and Trekker" isimli projeler yürütülmeye başlamıştır [11].

GPS uçaklarda da diğer yön bulma aygıtlarına ek olarak kullanılmaktadır [11]. Bir başka kullanım şekli ise bu teknolojinin arabalarda kullanılması, Arabalara GPS alıcısı takılarak araç kullanımı kolaylaşmıştır. Günümüzde birçok araçta yön güdüm kiti bulunmaktadır. Yön güdüm cihazında bulunan GPS alıcısı, uydu sinyalleri aracılığıyla aracın bulunduğu koordinatı belirleyerek sürücülere, belirlenen varış noktasına, sesli ve görsel yönlendirmelerle doğru bir şekilde ulaşılmasını sağlamaktadır. Ve bir çok araba kiralama firmaları, dağıtım şirketleri, kargo firmaları arabalarına GPS alıcısı takarak araçlarını takip edebilmekte ve müşterilerine online haritalar üzerinde ürünlerini takip edebilme imkanı sunmaktadır.

2.4. GPS Sisteminin Bölümleri

JPO sorumluluğunda geliştirilen GPS sistemi üç ana bölümden oluşmaktadır. Bunlar uydulardan oluşan Uzay Bölümü, tüm sistemi yöneten Kontrol Bölümü ile alıcıların bulunduğu Kullanıcı Bölümüdür.



Şekil 2.1. GPS Sisteminin Bölümleri

2.4.1. Uzay bölümü

GPS uyduları yer yüzeyinden yaklaşık 20200 km. yükseklikte, 6 orbital yüzeyinde, her yüzeyde 4 uydu olmak üzere, 24 tanedir. Yörüngeleri dairesel şekilde ekvator civarında birbirlerine eşit mesafede ve yine birbirlerine 60° 'lik açı ile yerleşirler. Bu şekilde, tüm dünyanın her anda 4 ile 8 arasında uydu ile kapsama altına alınması sağlanır [13].

Her GPS uydusu;

- Senkronize zaman sinyallerini (uyduların birbirleriyle ve yerle hassas atomik saatlerle uyumu),
- Tüm diğer uydulara ait konum bilgilerini,
- Yörünge Parametrelerine ilişkin iki taşıyıcı frekanstan (L1, L2) yayınlar.
- Kontrol Bölümü tarafından yayınlanan bilgileri alır.

Şekil 2.2. Gps Uzay Bölümü [14]

Uyduların her biri, iki değişik frekansta ve düşük güçlü radyo sinyalleri yayınlamaktadır (L1, L2). Sivil GPS alıcıları L1 frekansını (UHF bandında 1575,42 Mhz), ABD Savunma bölümü alıcıları L2 (1227,60 Mhz) frekansını dinlemektedirler [15]. Bu sinyal "Görüş Hattında" (Line of Sight) ilerler. Yani bulutlardan, camdan ve plastikten geçebilir ancak duvar ve dağ gibi katı cisimlerden geçemez. GPS sinyalleri binalardan yansıdığı için şehir içlerinde araziye oranla hassasiyeti azalır. Yeraltına kazılan tünellerde ise sinyal elde edilemez. Hatalı sinyallerin elde edilebileceği ya da hiç sinyal elde edilemeyen bölgelerde kullanılmak üzere geliştirilen Diferansiyel GPS' ler tarafından bu hatalar en aza indirilerek daha hassas bir yer ölçümü yapılabilir.

Daha rahat anlaşılması için, bildiğimiz radyo istasyonu sinyalleri ile L1 frekansını kıyaslamak istersek; FM radyo istasyonları 88 ile 108 Mhz arasında yayın yaparlar, L1 ise 1575,42 Mhz'i kullanır. Ayrıca GPS' in uydu sinyalleri çok düşük güçtedirler. FM radyo sinyalleri 100.000 watt gücünde iken L1 sinyali 20-50 watt arasındadır. Bu yüzden GPS uydularından temiz sinyal alabilmek için açık bir görüş alanı gereklidir. GPS uyduları tarafından gönderilen elektromanyetik dalgalar atmosferden geçerken bükülmeye uğrarlar. L1 ve L2 bantları farklı dalga boylarına sahip olduğundan farklı oranda bükülmeye uğradığından aradaki farklılık hesaplanarak atmosferik bozulma engellenerek çok daha hassas bir yer bilgisi hesaplanabilir. Sadece L1 bandı

kullanılarak (diferansiyel GPS ile dahi) 98 m. hassasiyet elde edilebilirken, L1 ve L2 bantlarının ortak kullanımı ile 1 m.'nin altında hassasiyete ulaşmak mümkün olmaktadır [16].

Her uydu yerdeki alıcının sinyalleri tanımlamasını sağlayan iki adet özel pseudo-random (şifrelenmiş rastgele kod) kodu yayınlar. Bunlar Korumalı (Protected P code) kod ve Coarse/Acquisition (C/A code) kodudur. P kodu karıştırılarak sivil izinsiz kullanımı engellenir, bu olaya Anti-Spoofing adı verilir. P koduna verilen başka bir isimde "P (Y)" ya da sadece "Y" kodudur [16].

Bu sinyallerin ana amacı yerdeki alıcının, sinyalin geliş süresini ölçerek, uyduya olan mesafesini hesaplamayı mümkün kılmasıdır. Uyduya olan mesafe, sinyalin geliş süresi ile hızının çarpımına eşittir. Sinyallerin kabul edilen hızı ışık hızıdır. Gelen bu sinyal, uydunun yörünge bilgileri ve saat bilgisi, genel sistem durum bilgisi ve ionosferik gecikme bilgisini içerir. Uydu sinyalleri çok güvenilir atom saatleri kullanılarak zamanlanır.

2.4.2. Kontrol bölümü

Kontrol Bölümü, GPS uydularını sürekli izleyerek, doğru yörünge ve zaman bilgilerini sağlar. Dünya üzerinde 5 adet kontrol istasyonu bulunmaktadır (Hawaii, Kwajalein, Colorado Springs (ana merkez), Ascension adası ve Diego Garcia). Bunlardan dördü insansız, biri insanlı ana kontrol merkezidir [17].

Ana kontrol istasyonu, tüm sistemin kontrolünden, her bir uydu için uydu efemeris bilgilerinin ve saat düzeltmelerinin hesabından sorumludur. Diğer 4 istasyon ise izleme istasyonu olarak görev yapmakta ve uydu efemerislerinin belirlenebilmesi için gerekli verileri toplamaktadır. Ayrıca Ascencion, Diego, Garcia, Cape Canavaral ve Kwajalin istasyonlarında efemeris bilgilerini ve saat düzeltmelerini uydulara yüklemek amacıyla yer antenleri de bulunmaktadır. Uydulara bilgi yükleme işlemleri günde bir ya da iki defa yapılmaktadır. Efemeris parametrelerinin GPS uydu yörüngelerine olan uyuşumu 4-6 saat kadar geçerli olup, bu süreden sonra bozulma zamanla orantılı olarak artmaktadır [16].

Şekil 2.3. GPS Kontrol Bölümü

Monitör İstasyonu, dakik atomik saat standardı sağlar ve görünürdeki tüm uyduların uzaklıklarını belirler. Bu uzaklıklar, her 1,5 saniyede bir hesaplanır, iyonosferik ve meteorolojik veri ile düzeltilip 15 dakikalık veri oluşturularak ana kontrol istasyonuna gönderilir [18].

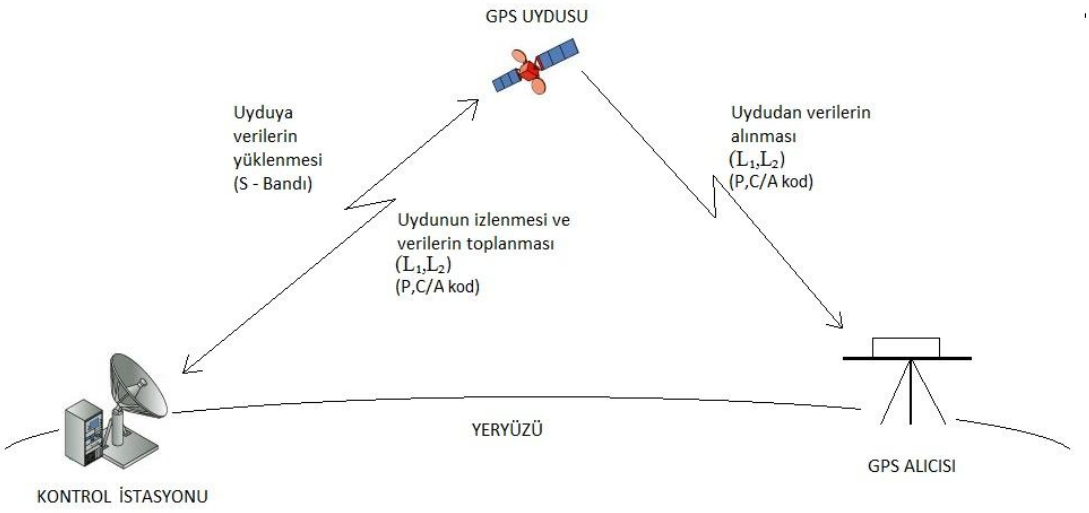
Yer Kontrol İstasyonu, birçok yer anteni barındırırlar, uydular ile haberleşme linkleri teşkil ederler. Ana Kontrol istasyonlarında hesaplanan uydu konumları, saat bilgileri GPS uydularına S bandı kullanılarak yüklenir. Eğer bir yer kontrol istasyonu devre dışı kalırsa, uydular önceden saklanmış yön güdüm verileri ile konum tahmini yapabilirler [18].

2.4.3. Kullanıcı bölümü

Kullanıcı bölümü yerdeki alıcılardır. Çeşitli amaçlarla GPS kullanarak yerini belirlemek isteyen herhangi bir kişi, sistemin kullanıcı bölümüne dahil olur. Genel olarak her türlü amaç için farklı duyarlıkları olan uygun donanımlı GPS alıcıları (receiver) bu bölümü oluşturur. Bir GPS alıcısı; algılayıcı (sensor), kontrol ünitesi, alıcı anteni ve güç kaynağından oluşur [19]. Ölçü sırasında;

- Anlık faz farkı ölçüleri (data, ham ölçüleri)
- Yayın efemerisi bilgileri (uydu yörünge bilgileri)
- Atmosferik bilgiler (iyonosfer ve troposfer bilgileri)
- Mesaj bilgileri(anten yüksekliği ve nokta bilgileri)

elde edilir. Jeodezik amaçla GPS ölçülerinde kullanılan iki çeşit alıcı vardır; alıcıları askeri ve sivil amaçlı alıcılar.



Şekil 2.4. GPS Kontrol-Uzay-Kullanıcı Bölümleri ilişkisi

2.5. GPS' in Çalışma Prensibi

GPS alıcısı yerini belirlemek için, öncelikle uyduların kesin yerini bilmelidir ve onlara ne kadar uzaklıkta olduğunu bulmalıdır. Alıcı uydudan iki çeşit bilgi alır. Bunlardan birisi, uyduların konumlarını bildiren "almanac data" almanak bilgisidir. Almanak bilgisi sürekli olarak yollanır ve GPS’ in hafızasında saklanır. Bu sayede GPS her uydunun yörüngesini bilir ve olması gereken konumu hesaplar. Uydular konum değiştirdikçe almanak bilgisi yenilenir. Uydu yörüngelerinde ufak sapmalar meydana gelebilir. Bu sapmaların hesaplanması için kontrol bölümü uyduların yörünge bilgilerini sürekli olarak izler. Elde edilen bu hata verileri Ana kontrol merkezine ulaştırılır ve düzeltilerek buradan uydulara geri gönderilir. Bu düzeltilmiş

kesin konum bilgilerine " Ephemeris Data" Geçici Bilgi adı verilir. Bu bilgiler güncelliğini 4 ila 6 saat arasında korur. Ephemeris bilgisi daha sonra kodlanarak GPS alıcısına gönderilir. Almanak ve Ephemeris bilgilerini alan GPS alıcısı, uyduların kesin konumlarını sürekli olarak belirler [20].

GPS alıcısının uyduların kesin konumlarını bilmesinin yanı sıra uydulara olan uzaklığını da bilmesi gerekir. Bu sayede, dünya üzerindeki yerini hesaplayabilir. Bunun için basit bir formül kullanılır. Uyduya olan uzaklık; gönderilen sinyalin geliş süresiyle, hızının çarpımına eşittir. Uzaklığı belirlemek için kullanılan bu formülde, hızı zaten bilmekteyiz. Radyo dalgasının hızı, atmosferdeki ufak etkiler sayılmazsa, ışık hızına eşittir (c=300.000km/sn).

Bundan sonra, formülün zaman kısmının hesaplanması gerekir. Çözüm uydulardan gelen kodlanmış sinyallerin içinde saklıdır. Gönderilen koda “Pseudo-Random Kod” adı verilir. Böyle adlandırılmasının sebebi, çok düzensiz bir sinyal olmasıdır. GPS alıcısı da aynı kodu üreterek, uydudan gelen kodla eşleştirmeye çalışır. Bu iki kodu karşılaştırarak aradaki gecikmeyi tespit eder, bu gecikme miktarı ile ışık hızının çarpımı mesafeyi verir.

Yaklaşık olarak bir uydudan sinyalin dünyaya ulaşma süresi 0,06 saniyedir. Saniyenin binde birinde oluşacak bir hata, mesafe ölçümünde 300 km’ lik bir kaymaya sebep olacaktır. GPS alıcısının saati, uydudaki saatler kadar hassas değildir. Alıcıya bir Atom Saati koymak ise çok pahalı ve çok hantal olurdu. Bu yüzden, uyduya olan mesafe ölçümü, “Pseudo Range” olarak adlandırılır. Bu bilgiyi kullanarak pozisyon belirlemek için, 4 uydu kullanılarak saat hatasını minimuma indirinceye kadar ölçüm yapılır [20].

Geometrik Hesabı da şöyle yapılır; diyelim ki, uyduların yerlerini ve uydulara olan uzaklıkları biliyoruz ve birinci uyduya olan uzaklık 20.000 km; bizim yerimiz, merkezi uydu olan ve 20.000 km çapındaki kürenin yüzeyi üzerindeki her hangi bir nokta olabilir. İkinci bir uyduya da 21000 km uzaklıkta olalım. Bu durumda, ikinci küre birinci küre ile kesişerek ara kesitte bir çember oluşturur. Eğer buna 22.000 km uzaklıkta üçüncü bir uydu eklersek, üç kürenin ortak kesim noktası olan 2 nokta elde

ederiz. İki olası pozisyon belirlenmesine rağmen bu iki nokta arasında büyük koordinat farkları mevcuttur. Bu iki noktadan hangisinin gerçek pozisyon olduğunu bulmak için, GPS alıcısına yaklaşık yükseklik verisinin girilmesi gerekir. Bu şekilde GPS geriye kalan iki-boyut içinde kesin pozisyonu belirleyebilir. Fakat üç-boyutta yer belirlenmesi için GPS dördüncü bir uydu daha kullanır. Diyelim ki dördüncü uyduda bizden 19.000 km uzaklıkta olsun, bu dördüncü küreyi, önceki kürelerle kesiştirirsek, elimizde sadece bir ortak kesim noktası kalır. Bu da üç-boyutta kesin konumu belirtir [21].

Çoğu modern GPS alıcıları paralel, çok kanallı çalışma sistemine sahiptir. Daha önceleri yaygın olan tek kanallı GPS alıcı modelleri çeşitli ortamlarda sürekli olarak uydu takip edemiyorlardı. Paralel alıcılar ise her biri bir uyduyu izlemek üzere, 5 ile 12 alıcı devresine sahiptirler. Bunların içinden en kuvvetli dört sinyal takip edilir. Paralel alıcılar uydulara hızla kilitlenebildikleri gibi, yüksek binalar, sık ormanlar gibi zor ortamlarda da efektif bir şekilde çalışırlar.

2.6. GPS Hata Kaynakları

GPS alıcılarının doğru konum ve hız bulmalarını etkileyen birçok faktör vardır. Genel olarak GPS sisteminin performansı GPS alıcıları ile GPS uyduları arasındaki uzaklığının doğru bulunmasına ve GPS uydularının sinyal gönderildiği andaki konumunun tam olarak bilinmesine bağlıdır [21].

GPS ile konum belirlemenin temelini zaman ölçümü oluşturduğundan konum belirlemede en büyük hata kaynağını uydu saati hataları oluşturmaktadır. Uydu saat hatası uydu saat zaman ile GPS zaman arasındaki farktır. Her uydu saatinin davranışının izlenmesinde en iyi çaba gösterilse bile dünyanın manyetik alanı, yerin ve ayın çekim etkisi, radyasyon vb. sebeplerden dolayı saatlerin davranışı duyarlı bir şekilde modellenememektedir. Bu etki, ölçü yapılan tüm noktalardaki alıcılar için aynı büyüklüğe sahiptir. Oluşacak bu hatayı önlemek amacıyla GPS kontrol merkezi tarafından her uydu için saat düzeltmeleri gönderilir. Düzeltilmiş zaman bilgisi GPS uyduları tarafından yön güdüm bilgisinin içerisine yerleştirilerek yayınlanır [22].

GPS ölçümlerinde bir hataya sebep olabilecek etken uyduların yörünge tahmin hatasıdır. Tüm uydular için, yön güdüm mesajında diğer parametrelerle birlikte yayınlanması için, optimal yörünge tahminleri hesaplanarak uydulara yüklenir. Buna karşın tahmin edilen yörüngelerinde ufak kaymalar yapabilirler.

GPS uydularından alıcıya gelen sinyaller atmosferik etkilerden olumsuz etkilenmektedir. Atmosfer tabakasının GPS sinyalleri üzerindeki bozucu etkileri iyonosfer ve troposfer tabakalarında meydana gelmektedir. GPS uyduları zamanlama bilgilerini radyo sinyalleri olarak gönderirler. Radyo sinyallerinin atmosfer içinde ışık hızında hareket ettiği kabul edilse de, ışık hızı sadece vakum ortamında sabittir. Radyo sinyalleri, içinde bulundukları ortama göre yavaşlama gösterirler.

İyonosfer, yerküreden 50 km. ile 1000 km. yükseklikte olan saçıcı bir ortamdır. Bu bölgede, güneşten gelen ultraviyole ışınlar iyonlaşarak gaz molekülü parçaları oluştur ve serbest elektronlar salınır. Bu serbest elektronlar, yayılan GPS işaretleri de dahil olmak üzere, tüm elektromanyetik dalga yayılımını etkiler [23]. İyonosfer etkisi, iyonosfer saçıcı bir ortam olduğundan, frekans bağımlıdır. GPS sistemi birkaç farklı frekans üzerinden çalışacak şekilde tasarlanmıştır ve bu özellik kullanılarak, iyonosfer etkileri ölçülebilir ve düzeltilebilir [24].

Troposfer, atmosferin en alt kısmıdır. GPS taşıyıcı frekansları için saçıcı olmayan bir ortamdır. GPS işareti üzerindeki troposfer etkileri, çalışma frekansından bağımsızdır. Elektromanyetik işaretler, troposferdeki nötr atom ve moleküllerden etkilenir. Bu etkilere, troposferik gecikme veya troposferik kırılım denir [25].

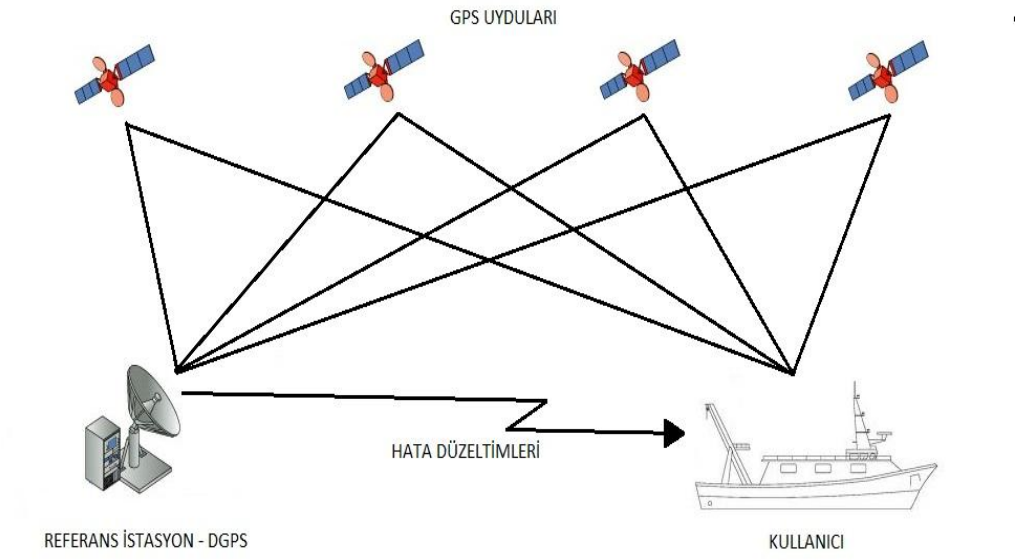
Alıcı ölçümlerindeki en büyük hata kaynaklarından birisi de çok yol etkisidir. İşaretin, alıcıya çevredeki nesnelerden, veya yer üzerinden yansıyarak birden fazla yol üzerinden ulaşması bu etkiyi doğurur.

Alıcı ölçümlerini doğal hataların dışında etkileyen diğer bir faktör ise, uydu konum bilgilerinin ve/veya uydu saatlerinin kasıtlı olarak bozulmasıdır. Bunlar ABD Savunma Bölümü tarafından devreye sokulan kasti hatalardır (selective availability) [26]. Eğer sabit bir GPS alıcısını hareketinin konum grafiğini, Seçici Kullanılabilirlik

devrede iken çizmek istersek, pozisyonumuzun 100 m. çapındaki bir daire içinde dolaştığını görürüz. Askeri alıcılarda bulunan kod çözücü anahtarlar, hangi hataların devrede olduğunu ve ne kadar olduğunu söyler, böylece hatalar giderilebilir. Bu yüzden askeri GPS alıcıları, çok daha hassas ölçüm kabiliyetine sahiptir [27].

2.7. GPS Hassasiyeti Artırma Yöntemi - DGPS

DGPS (Diferansiyel Yer Belirleme Sistemi-Diferantial Global Positioning System) GPS alıcılarının konum doğruluğunun iyileştirilmesi için kullanılan sistemdir. Sözde düzeltme (pseudorange correction) bilgisi üreterek bu kullanıcı alıcıya gönderir [28]. DGPS' teki temel fikir, iki ya da daha çok alıcının aynı uydudan aldığı sinyal sayesinde, ortak hataları gidermektir. Bu alıcılardan biri referans alıcıdır, sabittir ve konumu daha önceden duyarlı olarak belirlenmiş bir noktaya kurulur. Diğer alıcılar da, kullanıcılardır ve kullanıcılar referans istasyonun görüş alanı içindedirler. Referans istasyon gözlem yaptığı tüm uydulara ait uydu-alıcı uzaklıklarını hesaplayarak bu değerleri kendi duyarlı konumundan yararlanarak hesapladığı, olması gereken sözde mesafe ölçümüyle (pseudorange) karşılaştırır. Aradaki farklar gözlem hatası olarak yorumlanır ve bu farklar konumu belirlenecek olan noktalardaki gezen alıcılar tarafından kaydedilen gözlemlere düzeltme olarak getirilerek gezen alıcının konumu doğru olarak belirlenir [29].



Şekil 2.5. DGPS

2.8. GPS 'den Alınan Verinin (NMEA) İçeriği

GPS alıcıları uydudan almanac verisini alır ve üçlü kestirim metoduyla görünen uydularla kendi arasında ki mesafeyi hesaplayarak konumunu bulur. Veri alındıktan ve pozisyon hesaplandıktan sonra, veri NMEA 0183 standartlarına göre yapılandırılmış ve seri olarak 4800 baud hızında iletilir [30]. NMEA 0183 Ulusal Denizcilik Elektroniği Birliği (National Marine Electronics Association) tarafından farklı üreticilerin ürettiği ekipmanlar arasında iletişim sağlanabilmesi için standartlaştırılmış elektronik dildir [31].

GPS alıcısı tarafından verilen veri enlem, boylam, yükseklik, hız, zaman gibi birçok bilgi içerir. Bu standartta bir dizi veri bir cümle şeklinde iletilir. Ve her cümle birbirinden bağımsızdır.

$GPGGA,hhmmss.ss,llll.ll,a,yyyyy.yy,a,x,xx,x.x,x.x,M,x.x,M,x.x,xxxx*hh

Yukarıda GPS alıcısından elde edilen NMEA formatında örnek bir cümle bulunmaktadır ve açıklaması şu şekildedir [32]:

- Her cümle '$' işaretiyle başlar.
- Bütün standart cihazlar ne için kullanıldığını gösteren iki harflik ön eke sahiptir, GPS alıcıları için bu ön ek GP dir.
- Ön ekten sonra gelen harfler cümlenin içeriğini vermektedir. GGA = Konumlama Sistemi Düzeltilmiş Veri (Global Positioning System Fix Data)
- Zaman bilgisi (Greenwich zamanı)
- Enlem
- Enlem yarı küre bilgisi (N kuzey, S güney)
- Boylam
- Boylamın Greenwich yönü (W batı, E doğu)
- Çözüm durumu (0 çözüm yok, 1 mutlak pozisyon, 2 diferansiyel pozisyon)
- Kullanılan uydu sayısı
- HDOP (Yatay uydu geometrisi katsayısı)
- Yükseklik

- Yükseklik birimi (m, metre)
- Jeoid yüksekliği
- Sağlama Toplamı (Checksum)

BÖLÜM 3. SİSTEM TASARIMI

Sistem mobil uygulama, web sayfası, veritabanı, web servisi olmak üzere dört alt bölümden oluşmaktadır. Bu bölümde sistemin tasarımında kullanılan teknolojiler, yazılımlar hakkında genel bilgi ve kurulumu anlatılmaktadır. Mobil uygulama için platform olarak Android İşletim Sistemi tercih edilmiştir ve Android üzerinde uygulama geliştirmeye başlayabilmek için 4 yazılıma ihtiyaç vardır, Java Develpoment Kit (jdk 1.6) , Eclipse, Android SDK ve Android Development Tools Plugin. Veritabanı için MySQL Server, web servisi için RESTFUL mimarisi, web sayfası için JSP(Java Server Pages) tercih edilmiştir.

3.1. Android Nedir?

Android; açık kaynak kodlu, linux tabanlı, mobil cihazlar (akıllı telefon, PDA, tablet bilgisayar, vb.) üzerinde koşması amaçlanarak Google tarafından geliştirilmiş bir işletim sistemidir [33]. Android mobil cihazlar için geliştirilmiş olsa da farklı tür donanımlarda koşması için çalışmalar yapan firmalar mevcuttur.

3.1.1. Tarihi

Temmuz 2005'te Google Android Inc.'i almış ve ufak bir başlangıç şirketini Palo Alto'da kurmuştur. Android'in kurucuları, Google'a çalışmak için giden Andy Rubin (Danger' in kurucusu),Rich Miner (WildFire'ın kurucusu), Nick Sears ve Chris White'dır [34].

2007 yılında Google mobil cihazlar için açık standartlar oluşturmak için Open Handset Alliance (OHA) topluluğunu kurdu ve bu topluluğa önde gelen 34 teknoloji firmasının (HTC, Dell, Intel, Motorala, Qualcomm, Texas Instruments, Samsung, LG, T-Mobile, Nvidia, vb.) katılımını sağladı. Google OHA topluluğundaki diğer

firmalar ile beraber Android'in çıkartılması için çalıştı ve Android ilk versiyonu (Android 1.0) 2008 yılının sonlarında çıkartıldı [35].

3.1.2. Versiyonları

1. Nisan 2009'da Android 1.5 (Cupcake) yayınlandı. Bluetooth desteği, kamera kaydı, video gösterimi, yazı tahmin edebilen klavye ve animasyonlu ekran özellikleri Android'e kazandırıldı.
2. Eylül 2009'da Android 1.6 (Donut) yayınlandı.Ekran çözünürlüğü yükseltildi. Doğru çeviri yapma ve okutma özelliği Android'e kazandırıldı.
3. Ekim 2009'da Android 2.0 ve 2.1 (Eclair) yayınlandı. HTML 5 ve Bluetooth 2.1 desteği kazandırıldı. Sanal klavye ve yüksek ekran çözünürlüğü desteği ile özelleştirilebilen arayüz özelliği eklendi.
4. Mayıs 2010'da Android 2.2 (Froyo) yayınlandı. 720p ekran çözünürlüğü desteği eklendi.
5. Şubat 2011'de Android 2.3 (Gingerbread) yayınlandı. Çoklu kamera ve çoklu dokunmatik desteği eklendi.
6. Yine Şubat 2011'de Android 3.0 (Honeycomb) yayınlandı. Android işletim sistemi, tabletlerle uyumlu hale getirildi.
7. Ekim 2011'de Android 4.0 (IceCream Sandwich) yayınlandı. Yüz tanıma fonksiyonları ve NFC ile dosya paylaşımı özellikleri eklendi.
8. Mart 2012'de Android Market'in adı Google Play olarak değiştirildi.
9. Temmuz 2012'de 4.1 (Jelly Bean) yayınlandı. Aynı anda iki uygulama açma özelliği Android'e kazandırıldı.
10. Ekim 2012'de 4.2 (Jelly Bean Plus) yayınlandı.
11. 11 Şubat 2013'te 4.2.2 (Jelly Bean Plus) yayınlandı.

3.1.3. Uygulama mağazaları

Android’in geniş bir uygulama geliştirme kitlesi mevcuttur. Bu nedenle uygulama mağazalarındaki uygulama sayısı diğer işletim sistemlerine uyumlu uygulamalarla kıyaslandığında çok büyük olduğu açıkça görülmektedir. Android cihazlara uygulama geliştirmenin kolaylığı, maliyetsiz bir ortamdan kaynaklanmaktadır. Java

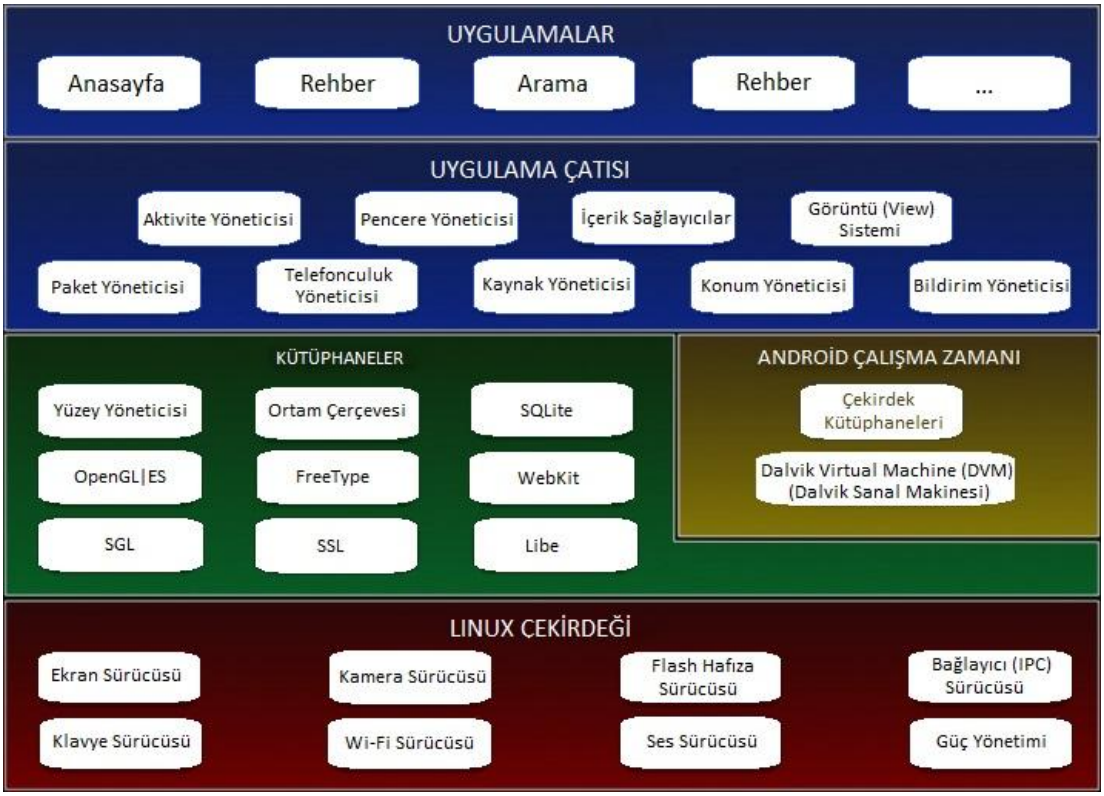
bilgisi. Eclpise kod geliştirme ortamı ve Eclipe üzerine kurulu ADT (Android Development Tool) eklentisi uygulama geliştirmek için yeterlidir. Ve android, özgür yazılım mantığını uygulama mağazalarına da yansıtmış durumda. Kullanıcılar tek bir uygulama mağazasına bağımlı değiller, Android farklı mağazalardan uygulamalar indirilmesine izin veriyor. Android uygulamaları indirebileceğimiz mağazalardan bir kısmı:

- Google Play
- SlideMe
- AndroidGear
- AppsLib
- insydemarket

3.1.4. Mimarisi

Android mimarisi 4 katmanda incelenmektedir.

1. Uygulamalar (Applications)
2. Uygulama Çerçevesi (Application Framework)
3. Kütüphaneler (Libraries) ve aynı seviyede bulunan Android Çalışma Zamanı (Android Run Time)
4. Linux Çekirdeği (Linux Kernel)



Şekil 3.1. Android İşletim Sistemi Mimarisi

3.1.4.1. Uygulamalar katmanı

Android İşletim Sistemi'nin kullanıcıya en yakın olduğu bölümdür. Uygulamaların içerisinde e-mail istemcisi, sms programı, takvim, google maps, telefon rehberi gibi temel uygulamalar ve sonradan yüklenen 3.parti yazılımlar (twitter istemcisi, facebook istemcisi gibi) yer almaktadır [36].

3.1.4.2. Uygulama çatısı (Application framework)

Android yazılım geliştiriciler için oldukça zengin bir platform sunar. Geliştiriciler yazılımlarında donanım bilgileri, konum bilgisi, arka plan servisleri gibi birçok bilgiye hiç bir kısıt olmadan rahatlıkla erişebilirler. Burada uygulama geliştiricileri için sunulan yöneticiler de yer almaktadır. Geliştirici kolaylıkla Java programlama dilini kullanarak Android İşletim Sistemi üzerinden uygulama geliştirebilir.

3.1.4.3. Kütüphaneler ve android çalışma zamanı

Android içerisinde yer alan ve genelde C++ ve C dilleri ile yazılmış olan kütüphaneler bulunmaktadır. Bu katmanda sistem kütüphaneleri, mp3, mpeg4, jpg gibi çoklu ortam bileşenleri için medya kütüphaneleri bulunur. Android İşletim Sistemi kendi ilişkisel verilerini tutabildiği bir veritabanına sahiptir. Bu veritabanı SQLite olarak isimlendirilir. Kütüphaneler katmanında veri tabanı için SQLite kütüphaneleri gibi temel kütüphaneler yer alır. Genellikle bu bilgilere uygulama çatısı aracılığı ile erişilir [37].

Android Çalışma Zamanı (Android Runtime) Çekirdek Kütüphaneleri ve Android Sanal Makinası (Dalvik Virtual Machine) bilgilerini taşıyan alan olarak görevlendirilmiştir.

3.1.4.3.1. Dalvik sanal makinesi (Dalvik virtual machine)

Java, 1995 yılında Sun Microsystems adlı şirket tarafından duyurulan bir programlama dili ve yazılım alt yapısıdır. Java programlama dilini diğer programlama dillerinden ayıran en önemli özellik sanal makine (virtual machine) kavramını başarılı bir şekilde uygulamasıdır.

Java Swing/AWT ya da Web uygulaması olarak geliştirilen sınıflar (.java), öncelikle bytecode diye adlandırılan (.class) dosya türlerine derlenir. Bu dosyalar işlemci üzerinde doğrudan çalıştırılabilir programlar yani executable değillerdir. Öncelikle bu bytecode'ların yorumlanıp üzerinde çalıştırılan işlemcinin komutlarına dönüştürecek bir katmana ihtiyaç vardır. Java Sanal Makinası (JVM) bu ara katmanı oluşturarak yazılan herhangi bir programı herhangi bir sistem üzerinde çalıştırılabilmesini sağlar. Java programlamanın sistemden bağımsız olmalarının temelinde, derlendiklerinde çalıştırılabilir kodlar yerine bytecode üretilmesi ve bunların sanal makine tarafından yorumlanması yatar [38].

Sun Microsystems, Java dilinin sadece kendisine ait bir dil ve teknoloji olarak kısıtlamamış olup, kamunun yararını gözeterek Java Community Process (jcp.org)

adı verilen bir topluluğa bırakmıştır. Bu topluluğun uygunluk onayına göre çeşitli şirketler çeşitli Java Sanal makineleri oluşturmuşlardır. Bu şirketlerin içerisinde Sun, Oracle, IBM, BEA gibi şirketler bulunmaktadır [38].

Google firması, Android üzerinde Java sanal makinesi olarak bu şirketlerin ürünlerini kullanmak yerine kendi geliştirdiği Dalvik adını verdiği bir Java Sanal Makinesi kullanmaktadır.

JVM ile DVM nin çalışma mantıkları aynı olsa bile aralarında çok önemli iki fark bulunmaktadır. Birincisi DVM; bir Stack makinesi olarak değil, bir Register makinesi olarak tasarlanmıştır. Register' lar direk mikroişlemcilerin içerisinde bulunan ara hafıza birimleridir. Register' lar, birden fazla ara sonuçları olan matematiksel işlemleri oldukça hızlandırmaktadırlar. Google firması üretmiş olduğu DVM ile Register makine kodları oluşturan bir Java Runtime ortamı oluşturmuştur. Böylelikle modern mikroişlemcilerin olanaklarını da optimal bir şekilde kullanmıştır. Yani Register makineleri, Stack makinelerine kıyasla daha hızlı çalışırlar [38].

Kısacası Android başarısının altında yatan, başka bir özellik Dalvik Sanal Makinesi'dir denilebilir. Dalvik Sanal Makinesi için sistemin en önemli parçasıdır denilebilir. Dalvik Sanal Makinesinin geliştirilme amacı minimum donanım konfigürasyonunda, maksimum performans elde edebilecek Java yazılımlarını çalıştırabilmektir.

Aşağıda Dalvik Sanal Makinesi için bazı özellikler listelenmiştir:

- Uygulama taşınabilirliği ve çalıştırılmasında tutarlılık sağlar.
- Optimize edilmiş dosya formatı (dex) ve Dalvik bytecode çalıştırılır.
- Java sınıf/jar dosyalarının yapısı dex formatına dönüştürülür.
- Gömülü ortamlar için tasarlanmıştır.
- Her süreç için ayrı bir dalvik sanal makinesi çalıştırılır.
- Belleği çok verimli kullanır.

DVM' nin bir diğer önemli özelliği de, daha çok az hafıza ile yetinmesidir. Böylelikle birden fazla VM'nin paralel bir şekilde küçük bilgisayar sistemlerinde de çalışması mümkün olmaktadır. Bu da farklı bir modelin oluşmasına imkan tanımaktadır. Çalıştırılan her program kendisi ile birlikte kendine ait bir process ve kendine ait bir DVM'de çalışmaktadır. Böylece Sandbox prensibi denilen tüm verilerin ve kaynakların dışarıdan gelecek erişime kapalı hale gelmesi sağlanır. Bu da program güvenliği için oldukça önemlidir [38].

3.1.4.4. Linux çekirdeği

Android mimarisinin en alt bölümüdür. Bu bölümde donanımsal bilgiler yer almaktadır. Uygulamaların çalışabilmesi için gerekli sürücüler bu bölümde yer alır. Aşağıda bu bölümde yer alan sürücüler listelenmiştir:

- Display Driver
- Camera Driver
- Flash Memory Driver
- Binder (IPC) Driver
- Keypad Driver
- Wi-Fi Driver
- Auido Drivers
- Power Management

3.1.5. Android bileşenleri ve yaşam döngüleri

Uygulama bileşenleri Android uygulamasının yapı taşlarıdır. Her bir bileşen sistemin uygulamanıza girebileceği farklı bir noktadır. Bütün bileşenler kullanıcı için asıl giriş noktası değildir ve bazıları birbirine bağlıdır, fakat her biri kendi başına bir oluşumdur ve özgül bir rol oynar- her biri uygulamanızın genel davranışını belirleyen özgün bir yapı taşıdır [39].

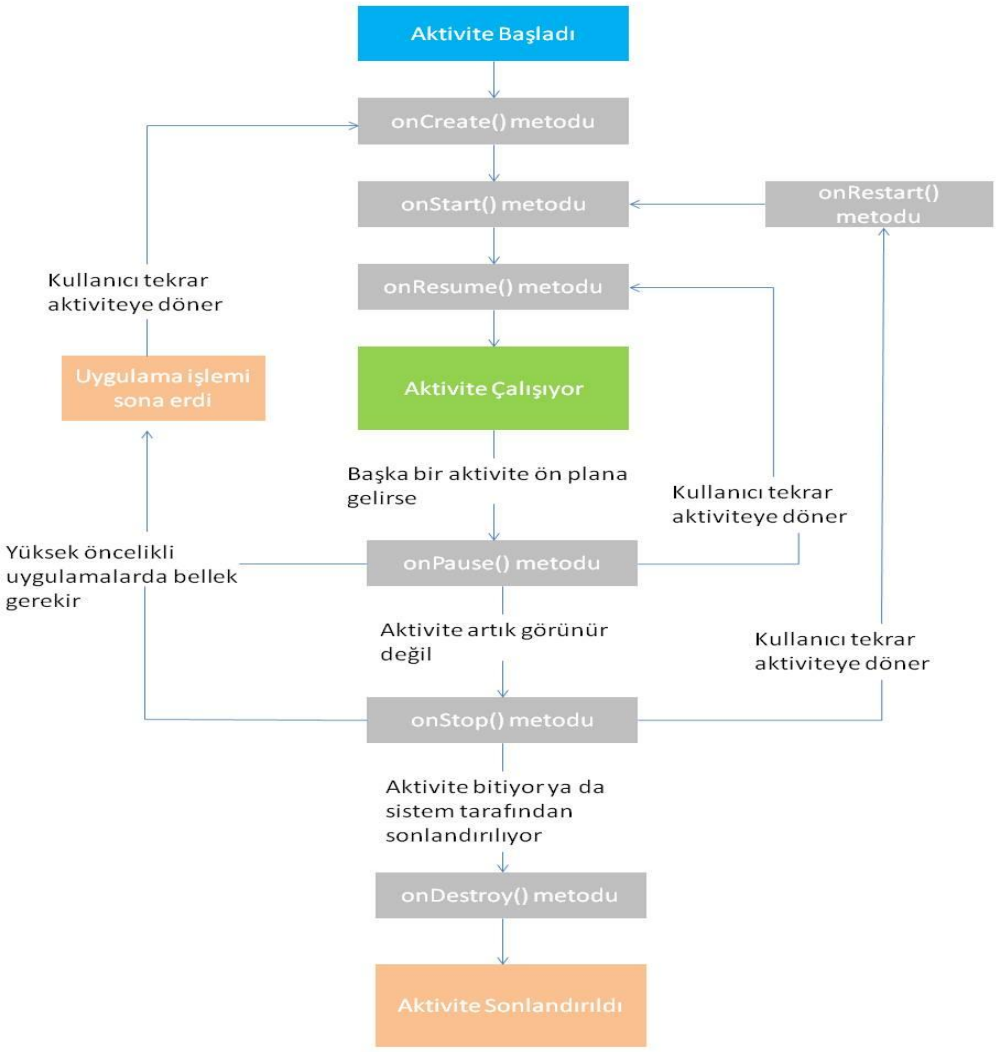
Erişim hakkı sistemi sayesinde programcı, her uygulaması için diğer uygulamaların kullanımına izin vereceği bileşenlerini belirleyebilir, standart uygulamaların

bileşenlerini kullanarak, yeni uygulamalar geliştirebilir. Böylelikle programcı her defasında onları yazmak zorunda kalmamış olur. Dört çeşit uygulama bileşenleri vardır.

3.1.5.1. Aktivite (Activity)

Aktivite, kullanıcıya görsellik sunan sınıftır. Bu pencere üzerinde etiket, metin giriş alanları ve buton gibi program elemanları yer alır. Her aktivitede yer alan elemanları göstermek için ayrı bir .xml dosyası oluşturulur. Oluşturulan her ekran için ilgili sınıf Aktivite sınıfından miras alacaktır. Android mimarisindeki yapıları gereği çalışma zamanı içerisinde sadece bir aktivite çalışabilir. Bu sırada daha önce, geri planda çalıştırılmış aktiviteler pasif duruma taşınır.

Aşağıdaki akış şeması bir Activity' nin önemli durum yollarını göstermektedir. Renksiz dikdörtgenler bir Activity' nin durumlar arasında geçiş yaparken işlemlerin gerçekleştirilmesi için çağrılan geri dönüşüm metotlarını ifade etmektedir. Renkli dikdörtgenler ise Activity' nin içinde bulunabileceği asıl durumları göstermektedir.



Şekil 3.2. Activite Yaşam Döngüsü [40]

Bir Activite' nin içerisinde gözlemlenebilecek 3 tane anahtar döngü vardır [40] :

1. Uygulamanın tüm yaşamı; ilk onCreate() metodunun çağrılmasıyla başlayıp tek ve son olarak onDestroy() metodunun çağrılması arasındaki geçen süredir. Programcı uygulamasında global durumda ki değerlerini onCreate() metodu içerisinde, kaynakların durdurulmasını ise onDestroy() metodu içerisinde set etmelidir.
2. Bir Activity' nin görünür ömrü onStart() metoduyla başlayıp, ona karşılık gelen onStop() metodunun çağrısı arasındadır. Bu süre zarfında program ön planda ve kullanıcıyla etkileşim halinde olmamasına rağmen, kullanıcı ekranda bu Activity'yi görür. Bu iki metot arasında kullanıcıya gösterilmesini

istediğimiz kaynakların yönetimi yapılır. Faaliyetimizi kullanıcı için görünür ya da gizli yapmak istediğimiz zamanlarda bu iki metod birçok kez çağrılabilir.

3. Bir Activity'nin ön planda ki ömrü ise onResume() metoduyla başlayıp, ona karşılık gelen onPause() metodunun çağrısı arasındadır. Bu süre zarfında program diğer bütün faaliyetlerin önünde ve kullanıcıyla etkileşim halindedir. Bir Activity sık sık yeniden başlatma ve durdurma durumları arasında gidip gelebilir. O yüzden bu durumların içerisine yazılan kodların işlenirliği karmaşık olmamalıdır.

3.1.5.2. Servisler (Services)

Servisler uygulamanın arka planda yapması gereken işlerini aktivitelerden bağımsız yapmalarını sağlarlar. Kullanıcının ekranında herhangi bir aktivite görünürken arka planda bir veya daha fazla sayıda servis çalışıyor olabilir. Örnek verilecek olunursa, kullanıcı cihazın menülerinden faydalanırken arka planda müzik çalar çalışıyor olabilir. Bu durumun sağlanabilmesi için müzik çalma yeteneğine sahip bir servisin çalışıyor olması gerekmektedir. Bu durum gibi görsel olmayan çoklu ortam işlemleri, web üzerinden veri aktarımı, kalıcı hafızaya veri yazma okuma gibi zaman alan işlemler kullanıcının ekran üzerindeki çalışmasını etkilemeden arka planda servisler aracılığıyla yapılabilir. Bir servis Service sınıfından türetilir.

Her servisin yaşam döngüsü onCreate() ile başlar ve onDestroy() metoduna kadar devam eder. Aktif olarak servisin çalıştığı kısım ise onStart() metodu sonrasıdır. Servisler kullanım biçimi olarak iki şekilde kullanılabilirler [41].

1. Herhangi birisi tarafından başlatılıp sonlandırılmasına izin verilebilir. Bu durumda uygulama servisi Context.startService() metodu ile başlatır ve Context.stopService() metodu ile sonlandırır. Ya da servis kendini Service.stopSelf() ya da Service.stopSelfResult metotlarıyla sonlandırabilir.
2. Diğer modelde ise programatik olarak bir ara yüz kullanılarak bu işlemler gerçekleştirilebilir. Kullanıcılar Context.bindService() metoduyla servise bağlantıyı sağlayabilir ve Context.unbindService() metodu ile servisle

bağlantısını kapatabilir. Bu bağlantı yönteminde birden çok istemci eş zamanlı olarak bağlantı açıp bağlantı kapatabilir.

Ayrıca bu modellerde önce startService() metodu ile servis başlatılıp yeniden servisin kullanımına ihtiyaç duyulduğunda bindService() metodu kullanılabilir. Bu durumda da tüm istemciler bağlantıyı kapatana kadar yapılan stopService() metot çağrıları bağlantının kapanmasını bekleyeceklerdir.

3.1.5.3. Yayın alıcılar (Broadcast receivers)

Yayın alıcılar görsel ara yüz sunmayan, sistemdeki yayınları alan ve uygulamanın isteklerine göre gerekli işlemleri yapan Broadcast Receiver sınıfından türetilmiş bileşenlerdir. Genellikle yayınlar Android' in kendisi tarafından yapılır. Örneğin pil azaldığında, dil değiştiğinde, güç bağlantısı yapıldığında, cihaz kapanırken ve bunun gibi sistem için önemli görüldüğü durumlarda, uygulamaların gerekiyorsa kendi işlemlerini yapmaları, bazı durumlar için önlemlerini almaları için yayın yapar.

Bir broadcast mesajı alıcıya ulaştığında android onReceive() metodunu çağırır ve Intent nesnesini mesajı da içeren bir şekilde bu metoda aktarır. Broadcast Receiver bileşeni tek metodu olan onReceive() çalışıyorken aktif olarak kabul edilir. Çalıştırılan kod onReceive() metodundan dönmüşse sistem onu artık aktif değil diye kabul eder [42].

3.1.5.4. İçerik sağlayıcılar (Content providers)

Bu tip bileşenler ise Android sistemi veya bir uygulama tarafından bütün sistemce erişilebilinen kaynaklara erişim sağlayan arabirimlerdir. Örneğin, Telefon rehberi, sistem üzerinde bulunan resim, müzik vb. verilerin uygulamalarca erişimini sağlayan arabirimlerdir. SQL benzeri erişim ara yüzüne sahiptirler.

Yukarıda ki bileşenlerden üçü - aktiviteler, servisler, yayın alıcılar - "intent" adı verilen eş zamanlı olmayan bir mesajla aktifleştirilirler. Intent 'ler Android için hangi bileşenin etkin hale geleceğinin belirlenmesini sağlayan bilgileri tutar ve aynı

zamanda yazılım bileşeni Intent nesnesini alınca hem ne yapması gerektiğini hem de ihtiyacı olduğu bilgileri Intent nesnesinden okur.

İçerik sağlayıcısı ise, bir ContentResolver (içerik çözümleyici) nesnesinden istek geldiğinde aktif hale getirilir. İçerik çözümleyici, içerik sağlayıcı ile ilgili olan tüm doğrudan işlemleri yönetir. İçerik sağlayıcı ile işlem yapması gereken bileşenler direk olarak iletişime geçmek yerine, ContentResolver nesnesi metotlarını kullanarak işlemlerini gerçekleştirirler. Böylelikle, içerik sağlayıcı ile bilgi isteğinde bulunan bileşen arasında güvenlik amaçlı bir soyutlama katmanı oluşturulur.

3.1.6. Sandbox prensibi

Android işletim sistemine ek uygulamaların yüklenebilmesi, akıllara bu güvenlik açığı oluşturur mu sorusunu getirmektedir. Çünkü bir uygulama başka bir uygulamanın verilerine izinsiz erişim sağlaması durumunda, ciddi bir güvenlik açığı oluşmuş demektir. Fakat, uygulamalar arasındaki veri alış verişinin istenen bir durum olmasıdır. Bunun için Android işletim sistemini her uygulamayı güvenli zırh olarak tabir edilen Sanbox' larda çalıştırmaktadır. Sanbox' lar, kısıtlı erişim haklarına sahip olan çalışma ortamlarıdır. Burada işletim sistemine veya diğer uygulamaların verilerine doğrudan erişim kısıtlanmıştır ve erişim hakları düzenlenmeden uygulamanın Sanbox' ın dışına çıkması mümkün değildir. Erişim haklarının düzenlenmesi uygulamaların içerisinde AndroidManifest.xml dosyasında yapılır [43]. Yani program kaynak dosyalarında yer alan aktivite, hizmet ve içerik sağlayıcılar, manifest dosyalarında bildirimi yapılmadığı takdirde sistem tarafından görülmez ve çalıştırılmazlar.

Manifest dosyası uygulamaya ait bileşenlerin bildirimine ek olarak aşağıdaki işlemleri de yapar:

- İnternet girişi veya kullanıcı bilgilerinin sadece okunur girişi gibi kullanıcı izinlerini belirlemek.
- Uygulamanın gerektirdiği minimum API seviyesini bildirmek.
- Uygulamanın gerektirdiği veya kullandığı donanım ve yazılım özelliklerini bildirmek (kamera, bluetooth hizmeti gibi).

- Android API'leri dışında uygulamanın gerektirdiği API kütüphaneleri (Google Maps kütühanesi gibi).

Aşağıda manifest dosyası içinde basit bir aktivite bildirimi yer almaktadır:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest ... >
   <application android:icon="@drawable/app_icon.png" ... >
      <activity android:name="com.example.project.ExampleActivity"
            android:label="@string/example_label" ... >
      </activity>
      ...
   </application>
</manifest>
```

3.2. MySQL Server

MySQL en popüler açık kaynak kodlu, ilişkisel SQL veritabanı yönetim sistemidir, Oracle Şirketi tarafından geliştirilip, dağıtılıp, desteklenmektedir [44].

Bir veritabanı yapılandırılmış bir veri yığınıdır. Basit bir alışveriş listesinden bir resim galerisine veya bir şirket ağındaki büyük boyutlarda ki veriye kadar her şey olabilir. Bir bilgisayar veritabanında depolanan veriye ekleme yapmak, ulaşmak ve işlem yapmak için MySQL server gibi bir veritabanı yönetim sistemine ihtiyaç vardır. Bilgisayarlar büyük miktarda veriyi yönetmekte iyi olduklarından, veritabanı yönetim sistemleri, bağımsız yardımcı programlar olarak veya diğer uygulamaların bir parçası olarak işlem yapmada merkezi rol oynar.

MYSQL ilişkisel veritabanıdır. İlişkisel bir veritabanı verinin hepsini büyük bir depoya koymaktansa veriyi ayrı tablolarda depolar. Veritabanı yapıları hız için optimize edilmiş fiziksel dosyalar halinde düzenlenirler. Veritabanları, tablolar, sanal tablolar (views), satırlar ve sütunlar gibi nesnelerle mantık modeli esnek bir programlama ortamı sağlar. Farklı veri alanları arasındaki ilişkileri yöneten kurallar

koyarsınız, mesela bire-bir, biri-birçoğa, tek, ilişkisel veya isteğe bağlı ve farklı tablolar arasında işaretçiler. Veritabanı bu kuralları dayatır ki, iyi tasarlanmış bir veritabanıyla uygulamanız asla çelişkili, tekrar eden (duplicate), artık satır, güncellenmemiş veya eksik veri görmez [44].

MySQL yazılımı açık kaynak kodludur ve bu demektir ki isteyen herhangi biri bu yazılımı kullanabilir ve modifiye edebilir. Herkes bu yazılımı internetten indirebilir ve herhangi bir ödeme yapmaksızın kullanabilir. Dilerseniz kaynak kodunu inceleyip ihtiyaçlarınıza göre değiştirebilirsiniz. MySQL GPL (General Public License-Genel Kamu Lisansı) kullanır. MySQL kodunu ticari bir uygulamaya katmak isterseniz MySQL'den ticari sürümünü satın alabilirsiniz [44].

MySQL veritabanı sistemi mysqld sunucusu etrafında merkezlenen bir istemci/sunucu mimarisi kullanır. Sunucu veritabanlarını fiilen değiştiren programdır. İstemci programları bunu doğrudan yapmazlar; bunun yerine, niyetinizi sunucuya yapılandırılmış sorgu dili (SQL) ile yazılmış sorgularla iletirler. İstemci yazılım veya yazılımlar kendisinden MySQL 'e ulaşmak istediğiniz makinelere yerel olrak yüklenirler, fakat istemciler bağlanabildiği sürece sunucu herhangi bir yere yüklenebilir. MySQL kalıtımsal ağlandırılmış bir veritabanı sistemidir, bu sayede istemciler makinenizde yerel olarak çalışan veya başka bir yerde çalışan bir sunucuya bağlanabilirler, hatta dünyanın öbür ucunda çalışan bir makineye. İstemciler birçok amaç için yazılabilirler, fakat her biri bağlanarak, veritabanı işlemleri gerçekleştirmek için SQL sorguları göndererek ve ondan sorgu sonuçları alarak sunucuyla etkileşirler [45].

MySQL' in tarihçesinden bahsedecek olursak, şuan MySQL’ in 6. sürümünün piyasaya çıkartılması için çalışmalar devam etmektedir, ilk olarak Michael Widenius ve David Axmark bu yazılıma 1994 yılında başlamışlardır. İlk sürüm 23 Mayıs 1995′de yayınlanmıştır, ardından 8 Ocak 1998 senesinde yazılımın ilk Windows uyarlaması Windows 95 ve NT sürümleri için yapılmıştır. İlk sürümün çıktığı tarihi ve Windows uyarlamalarını saymaz isek, MySQL ilk hali ile yaklaşık 5 sene kullanılmıştır. Versiyon 3.23 adı ile Haziran 2000′de beta sürüm çıkarılmış yapılan testler sonucunda Versiyon 3.23 sürümü Temmuz 2001′de yayınlanmıştır. Bu

sürümün hemen ardından Versiyon 4′ün beta çalışmalarına başlanılmış ilk beta sürüm Ağustos 2002′de duyurulmuş kararlı sürüm ise bundan 6 ay sonra Mart 2003′de çıkarılmıştır. 5 ay sonra yapımcılar 4.0.1 sürümünün beta çalışmalarını başlatmış, ağustos 2003 de beta sürüm yayınlanmıştır. Versiyon 4.1 ise Haziran 2004 de test edilmeye başlanmış, Ekim 2004 de yayınlanmıştır. 2005 yılında versiyon 5 için kolları sıvayan yapımcılar, Mart 2005 de beta, Ekim 2005 de ise kararlı sürümü yayınlamışlardır. Bu sürede yapımcı Sun Microsystems MySQL AB firmasını 26 Şubat 2008 yılında satın almıştır. Böylelikle MySQL' in geliştirilmesi için daha çok imkan bulunabilecek ve sürümler daha da iyi olabilecektir. Sun Microsystems' in MySQL AB'yi satın almasından sonra ilk sürüm 27 Kasım 2008′de çıkartılmış olan 5.1 sürümüdür [46].

3.3. Web Servisleri

Web 'in yaygınlaşmasından sonra farklı platformların birbirleriyle haberleşme ihtiyacı artmıştır. Bunun için web servis uygulamaları geliştirilmiştir.

W3C (World Wide Web Consortium) tarafından yapılan resmi tanımıyla web servisleri, bilgisayarlar arasında ağ üzerinden etkileşimi ve uyumluluğu sağlayacak yazılım sistemleridir. Günümüzde birbiriyle haberleşecek sistemleri gerçeklemek için en çok tercih edilen yöntem web servisleridir [47].

Bir web servisi, uygulamalar ve sistemler arasında veri alışverişi için kullanılan açık standartlar ve protokoller topluluğudur denilebilir. Çeşitli programlama dillerinde yazılan ve çeşitli platformlar üzerinde çalıştırılan yazılım uygulamaları, internet gibi büyük bilgisayar ağları üzerinde veri alışverişi yapmak için web servislerini kullanabilirler. Asıl fikir, servisleri internet üzerinde dağıtmak ve onları istemciler için ulaşılabilir hale getirmektir. Yani haberleşecek sistemlerin birbirlerinden haberdar olması veya platformlarının uyumlu olması gerekmez. XML Web Servislerine farklı bir bilgisayar ve farklı bir platformdan istemci olunabilir. Bu birlikte çalışabilirlik özelliği (Java ve Python ya da Windows ve Linux uygulamaları gibi) açık standartların kullanımı sayesindedir.

Web servisler çok amaçlı kullanılabilirler. Öncelikli amaçlar arasında, farklı sistemlerin birbirine entegrasyonu – mesajlaşması bulunmaktadır. Veri alışveriş yöntemine ait standartlar olduğundan dolayı web servisler platform bağımsızdır. Yani, Net ile yazılan bir web servis Java uygulamalarında kullanılabildiği gibi, Java ile yazılmış bir web servis aynı mantıkla Net uygulamalarında ve ASP. Net web sitelerinde kullanılabilir.

Web Servislerinin 2 farklı kullanım şekli vardır:

1. Uygulamaların sıkça ihtiyaç duydukları işler vardır. Bunları sürekli tekrar yapmaktansa servisler tarafından yapılıp uygulamalardan çağrılırlar. Örneğin hava durumunu anlık olarak öğrenen program gibi.

2. Başka platformda çalışan bir uygulama ile haberleş ilerek veri alış verişinde bulunulur.

Web servis altyapısında iki popüler teknoloji kullanılmaktadır; RESTFUL (Representational State Transfer -Temsili Durum Transferi) ve SOAP **(**Simple Object Access Protocol - Basit Nesne Erişim Protokolü).

3.3.1. SOAP (Simple object access protocol-basit nesne erişim protokolü)

SOAP (Basit Nesne Erişim Protokolü) dağıtık uygulamalarda ve web servislerinin haberleşmesinde kullanılmak üzere tasarlanan, RPC (Remote Procedure Call) modelini kullanan, istemci/sunucu mantığına dayalı bir protokoldür. Daha genel olarak SOAP, web üzerinden fonksiyonları kullanmak için geliştirilmiş bir sistemin XML tabanlı kurallar topluluğudur. SOAP ile ilgili bütün mesajlar XML formatında iletilir ve temel olarak bir SOAP mesajı 3 şekilde oluşabilir.

1. Metot Çağırımı
2. Cevap Mesajı
3. Hata Mesajı

Aşağıda basit bir SOAP mesaj yapısı yer almaktadır;

```
<SOAP-ENV:Envelope xmlns:SOAP-
ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header>
    ....
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
    ....
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

3.3.2. REST (Representational state transfer - temsili durum transferi)

REST web standartlarını ve HTTP protokolünü baz alan bir mimaridir. İlk olarak 2000 yılında Roy Fielding tarafından tanımlanmıştır. REST istemci-sunucu iletişimiyle ilgili bir mimari. HTTP protokolü ile paralel olarak gelişmiş olmasının yani sıra bugün en çok hepimizin aşina olduğu World Wide Web sisteminde kullanılıyor. REST mimarisini kullanan servislere genel olarak RESTFUL servis deniyor. Ana fikir aslında istemci-sunucu arasında ki veri alışverişini SOAP, RPC gibi kompleks mimarilerle sağlamak yerine, HTTP protokolü üzerinden sağlamak. Çünkü zaten World Wide Web dediğimiz yapı HTTP protokolü üzerine kurulu. RESTFUL servisler SOAP, RPC' nin aksine basit ve hafiftirler.

Rest mimari HTTP protokolleri arasından PUT, GET, POST, DELETE metotlarını kullanır;

- Bir web sunucusunda bulunan kaynağı (resource) edinmek için GET komutunu kullanır. Bu kaynak JPEG, HTML, PDF olabilir. GET metodu ile kaynak değiştirilemez.
- REST mimarisinde POST metodu yeni bir kaynağın oluşturulması (create) için kullanılır. Örneğin POST metodunu kullanarak sanal bir kütüphane sistemine yeni bir kitap ekleyebilir.
- REST mimarisinde PUT mevcut bir kaynağı değiştirmek (UPDATE) için kullanılır. Örneğin sanal bir kütüphanedeki bir kitabın bilgilerini değiştirebiliriz.

- REST mimarisinde DELETE metodu mevcut bir kaynağı silmek için kullanabiliriz. Örneğin sanal bir kütüphanedeki bir kitabı silmek.

Bu iki teknolojinin en temel farklılıkları SOAP' ın veri isteklerini HTTP DATA kısmında halletmesi. RESTFUL web servislerinde ise istekler HTTP HEADER kısmında halledilir. Bunu sağlamak için HTTP protokolünde bulunan POST,PUT,GET,DELETE parametreleri kullanır. Açıkçası bu da RESTFUL web servislerinin en önemli özelliğidir.

Genel olarak farkları ise;

- REST, SOAP' a göre daha hızlı ve hafif bir yöntem, çünkü REST' de ekstradan XML tanımları yok.
- REST bir servise erişmek nispeten daha basit. Sadece browser dan çağırdığınızda bile tam sonucu alabilirsiniz.
- REST daha az trafik harcar.
- Veriler REST' de işlenmeden de okunabilirliği yüksek olur.
- SOAP' ta WSDL' den otomatik Proxy Class oluşturulup, istemci uygulamaya kolay entegre edilebilir.
- REST' de istemci uygulamaya entegre etmek için Proxy Class yazılması gerekir.
- SOAP tip korumalıdır. SOAP Header' da belirtilen şekilde veri gelir.
- REST' de veri tanımlaması yoktur.
- SOAP alt yapısında XML kullanır (JSON veri gömülebilir).
- REST verileri XML veya JSON olarak gönderebilir.
- REST, veritabanına uzaktan erişim gibi çalışır. SOAP ise uzaktan metot çağırma(RPC) ilkesine dayanır.
- REST' i uygulamak için URL etkin şekilde kullanılır ve URL tasarımı yapılması gerekir.
- SOAP' ta ise uzaktan metot çağrıma tekniği kullanılır.

Yukarıda listelediğim RESTFUL ve SOAP servislerinin farklarını göz önünde bulundurarak RESTFUL servislerinin avantajları daha fazla olduğu için

uygulamamda, onu seçtim. Ve servislerde veri alış verişi için JSON biçimlendirme yöntemini kullandım.

JSON (JavaScript Object Notation), XML' e alternatif olarak kullanılabilen JAVASCRIPT tabanlı veri değişim formatıdır. JSON' un temel amacı veri alış verişi yaparken daha küçük boyutlarda veri alıp göndermektir. Programlama dilinden bağımsızdır. İnsanların okuyup yazabilmesi kolaydır. Bu özellikleri sayesinde JSON ile çok hızlı web uygulamaları oluşturabilir.

3.4. Java Development Kit Kurulumu

Android İşletim Sistemi Java tabanlı olduğu için ilk olarak Java JDK (Java Development Kit) Oracle firmasının resmi sitesinden indirilerek kurulmalıdır. Kurulumu tamamladıktan sonra JAVA_HOME ortam değişkenini oluşturulup JDK' nın kurulduğu adresin değer olarak verilmesi gerekmektedir.

Şekil 3.3. JDK seçimi [48]

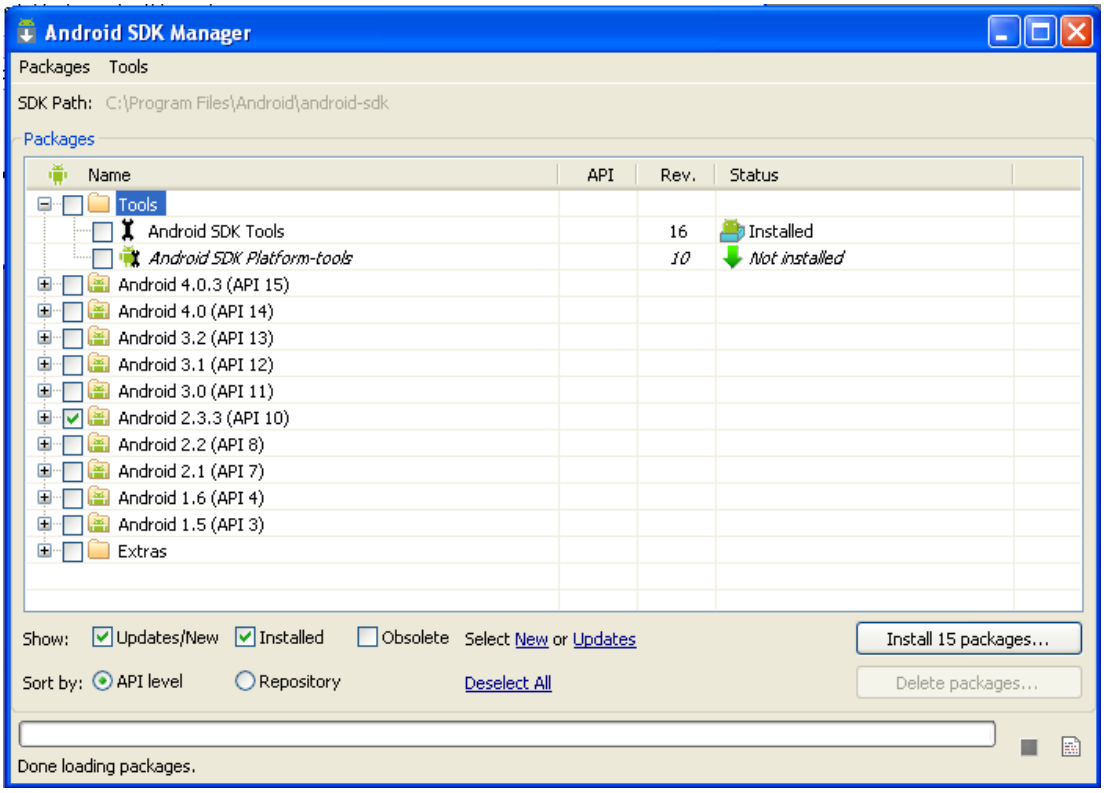
3.5. Eclipse Kurulumu

Android programlama için sadece editör olarak Eclipse programı yoktur. Yalnız Google firması Eclipse' i önerdiği ve uygulama geliştirmeyi kolaylaştıracak birçok eklenti sunduğu için bu editör seçilmiştir. Eclipse IDE for Java EE Developers 'ın versiyonu tercih edilmiştir, daha fazla IDE sağladığı için. Ayrıca Eclipse daha sonra mevcut editöre "Install New Software " seçeneği sayesinde istenilen IDE' yi ekleme imkanı sunar. Eclipse editörü portable (taşınabilir) olduğundan indirilen zip dosyası istenilen dizinde açıp "exlipse.ex"e uygulamasını açarak kullanıma hazır hale getirilir.

Şekil 3.4.Eclipse açılış sayfası

3.6. Android SDK Kurulumu

Android SDK 'sını "http:developer.android.com/sdk/index.html" adresinden indirip, zip dosyası uygun bir klasöre açılır ve "SDK Manager.exe" dosyası çalıştırılır. Ve Manager aracılığıyla hangi Android platformu için uygulama yazılacaksa onun API' si indirilir.

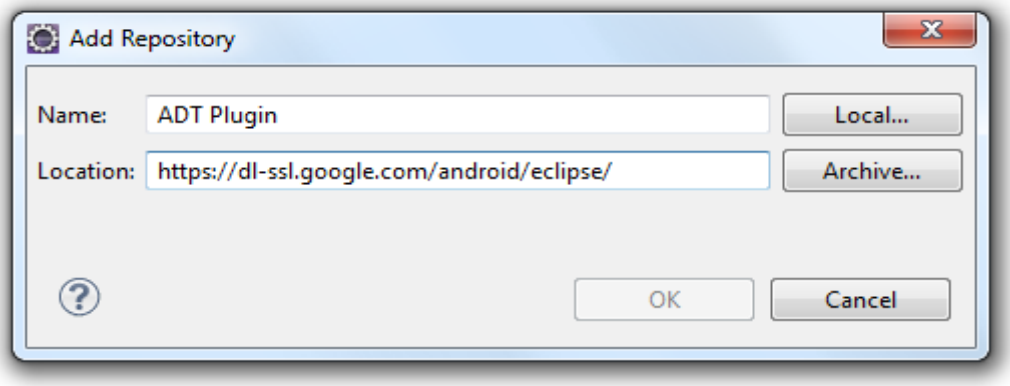


Şekil 3.5. Android SDK manager sayfası

3.7. Android Development Tools (ADT) Eklentisi

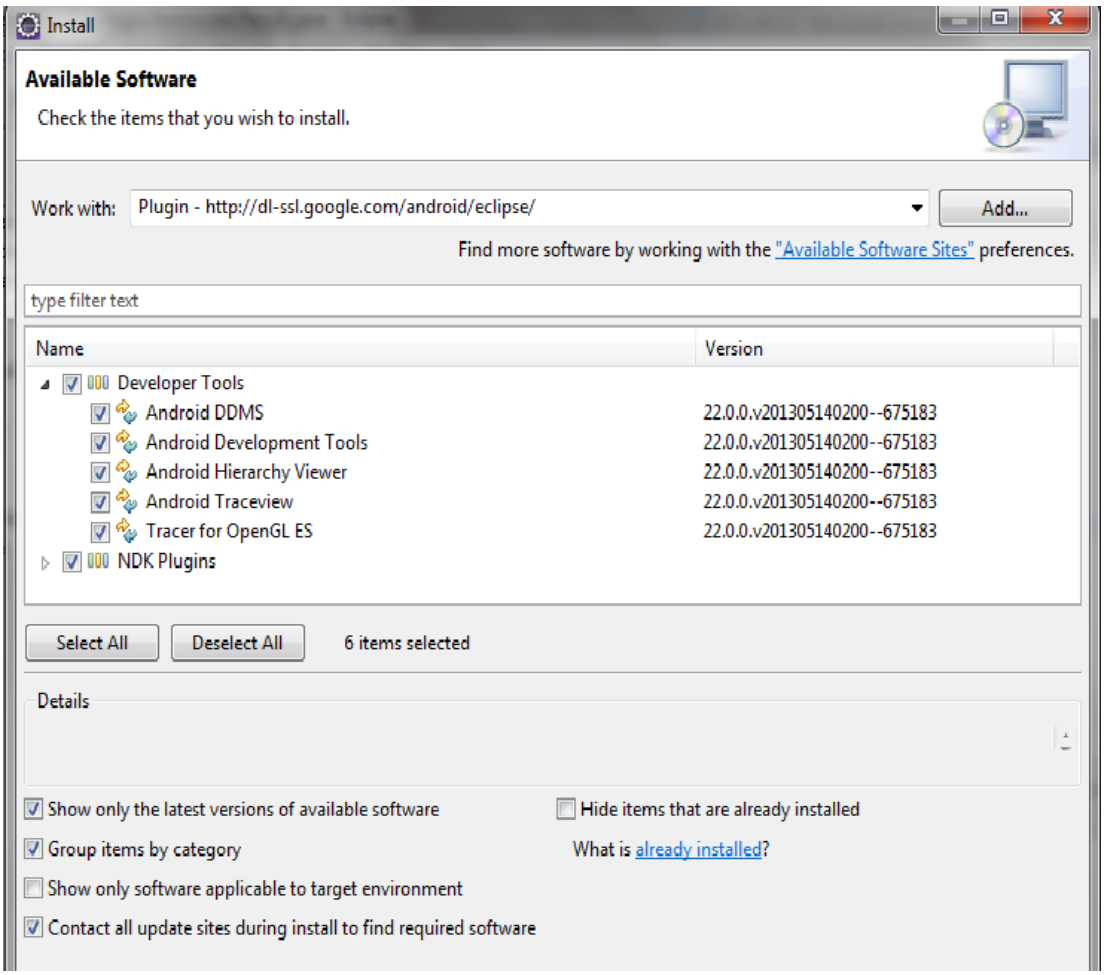
Google, Android uygulamaları için Android Development Tools (Android Geliştirme Araçları) adında bir eklenti geliştirmiştir. Bu eklenti sayesinde ara yüz elemanlarını sürükle bırak şeklinde oluşturulabilir, uygulama konfigürasyonunu XML dökümanıyla uğraşmaksızın kolayca değiştirilebilir ve ayrıca uygulamalar debug (hata ayıklama) edilebilir.

ADT eklentsini kurabilmek için Eclipse'den Help > Install New Software bölümüne tıklıyoruz. Açılan pencereden Add butonunu tıklayarak gelen ekranda name yazan yere bir isim verip, Location kısmına eklentimizin adresini https://dl-ssl.google.com/android/eclipse/ yazıp Ok diyoruz.



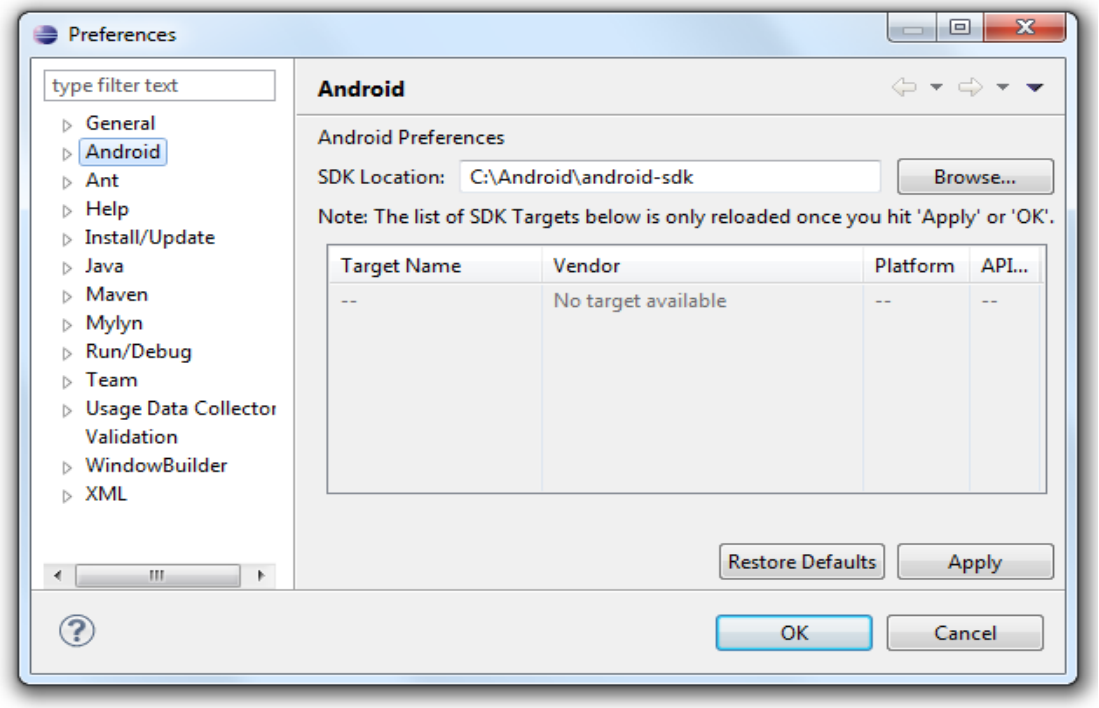
Şekil 3.6. Plug-in ekleme sayfası-1

Verdiğimiz linkten çektiği geliştirme araçlarını kuruyoruz.



Şekil 3.7. Plug-in ekleme sayfası-2

Eklentiyi kurduktan sonra Eclipse'i yeniden başlatıyoruz. Son olarak Eclipse 'e SDK 'nın yerini göstermeliyiz. Bunun için Window (pencere) menüsünden Preferences (seçenekler) ekranını açıp Android menüsünün altında SDK Location (SDK Konumu) bölümüne SDK 'nın kurulu olduğu adresi yazıyoruz.



Şekil 3.8.Eclipse' in tercihler sayfası

3.8. Sanal Cihaz Oluşturma

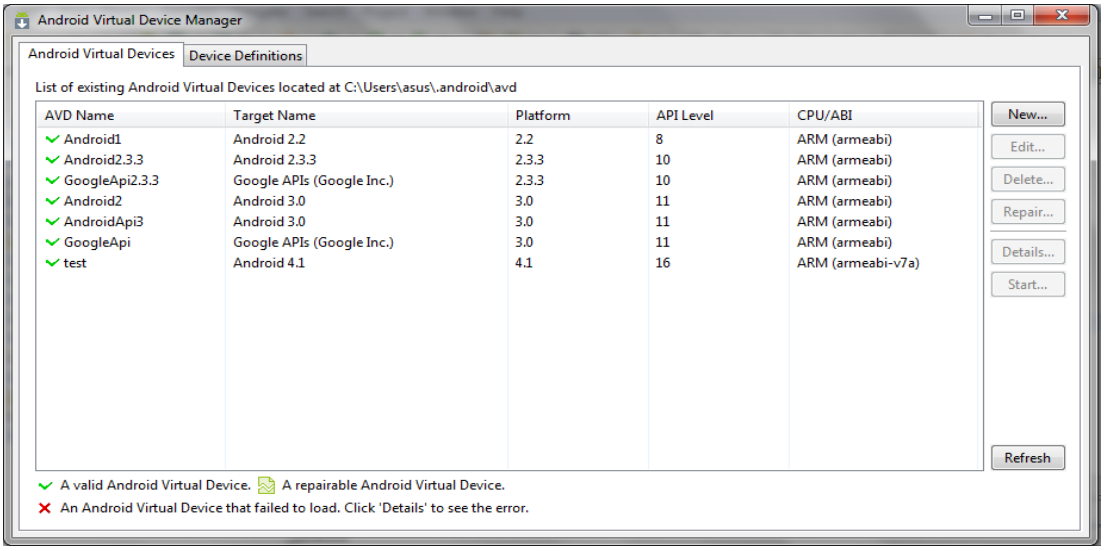
Android SDK bilgisayarınızda çalışan bir sanal mobil cihaz emülatörü içerir. Emülatör, prototip geliştirmenizi ve fiziksel bir cihaz kullanmadan Android uygulamaları geliştirip test etmenizi sağlar. Android emülatörü, tipik bir mobil cihazın telefon görüşmeleri haricinde, donanım ve yazılım tüm özellikleri taklit eder. Fare veya klavyenizi kullanarak "basıp" uygulamanız için olaylar oluşturabileceğiniz, çeşitli yöngüdüm ve kontrol tuşları sunar [49].

Şekil 3.9. Emülatör' ün ekran görüntüsü

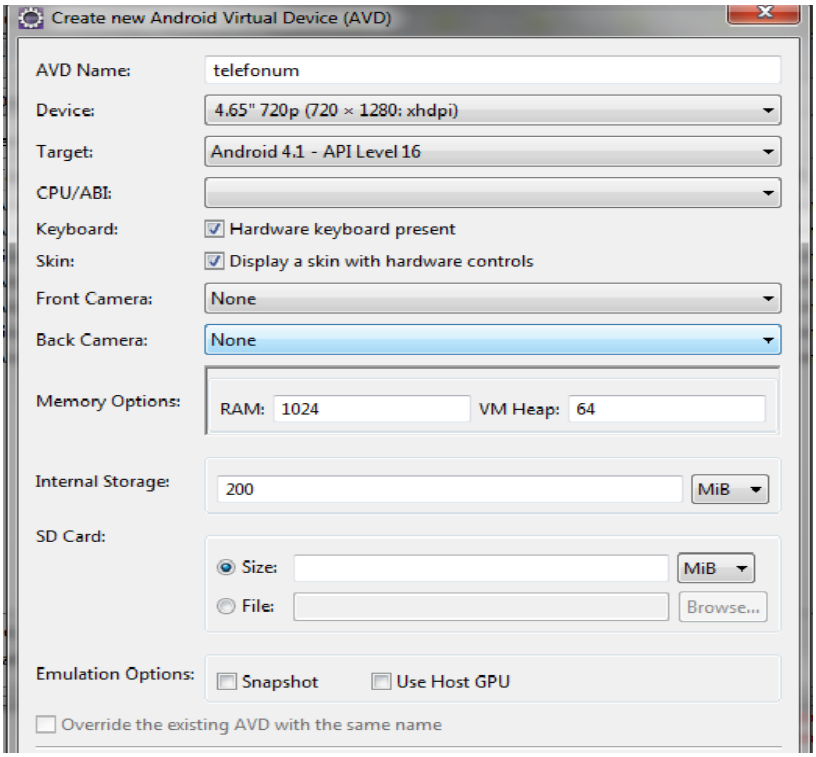
Android emülatörü mobil cihazlarda bulunabilecek birçok donanım özelliğini destekler [49];

- An ARMv5 İşlemci ve Bellek Yönetim Birimi
- A 16-bit LCD ekran
- Bir veya iki klavye
- Giriş ve çıkış özelliklerine sahip bir ses yongası
- Flash bellek bölümleri (geliştirilen cihazdaki disk imaj dosyalarıyla sanallaştırılır.)
- Bir GSM modem , (simüle SIM Kart dahil olmak üzere)
- Sensörler, (USB bağlantılı bir Android cihazdan verileri kullanarak)

Android Sanal Makine Yöneticisi aracılığıyla emülatör oluşturup, konfigürasyonu yapabilir. Emülatör oluşturmak için Eclipse >Window>AVD Manager menüsünü açılır. Sağ taraftaki New (Yeni) butonuna basarak sanal makinenin özellileri seçilerek sanal makine oluşturulur.



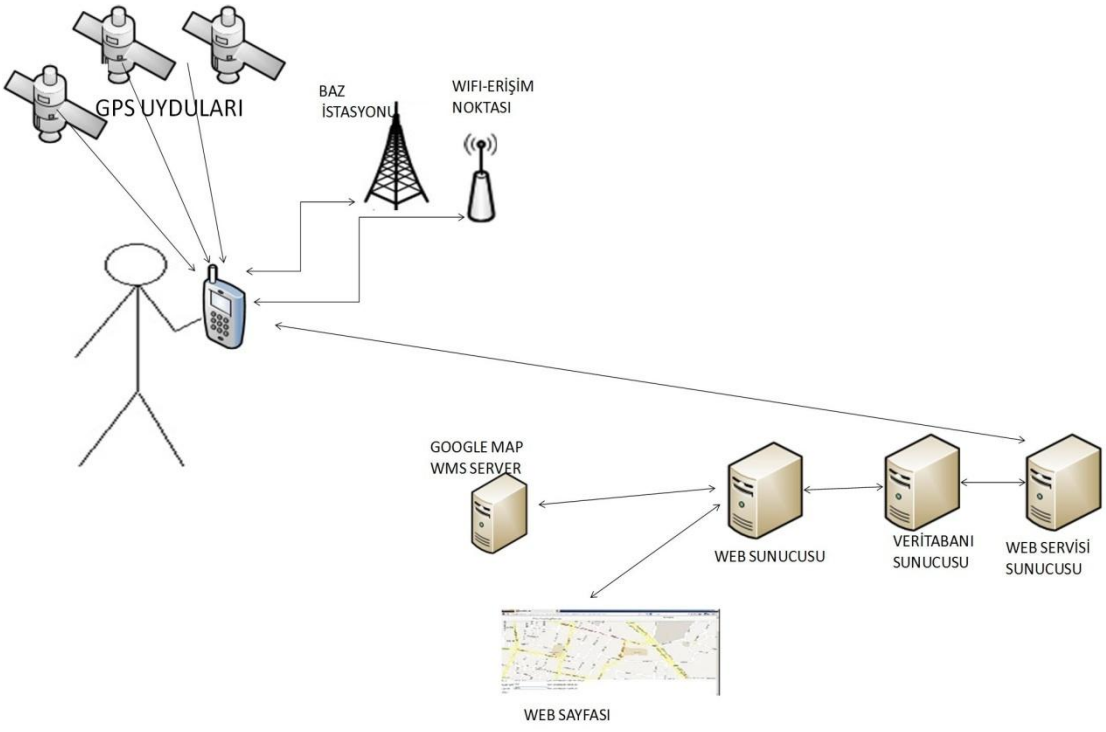
Şekil 3.10. AVD manager



Şekil 3.11. Sanal cihaz oluşturma ekranı

BÖLÜM 4. SİSTEMİN GERÇEKLENMESİ

Sistem 3. bölümde de değinildiği gibi Android İşletim Sistemine sahip bir telefona yüklenilecek bir adet mobil uygulama, çocuğun konumunu takip edebilmek için bir web sayfası, verilerin depolandığı (çocuğun bilgileri, ailenin bilgileri, çocuğun konumları, ihlaller, sınırlar) bir veri tabanı ve mobil uygulamayla veri tabanı arasında veri alış verişini yöneten bir adet web servisi olmak üzere 4 alt bölümden oluşmaktadır. Sistemin tasarımı aşağıdaki Şekil: 4.1 'de ki gibidir.

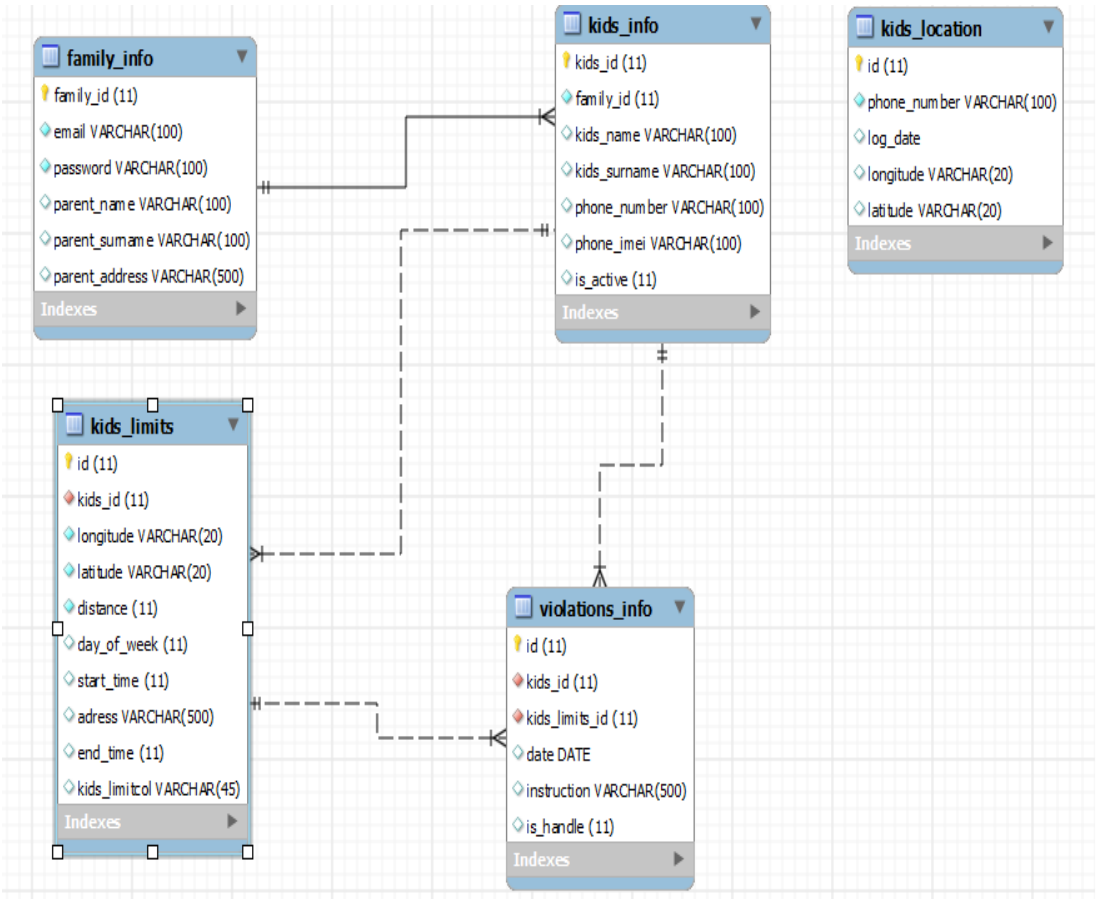


Şekil 4.1. Sistem tasarımı

4.1.Veritabanı Oluşturulması

Sistemde MySQL server ücretsiz ve web projelerinde kullanım kolaylığı açısından çok tercih edilen veritabanı yönetim aracı olduğu için veritabanı olarak MySQL Server 5.6 sürümü kullanılmıştır.

Sistem "kidstrackingdb" adında beş tablodan oluşan ilişkisel bir veritabanına sahiptir ve diyagramı aşağıdaki şekilde verilmiştir.



Şekil 4.2. kidstrackingdb veritabanı diyagramı

İlk tablo "family_info" tablosudur. Ailenin bilgilerini depolaması için oluşturulmuştur. Ebeveynin web adresine girip kaydolduğunda oluşturduğu email, şifre, isim, soy isim, adres ve aileyi tanımlayan birincil anahtar olarak atanan, otomatik artan family_id' yi tutan tablodur.

"kids_info " tablosu çocuğun bilgilerini tutmak için oluşturulmuştur.

"kids_location" tablosu ise çocuğun konum bilgilerini tutmak için oluşturulmuştur. Bu tabloda çocuğun hangi saatte nerede olduğu enlem ve boylam bilgileriyle tutulmaktadır. Yukarıdaki diyagramda bu tablonun diğer tablolarla ilişkisel olmadığı gözükmektedir. Ancak "phone number" unic bir değer olarak düşünülüp işlem yapılmaktadır. Çünkü bir id (çocuğa) 'ye ait bir telefon numarası olduğu düşünülmektedir.

"kids_limit" tablosu ise ebeveynin web sitesi üzerinden belirlediği, hangi gün ve hangi saatler arasında çocuğun nerede olması gerektiği sınırlamalarının tutulduğu tablodur.

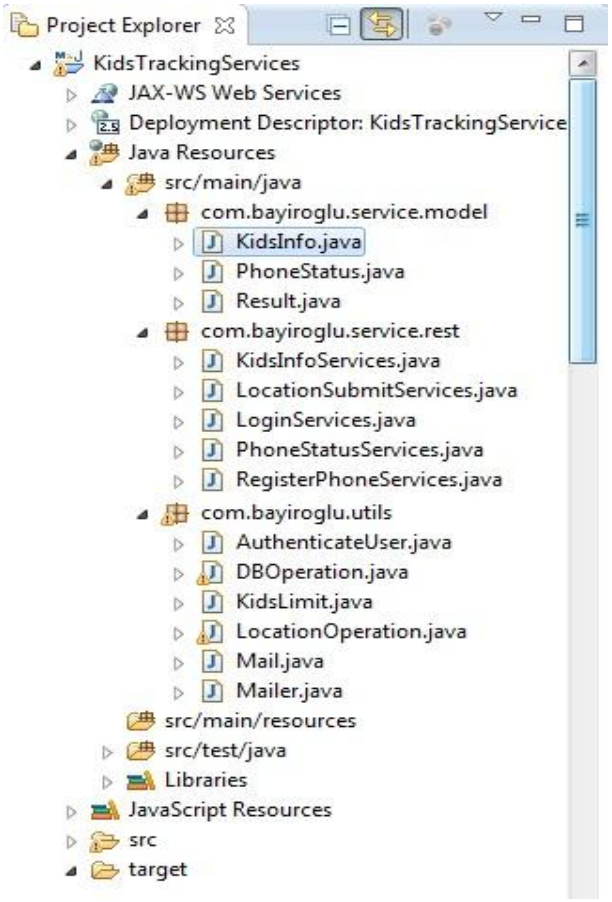
"violation_info" tablosu ise çocuğun sınır ihlallerinin tutulduğu tablodur.

4.2. Web Servis

Web servisleri yazılırken daha hızlı ve uygulanabilirliği daha kolay olduğu için Rest mimarisi seçilmiştir. Web servisi, mobil uygulama ve veritabanı arasında veri alış verişini yönetecek şekilde tasarlanıp ve Eclipse Indigo editöründe Maven projesi olarak geliştirilmiştir.

Apache Maven teknolojisinin ne olduğundan bahsedecek olursak, geliştirilen Java projelerinin geliştirme süreçlerini basitleştirmek, standartlaştırmak, dokümantasyon hazırlamak, kütüphane bağımlılıklarını kullanıcı zahmetinden kurtarmak için kullanılan bir araçtır. Maven ile hemen hemen bütün Java projelerinin iskeleti oluşturulabilir. Bu iskelet standartlaştığı için, siz oluşturduğunuz projeyi geliştirme ortamınızdan bağımsız olarak geliştirirsiniz. Proje iskeleti standart olduğu için Java projenizi ister Eclipse ile ister Netbeans ile isterseniz başka idelerle geliştirebilirsiniz. Ayrıca Java projeleriyle uğraşırken başınızı ağrıtan jar kütüphaneleriyle kesinlikle haşır neşir olmazsınız.

Web servisinin mimarisini oluşturan sınıfların Project Explorer da ki görüntüsü aşağıdaki gibidir.



Şekil 4.3. Web Servis Sınıfının Mimarisi

KidsTrackingServices sınıfı 3 tane paketten oluşmaktadır. Projede kullanılan paketler, paketlerin sahip olduğu sınıflar ve özellikleri şu şekildedir:

1. com.bayiroglu.service.model: Servis çağrılması sonucunda dön derilecek sınıfların tutulduğu tablodur. Rest servislerde sonuç olarak dön derilecek değer sistemde tanımlı olan veri türlerinden (int, String, boolean, double) bile olsa dönen değer için bir sınıf tanımlamak daha sağlıklıdır. Servis, mobil uygulamayla haberleşeceği için bu paketin altındaki sınıfların aynısı mobil uygulamada da vardır.

    - KidsInfo.java: Çocuk bilgilerini tutmak için oluşturulmuştur.

- PhoneStatus.java: Telefonun sistemde kayıtlı olup olmadığı sonucu tutan sınıftır.
- Result.java: Sonuş olarak true, false değerleri döndürmek için oluşturulmuş bir sınıftır.

2. com.bayiroglu.utils: uyguma içinde çok sık kullanılan veritabanı ve diğer işlemler için kullanılan fonksiyon ve procedürlerin tutulduğu pakettir.

    - AuthenticateUser.java: Bu sınıfta servisler için kullanıcı adı ve şifre oluşturulur. Servisi her çağıran sınıf için kullanıcı adı ve şifre doğrulamasını yapar. Yoksa isteyen herkes servislere ulaşabilir. Bu bir güvenlik önlemidir. Servislere Header olarak geçirilir.
    - DBOperation.java: Veritabanı ile ilgili işlemlerin yapıldığı sınıftır.com.bayiroglu.service.rest paketindeki Rest servisler içinde veritabanı işlemlerini yapan fonksiyonların bulunduğu sınıftır. Yani diğer sınıflarda veritabanında yapmak istedikleri değişiklikleri bu sınıfın fonksiyonları aracılığıyla yaparlar.
    - KidsLimit.java: Çocukların konum sınırlama bilgilerini tutan sınıftır. Bu sınıf DBOperation.java sınıfının içinde çocuğun limitlerini veritabanına yazmak hem de bulunduğumuz an içinde çocuk için sınırlama var mı onu kontrol etmek için kullanılır.
    - LocationOperation.java: Geolocation verisini yani enlem ve boylam bilgisini adrese çeviren ve çocuğun girilen adresler içinde olup olmadığını döndüren fonksiyonları tutan sınıftır. Bu sınıfta, DBOperation.java sınıftaki getKidsLimitForCurrentTime() metoduyla çocuğun o anki konumunu alıp bu sınıfın isViolate() metodu aracılığıyla ebeveynin belirlediği sınırların içinde mi onu test eder. Bu testi gelen koordinat bilgileriyle ebeveynin belirlediği sınırların arasındaki uzaklığı iki nokta arasındaki uzaklık formülünden bulup mesafenin ebeveynin belirlediği mesafeden küçük olup olmadığını kontrol eder.

- Mail.java: Mail yapısını tutan sınıftır. Mail kime gönderileceğini, kimden gideceğini, başlığını ve konusunu parametre olarak alıp daha sonra kullanılmak üzere set eden sınıftır.
- Mailer.java: Mail göndermek için kullanılan sınıftır.

3. com.bayiroglu.service.rest:Servislerin tutulduğu pakettir.

- KidsInfoServices.java: kidsinfo (çocuğun bilgilerini) değerini kullanıcıdan aldığı IMEI ve telefon numarasına göre çocuk bilgilerini veritabanından okuyarak servise verir, serviste istek sonucu olarak döner.
- LocationSubmitService.java: Telefondan gelen konum bilgisini alarak veri tabanına yazar. Servise istemci tarafından telefon numarası, enlem, boylam ve tarih bilgileri geçirilerek gönderilir. LocationOperation.java sınıfı aracılığıyla çocuğun bulunduğu konum yasaklı bölge mi diye kontrol ediyor. Yasaklı olduğu değeri dönmüşse DBOperation.java sınıfında ki isViolationHandle() metodu aracılığıyla, veritabanında ki violation_info tablosunda bu yasağın yakalanıp yakalanmadığına bakıyor. Yakalanmamışsa mail sınıfını çağırıyor. Gereksiz trafik olmaması açısından bir saatte bir mail atıyor, çünkü bir adımda bile konum değişebilir ve sürekli mail atması gereksiz trafiğe sebep olup mailin sahibi ebeveyni rahatsız edebilir.
- LoginServices.java: Mobil uygulama tarafında uygulamaya giriş yapılırken girilen e-mail ve şifreyi doğrulamak için kullanılan servis sınıfıdır.
- PhoneStatusServices.java: Mobil uygulama tarafında kullanıcı adı ve şifreyle giriş yapıldıktan sonra, telefon giriş yapan kullanıcı için tanımlımı, tanımlanmış fakat aktif edilmemiş mi yada tanımsız mı o bilgiyi tutar. Eğer durum değeri -1 ise; telefon numarası aile tarafından tanımlanmamıştır, 0 ise; telefon numarası aile tarafından tanımlanmış ama henüz telefona uygulama kurulup aktif hale getirilmemiştir, 1 ise telefon numarası aile tarafından tanımlanmış ve aktiftir.

- RegisterPhoneServices.java: PhoneStatusServices sonucu dönen değer 0 ise telefona ilk defa giriş yapılıyordur ve DBOperation.java sınıfında ki registerPhone() metoduyla sistem tarafından telefon IMEI numarası çekilip, veri tabanına kaydedilir. Ve program o telefon için aktif hale gelir.

4.3. Mobil Uygulama

Konum bilgileri mobil geliştirme dünyasında giderek önem kazanmaktadır. Uygulamalarda konum verilerini kolayca elde edilip işlenebilmesi günümüz mobil platformlarının önemli bir özelliği haline geliyor. Android bu işlevselliği konumu hizmeti ile sağlar.

Android bir uygulamaya, konum bilgisini sağlamak için farklı metotlar sunar. Bu metotlar konum sağlayıcılar (location providers) olarak adlandırılır ve hepsinin kendine özel güçlü ve zayıf yönleri vardır. Bu sistemde gerçeğe yakın konum verisini elde edebilmek için bütün konum sağlayıcıları bir arada kullanarak zayıf yönlerinin etkilerini azaltmaya çalışılmıştır. Konum sağlayıcılar; GPS algılayıcı ve Network algılayıcıdır. GPS mevcut konumu belirlemek için en iyi tahmini veren güzel bir teknolojidir fakat kapalı ortamlarda, gökyüzünü tam göremediği ortamlarda istenilen sonuç elde edilememektedir. Network tabanlı konum sağlayıcılar cell id (baz istasyonu) 'ler ve kablosuz ağ erişim noktaları aracılığıyla konum bilgisi elde ederler. GPS'in kapalı veya konum bilgisini alamadığı alanlarda cihazlara konum bilgisi elde etmesini sağlar.

Wi-Fi tabanlı konum bulma bir cihaza erişebileceği Wi-Fi erişim noktalarını ve o erişim noktalarının o anki sinyal güçlerini buldurur, sonra cihaz Google konum hizmetine (Android konum hizmetinden farklı) sorgu yapar, bu da Wi-Fi bilgisine dayalı olarak konum bilgisi sağlar. Cihaz tarafından toplanan Wi-Fi bilgisinde erişim menzili içinde olan Wi-Fi erişim noktalarının zorunlu erişim denetimi (MAC) adresleri ve o erişim noktalarından alınan sinyalin gücü bilgisi de bulunur. Görünür Wi-Fi erişim noktaları bilgisine dayalı olarak, Google konum hizmeti Wi-Fi erişim noktaları ve konumları hakkında bilgi edinmelidir. Bu bilgi, kullanıcı Konum

Ayarları ekranından Google konum hizmetini aktifleştirdiğinde Android cihaz tarafından toplanmaktadır.

Android ve Google konum hizmeti birlikte çalışarak Wi-Fi bilgisine benzer bir şekilde baz istasyonlarının ID'lerini konum bilgisiyle eşleştirir. Bir cihaz ağ sağlayıcısını kullanmak üzere ayarlandıktan sonra, görünür kablosuz ağlar ve mevcut baz istasyonu ID'si üzerine veri toplar. Baz istasyonları için, bu veride cihazın o anda bağlı olduğu baz istasyonu ve cihazın o an ki GPS konumu da vardır. Bu bilgiyle, Google konum hizmeti baz istasyonlarının konumlarını da içeren baz istasyonlarının bir haritasını geliştirebilir.

Bir cihaz o anki konumunu bulması gerektiğinde, o an bağlı olduğu baz istasyonunun ID'sini, aynı zamanda önceden kullandığı baz istasyonlarının geçmiş bilgisini Google konum hizmetine gönderir. Bu bilgiyle, Google konum hizmeti baz istasyonu ağı hakkında sahip olduğu bilgiye bağlı olarak cihazın konumu hakkında bilgi sağlayabilir. Eğer birden fazla baz istasyonunun ID'si Google konum hizmetine gönderilirse, üçlü kestirim kullanarak daha yüksek hassasiyet sağlayabilir. Cihaz yalnızca bir baz istasyonu ID'si gönderirse Google konum hizmeti üçlü kestirim yapamaz.

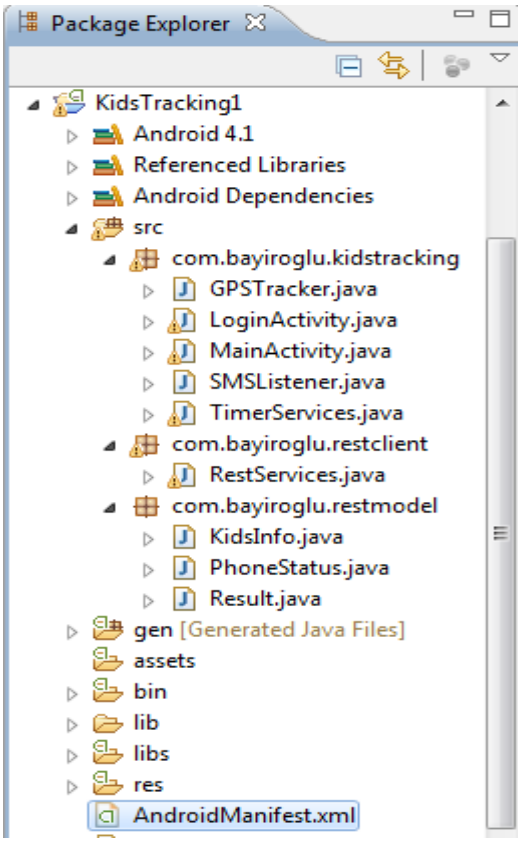
Android'te konum bazlı işlemler için kullanılacak sınıflar "android.location" paketi içerisinde bulunur. Konum servisleri bir Manager (Yönetici) sınıfı ile yönetilir ve bu sınıfın adı "LocationManager" dır. Konum algılama, mevcut konum sağlayıcıların bulunması vb. işlemlerin yönetimi bu sınıf üzerinden yapılır. Konum servisine aşağıda ki gibi erişilebilir;

```
LocationManager manager=( LocationManager) getSystemService
(Context.LOCATION_SERVICE);
```

Mevcut konumu elde edebilmek için, "LocationManager" sınıfının "requestLocationUpdates() " metodu kullanarak konum güncellemelerinden haberdar olmak istediğinizi belirtmeniz gerekmektedir. Güncellemeleri dinleyebilmek için de bir "LocationListener" arayüzü oluşturulmalıdır.

Projenin Mobil kısmı Eclipse Indigo editöründe Android projesi olarak geliştirilmiştir. Mobil uygulama kısmındaki hedef konum bilgilerinin elde edilip veritabanına işlenmek üzere kaydedilmesidir.

Mobil uygulamanın mimarisini oluşturan sınıfların Project Explorer da ki görüntüsü aşağıdaki gibidir.



Şekil 4.4. Web Uygulamasının Sınıf Mimarisi

KidsTrackingServices sınıfı 3 tane paketten oluşmaktadır. Projede kullanılan paketler, paketlerin sahip olduğu sınıflar ve özellikleri şu şekildedir:

1. com.bayiroglu.restmodel: Servisler kısmında da belirttiğim gibi mobil uygulama,servis ile haberleşeceği için bu paketin altındaki sınıfların aynısı servis kısmında da vardır.

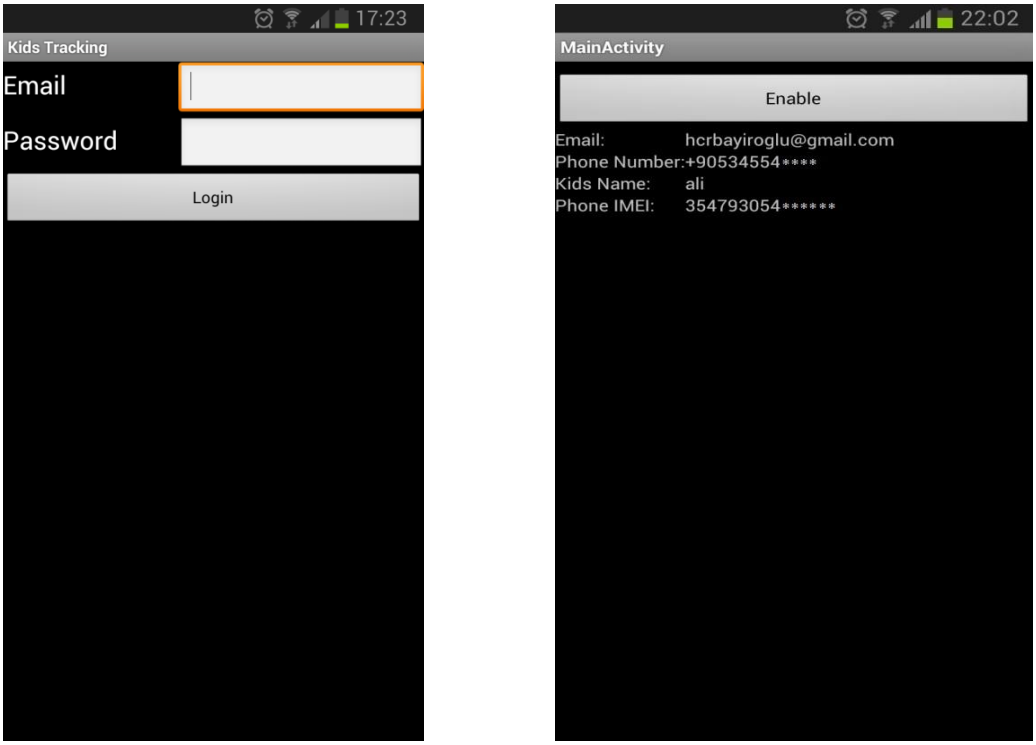
    - KidsInfo.java: Çocuk bilgilerini tutmak için oluşturulmuştur.

- PhoneStatus.java: Telefonun sistemde kayıtlı olup olmadığı sonucu tutan sınıftır.
- Result.java: Sonuç olarak true, false değerleri döndürmek için oluşturulmuş bir sınıftır.

2. com.bayiroglu.kidstracking: aktivitelerin ve timer servisinin, ve location listener'ın tanımlı olduğu yerdir.
    - LoginActivity.java: Uygulama ilk çalışmaya başladığında bu sınıf çalıştırılır. Kullanıcının doğru şifre girip girmediği, telefonun aktif olup olmadığı burada kontrol edilir. Kullanıcının girdiği bilgiler doğru ise buradan ikinci aktivite olan main aktivite çalıştırılır.
    - MainActivity.java: Giriş işleminden sonra çalıştırılan aktivitedir. Bu aktivitenin içinde telefonun sahibi olan çocuğun bilgileri servisler kullanılarak veri tabanından alınır ve ekranda gösterilir. Burada ki buton aracılığı ile ; Location listener ve sms listener enable ve disable edilebilir.
    - GPSTracker.java: Network bilgisini ve GPS bilgisini kullanarak konum bilgisini getiren sınıftır.
    - TimerServices.java: Programlamada thread yapısında olan servistir 10 saniyede bir konum bilgisini servis aracılığı ile veri tabanına iletir. Ayrıca telefonda internet olup olmadığı kontrolu de buradan yapılmaktadır. Eğer internet yoksa konum verileri telefonda bir txt dosyasına kaydedilmektedir. Bu kontrolü "postlocation" fonksiyonu yapmaktadır. "postlocation" fonksiyonu çalışmadığı zaman, lokasyon bilgisi dosyaya yazılıyor. "TimerServices" sınıfı "postlocation" fonksiyonu çalıştığı zaman dosyada kayıt var mı diye kontrol ediliyor ve varsa onları da servis aracılığı ile veri tabanına gönderiyor.
    - SmsListener.java: Telefona gelen mesajları dinleyip, gelen yazı "nerdesin" sorusuyla adresi gönderen sınıftır.

3. com.bayiroglu.restclient: Web servisi tarafında oluşturulan servisleri çağırmak için bulunan fonksiyon ve prosedürlerin tutulduğu RestServices sınıfını tutar.

Mobil uygulamanın ekran görüntüleri aşağıda verilmiştir;



Şekil 4.5. Mobil Uygulamanın Ekran Görüntüleri

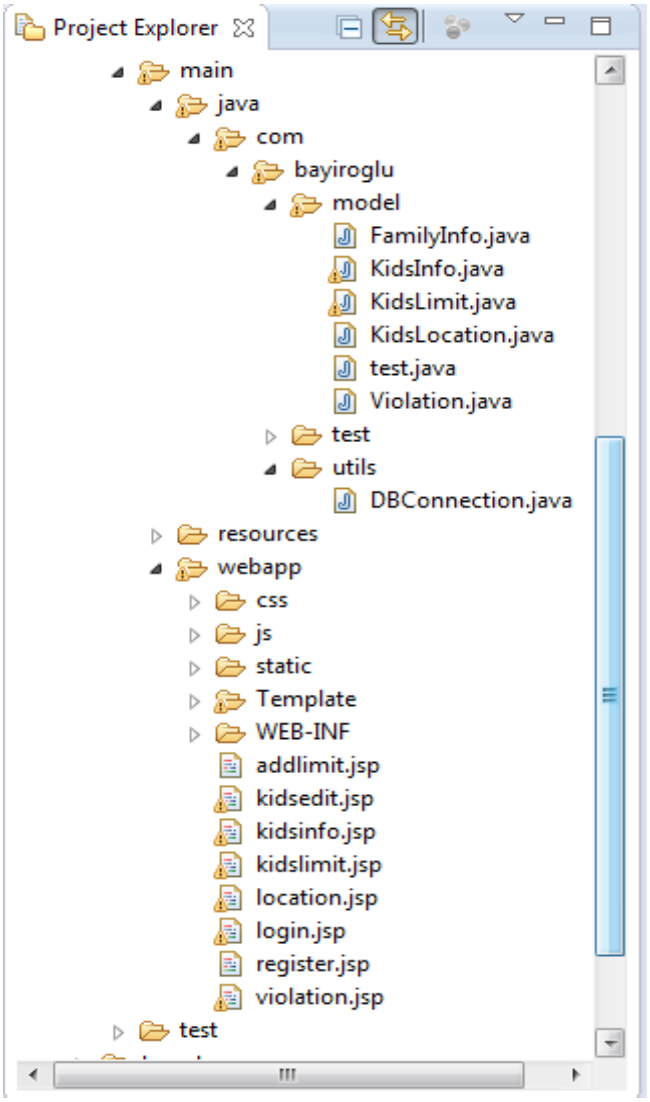
4.4. Ara Yüzü

Projenin web ara üzü Eclipse Indigo editöründe Maven projesi olarak geliştirilmiştir. Web ara yüzü Jsp (Java Server Page) teknolojisi kullanılarak yazılmıştır. Sitede Google firmasın sunmuş olduğu Google Maps Api' si sayesinde haritayı kullanıp, üzerinde işlem yapabiliyoruz. Haritayı kullanabilmek için Google Maps Api Key elde etmemiz gerekiyor. Bunu da "https://code.google.com/apis/console/" sitesine girerek, kayıt yaptırıp elde ediyoruz.

JSP web sayfalarında Java dilini kullanarak dinamik web sayfaları oluşturmamızı sağlayan bir Java teknolojisidir. Jsp (Java Server Page) Html dili içine yazılır ve özel etiketler kullanarak ” <% . . . . . %> ” yazılır. JSP Sayfaları Servletlere çevrilir ve Servletler şeklinde çalışırlar. Servlet ise yazılan JSP sayfasının Java koduna çevrilmiş halidir. JSP sayfaları sıradan Web Sunucularında değil de Uygulama Sunucusunda çalışır.

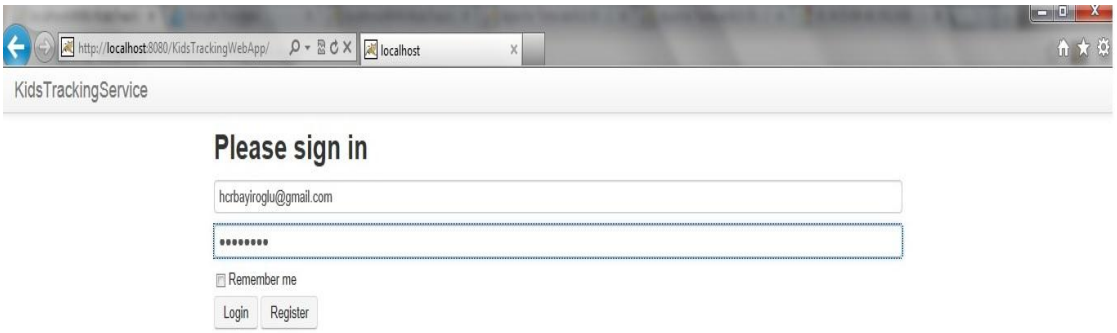
JSP içinde doğrudan Java kodu yazmak yerine, bu kodların JSP içine dahil edilmesi tavsiye edilir. Yani JSP içine veritabanı bağlantıları için Java kodu yazmak yerine, bu kodları bir Java sınıfı içine yazıp, JSP içinden bu sınıfın çağırılması tavsiye edilir. O yüzden bu sistemde veritabanı işlemleri için "com.bayiroglu.model" paketinde model sınıflar oluşturarak JSP sayfalarında kullanılacak verileri Java sınıflarıyla elde edilmiştir. "com.bayiroglu.utils" paketindeki "DBConnetion.java " sınıfı veritabanına bağlantıyı sağlamaktadır.

Web sayfasının mimarisini oluşturan sınıfların Project Explorer da ki görüntüsü aşağıdaki gibidir.



Şekil 4.6. Web Sayfasının Sınıf Mimarisi

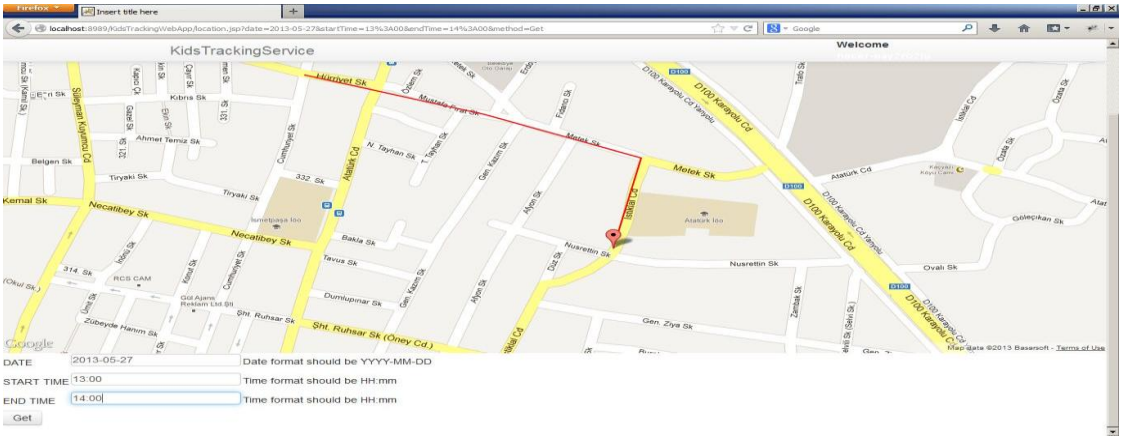
Web sayfasında ilk ekrana "login" (giriş) sayfası geliyor. Bu sayfada kullanıcı "register" butonuyla sisteme kayıt olabiliyor veya daha önceden kayıt olduğu mail adresi ve şifresiyle sisteme girişini yapıyor. Giriş yaptığında "kidsinfo" sayfasına yönlendiriliyor. Bu sayfada ebeveyne ait çocukların bilgileri görüntüleniyor. Ebeveyn sisteme ilk defa giriyorsa hiç bir veri göremeyecektir. Sol üst köşedeki "Add" butonuyla "kidsedit " sayfasına gidip yeni bir çocuk girişi yapmalıdır ve ayrıca "kidsinfo" sayfasında çocuğun üzerine tıklandığında "kidsedit" sayfasına yönlendiriliyor ve böylece kullanıcıya önceden girilen bilgilerde değişiklik yapma imkanı sağlıyor.



4.7. Şekil Web Sayfasının Giriş Ekran Görüntüsü

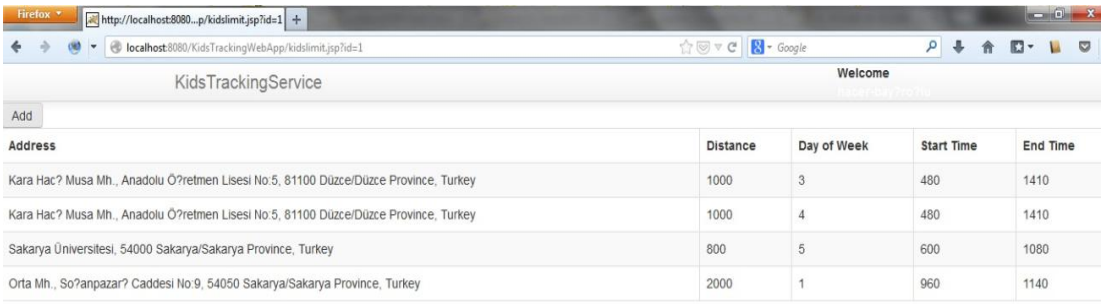
Şekil 4.8. Web Arayüzünde "kidsinfo" Sayfasının Ekran Görüntüsü

"kidsinfo" sayfasında "where is my kid" butonu tıklandığında "location" sayfasına yönlendiriliyor. Burada çocuğun mevcut konumu harita üzerinde gösteriliyor. Ayrıca bu sayfada istediğimiz gün ve saatleri girerek geçmiş konumları harita üzerinde görebiliyoruz.



Şekil 4.9. Çocuğun Gezdiği Yerleri Gösteren Ekran Görüntüsü

"kidsinfo" sayfasında "Add Limit" butonu tıklandığında "kidslimit" sayfasına yönlendiriliyor ve bu sayfada önceden belirlenmiş sınırlamaları görebiliyoruz. Ayrıca "Add" butonuyla "addlimit" sayfasına yönlendirilerek, bu sayfadaki "Find Adress" butonuyla sınırlamayı belirlemek istediğimiz adresi girip, sistemin adresin enlem ve boylamını bulmasını sağlıyoruz ve daha sonra bu adrese hangi gün hangi saatler arasında ve ne kadar mesafeyle sınırlama girmek istediğimizi belirtiyoruz.



Şekil 4.10. Sınır Belirleme Ekranı

"kidsinfo" sayfasında "violation" butonu violation sayfasına yönlendiriliyor ve bu sayfa bize harita üzerinde mevcut konum bilgisini gösteriyor ve haritanın altında ise son yirmi ihlali adresleriyle beraber sıralıyor. İhlalin üzerine tıkladığımızda da harita üzerinde gösteriyor.

4.4. Maliyet Analizi

Bu bölümde ;böyle bir uygulamanın Google Play Store' a yüklenip ebeveynlerin kullanımına açmak için satıcı firmaya mal olan yaklaşık maliyet ve ebeveynin böyle bir hizmete en az ne kadar ücretle sahip olabileceği hesaplanmıştır.

Projenin en başta yazılım maliyeti vardır. Projenin sunucu kısmı için bir bulut sunucusu tercih edilebilir. Daha sonra web sitesinin yayına çıkabilmesi için bir alan adı tescili alınmalıdır. Bu projede kullanılan Google Maps API ticari amaçla kullanılacağı için ve belirli bir harita hitini aşacağı için ücrete tabidir. Ayrıca projeye yönelik güncellemeler yapılmalı ve teknik desteği verilmelidir.Ve Google Play Store' a uygulamayı yükleyebilmek için üye olmak gerekmektedir ve bu üyelik işlemi için bir defaya mahsus ücret alınmaktadır.

Ebeveyn çocuk takip uygulamasının bir firmaya maliyeti aşağıda ki gibidir:

Maliyet	Fiyat
Yazılım	2000$
Bulut Sunucu	60*12=720$
Alan Adı Tescili	8$
Google Maps API Premium	10$
Ar-Ge Çalışmaları ve Teknik Destek	800$
Google Play Store' a Yükleme	25$

Toplam: 3563$ yıllık maliyet
3563÷12 =296.9 $ aylık maliyet

Uygulama Google Play Store' a ücretsiz yüklenip, kullanıcılardan servis hizmetinin ücreti talep edilebilir. Bu uygulamayı ortalama 100 kişi kullansa firmaya kişi başı maliyeti yaklaşık 3$'dır. Firma kişi başı 5$ karla satsa ebeveyn böyle bir uygulamaya yaklaşık 8$' a sahip olabilir.

BÖLÜM 5. SONUÇLAR VE ÖNERİLER

Günümüzün çevresel şartları ebeveynlerin çocuklarını sürekli gözetim altında tutmak istemelerine neden olmaktadır. Bu çalışma, GPS modüllerinin telefonlara entegre edilecek kadar küçüldüğü, 3G teknolojisi ile mobil cihazlarda çok yüksek hızlara çıkıldığı bir dönemde, çocukların bulundukları yerleri internet üzerinden merkezi bir sistemde birleştirerek ebeveynlerin paylaşımına ve kontrolüne açan bir sistemdir.

GPS, Dünya üzerindeki bir nesnenin yükseklik, enlem, boylam cinsinden koordinat bilgilerini sağlayan küresel konumlama sistemidir. Bu sistemin ilk kuruluş hedefi tamamen askeri amaçlar içindi ve GPS alıcıları yön bulmakta, askeri çıkartmalarda ve roket atışlarında kullanılmak üzere tasarlanmıştır. Ancak 1980' lerde GPS sistemi sivil kullanıma da açılmasıyla günümüzün birçok alanında kullanılan hayati önem taşıyan bir araç haline gelmiştir. GPS bu çalışmanın, en az hatayla konum verisi elde edilebilmesinden dolayı, temelini oluşturmaktadır.

Çalışma kapsamında mobil, web servisi ve web platformlara özel üç uygulama geliştirilmiştir. Kullanıcıların konum bilgileri mobil uygulamada elde edilip, web servisi yardımı ile sistemin veritabanına kaydedilmektedir. Web uygulaması ise ebeveynlerin, kullanıcıların konumlarını anlık olarak görebildikleri, sınırlamalar belirleyip, sınır ihlalleri olduğunda bilgilendirildikleri uygulamadır. Web uygulaması herhangi bir tarayıcıdan ulaşılabilmektedir. Ancak mobil uygulama, Android İsletim Sistemine sahip mobil cihazlar üzerinde çalışmaktadır. Android üzerinde uygulama geliştirmeye başlayabilmek için 4 yazılıma ihtiyaç vardır, Java Develpoment Kit (jdk 1.6) , Eclipse, Android SDK ve Android Development Tools Plugin. Veritabanı için MySQL Server, web servisi için RESTFUL mimarisi, web sayfası için JSP(Java Server Pages) tercih edilmiştir. Web sayfasında kullanıcının konumunun gösterildiği haritalar için Google Maps API tercih edilmiştir. Google Maps API'leri,

geliştiricilerin Google Maps uygulamasını kendi uygulamalarına entegre etmelerini sağlayacak Javascript destekli kütüphanelerdir.

Bu çalışma öncelikli olarak cihazın, en iyi konum tahminini veren konum sağlayıcısı olan GPS modülünü kullanarak konum bulur. Eğer herhangi bir neden den ötürü GPS kullanılamazsa, kablosuz ağ erişim noktaları aracılığıyla konum bilgisi elde eder ve o an için telefonun Wi-Fi özelliği de kullanılamazsa baz istasyonlarından mobil ülke kodu, mobil ağ kodu, mobil bölge kodu, baz istasyonu ID (cell-id)'si alınarak daha düşük de olsa bulunduğu mahalleye kadar konum bilinebilecek bir hassasiyetle konum tespit edilebilir. Sistemde mobil telefonda internet olduğu sürece konum verileri enlem, boylam, zaman bilgileri 10 sn' de bir veri tabanına kaydedilmektedir. İnternet olmadığında ise bu veriler telefonda bir dosyaya kaydedilip, internet geldiğinde veri tabanına aktarılmaktadır. Telefona "nerdesin" mesajı geldiğinde otomatik adres bilgisi gönderilmektedir. Web sayfası üzerinden ebeveyn mevcut konumunu, geçmişe dönük bulunduğu yerleri harita üzerinden izleyebilmektedir. Ayrıca web sayfasından istediği saat, yer sınırlamaları girip, bu sınırlar aşıldığında maille bilgilendirilmektedir.

Telefonda GPS, Wi-Fi kapalıysa yalnızca konum bilgisini GSM Baz İstasyonlarından elde edilebilmektedir. Sistemde elde edilen bu veri konum bilgisine dönüştürülemeyip herhangi bir yerde kaydedilmemektedir, yalnızca mesaj atıldığında elde edilmektedir. Ancak başka ücretsiz siteler bu verilerin girilerek konum bilgisinin harita üzerinde gösterilmesi hizmetini sunmaktadır.

Gelecek çalışmalarda GSM Baz İstasyonlarından elde edilen değerler sadece mesajla ebeveyne iletilmeyip, kayıt altına alınıp web sayfasında takibi sağlanabilir. Ayrıca sistemi milyonlarca anne ve babanın hizmetine sunulabilmek için sistemin mimarisi geliştirip güçlendirilmelidir, özellikle konum verilerinin kaydedildiği tablolar daha optimal hale getirilebilir. Ayrıca şuan sistem sadece Android İşletim Sistemine sahip telefonlara hizmet vermektedir. Aynı yazılım diğer mobil telefonlar içinde geliştirilip tek bir web sitesi üzerinden takibi sağlanabilir. Bu sistem özellikle çocukların takibi için tasarlandığından onlar için sisteme acil durum planı eklenebilir.

KAYNAKLAR

[1] CHANDRA, A., JAİN S., ABDUL QADEER, M., Implementation of Location Awareness and Sharing System Based on GPS and GPRS using J2ME, PHP and MYSQL, 2011 Computer Research and Development (ICCRD), 2011 3rd International Conference on (Volume:1), 11-13 March 2011.

[2] LİAO, Y., CHUANG, C., JENG, J., CHEN J., Systematic Design for the Global Positional Systems with Application in Intelligent Google Android phone, 2011. System Science and Engineering (ICSSE), 2011 International Conference, 8-10 June 2011.

[3] GUPTA, R., REDDY, B., GPS and GPRS Based Cost Effective Human Tracking System Using Mobile Phones, VIEWPOINT, Volume 2, January-June 2011.

[4] ISMAEL, A. G., An Emergency System for Succoring Children Using Mobile GIS, International Journal of Advanced Computer Science and Applications, Volume 3, No. 9, 2012.

[5] XU, G., GPS Theory, Algorithms and Applications, Springer-Verlag, Heidelberg, 2003.

[6] About GPS, http://www8.garmin.com/aboutGPS, Erişim Tarihi: 5.04.2013.

[7] KURUMAHMUT, F. K., İstasyonlar Arası Yükseklik Farkının GPS Konum Belirleme Duyarlığı Üzerine Etkisi, Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, 2008.

[8] YILDIZ, F., KAHVECİ, M., PENG, GPS Global Konum Belirleme Sistemi, Teori-Uygulama, 3.Baskı, 2007.

[9] GPS, http://320volt.com/gps-incelemesi-ozellikleri-kullanim-alanlari-detaylar/, Erişim Tarihi: 4.04.2013.

[10] GPS, http://en.wikipedia.org/wiki/GPS, Erişim Tarihi: 05.04.2013.

[11] GPS, http://tr.wikipedia.org/wiki/GPS, Erişim Tarihi: 5.04.2013.

[12] Mapping, http://www.trimble.com/gps_tutorial/gpswork-map.aspx, Erişim Tarihi: 03.04.2013.

[13] GPS, http://www.navcen.uscg.gov/?pageName=GPSmain, Erişim Tarihi: 03.04.2013.

[14] Space Segment, http://infohost.nmt.edu/~mreece/gps/whatisgps.html, Erişim Tarihi: 03.04.2013.

[15] GPS Performance Standards and Specifications, http://www.gps.gov/technical/ps/2008- SPS-performance-standard.pdf.

[16] ÇINAR, S., GPS ile Araç Takip Sistemi, Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, 2005.

[17] GPS Control Segment, http://www.kowoma.de/en/gps/control_segment.htm, Erişim Tarihi: 7.04.2013.

[18] HOFMANN-WELLENHOF, B., LİCHTENEGGER, H. AND COLLİNS, J., Global Positioning System Theory and Practice, Springer-Verlag Wien,New York, 2001.

[19] GPS Receiver, http://en.wikipedia.org/wiki/Gps_receiver, Erişim Tarihi: 7.04.2013.

[20] TOPATAN, S., GPS Konum Belirleme Algoritmalarının Uygulanması, Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, 2008.

[21] KAPLAN, O., GPS, GPRS ve GIS Teknolojileri Kullanılarak 112 Acil Yardım Merkezi Otomasyon Sistemi Tasarımı.

[22] DEMİR, Y., GPS Anteni Faz Merkezi Parametrelerinin Baz Ölçüleriyle Test Edilmesi, Yüksek Lisans Tezi, Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsü, 2006.

[23] CAN, Z., GÜMRÜKÇÜ O., İyonosfer ile GPS Sinyallerinin Etkileşimi, İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi Yıl:8 Sayı:15 Bahar 2009/1 s. 69- 77.

[24] ARSLAN, N., DEMİREL, H., The Impact of Temporal Ionospheric Gradients in Northern Europe on Relative GPS Positioning, Journal of Atmospheric and Solar-Terrestrial Physics, 2008.

[25] ÇETİN, M., GPS' e Atmosferin Etkileri, Harita ve Kadastro Mühendisleri Odası Dergisi, sayı: 86, syf:14-20, 1999.

[26] ŞANLIOĞLU, İ., Global Konum Belirleme Sistemi Yazılımlarının Veri İşleme Modüllerinin Uluslararası GPS Servisi Ürünleri Kullanarak Test Edilmesi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, 2004.

[27] BATAK, T., GPS İle Tespit Edilen Konuma Göre Bölgesel Reklam Yayını Uygulaması,Y üksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü, 2006.

[28] JİN, M. H., HAN, Y. H., CHOİ H. H., PARK C., HEO M., LEE S. J., GPS Spoofing Signal Detection and Compensation Method in DGPS Reference Station, 11th International Conference on Control, Automation and Systems , 2011.

[29] KAHRAMAN S., DGPS Tekniği Kullanılarak Hareket Eden Bir Aracın Hassas Konumunun Seri İletişim Yöntemi İle Tespiti ve GPS Ölçümüne Göre Yapılan Hata Oranının Karşılaştırılması, Osmangazi Üniversitesi Fen Bilimleri Enstitüsü, 2004.

[30] GPS Receivers and NMEA Standard, http://www.engineersgarage.com/tutorials/gps-receivers-nmea-standards, Erişim Tarihi: 7.04.2013.

[31] NMEA Standard, http://www.nmea.org/content/nmea_standards/nmea_0183_v_410.asp, Erişim Tarihi: 7.04.2013.

[32] NMEA Standard, http://www.kh-gps.de/nmea-faq.htm, Erişim Tarihi: 7.04.2013.

[33] Android , http://www.android.com/about/, Erişim Tarihi: 10.04.2013.

[34] Android İşletim Sistemi, http://tr.wikipedia.org/wiki/Android_(i%C5%9Fletim_sistemi), Erişim Tarihi:10.04.2013.

[35] ÖNDER, M. , MERMERKAYA, A. O., Merhaba Android, Pusula, 2011.

[36] Android Architecture, http://www.android-app-market.com/android architecture.html, Erişim Tarihi: 10.04.2013.

[37] Android ÇalışmaZamanı, http://www.gdgankara.org/2012/02/02/mobil-isletimsistemleri/?lang=tr, Erişim Tarihi:11.04.2013.

[38] NARMAN, A. E., Android Programlama,2.Baskı,2013.

[39] Android Components, http://developer.android.com/guide/components/fundamentals.html, Erişim Tarihi:10.04.2013.

[40] Activity, http://developer.android.com/reference/android/app/Activity.html, Erişim Tarihi:11.04.2013.

[41] Servisler, http://androidgelistir.blogspot.com/2010/07/android-ile-uygulamasnn-bilesenleri-ve.html, Erişim Tarihi: 14.04.2013.

[42] Broadcast Receiver, http://developer.android.com/reference/android/content/BroadcastReceiver.html, Erişim Tarihi: 14.04.2013.

[43] ÖĞÜTME, N., Android, 2011.

[44] What is MySQL?, http://dev.mysql.com/doc/refman/5.5/en/ ,MySQL 5.6 Reference Manual Including MySQL Cluster NDB 7.3 Reference Guide, Erişim Tarihi: 15.04.2013.

[45] MySQL, KREGER, H., Web Services Conceptual Architecture (WSCA) 1.0, IBM Software Group, May 2001. DuBois P., MySQL Cookbook, 2002, Erişim Tarihi: 15.04.2013.

[46] MySQL , http://www.daha.net/blog/mysql-hakkinda-bilmeniz-gereken-her-sey/, Erişim Tarihi: 15.04.2013.

[47] KREGER, H., Web Services Conceptual Architecture (WSCA) 1.0, IBM Software Group, May 2001.

[48] JDK, http://www.oracle.com/technetwork/java/javase/downloads/index.html, Erişim Tarihi: 15.05.2013.

[49] Emulator, http://developer.android.com/tools/help/emulator.html, Erişim Tarihi: 17.05.2013.

EKLER

EK -A

Mobil Uyglama

LoginActivity.java

```
package com.bayiroglu.kidstracking;
import java.util.Currency;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.TelephonyManager;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import com.bayiroglu.restclient.RestServices;

public class LoginActivity extends Activity {
@Override
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.login);
Button btnLogin = (Button) findViewById(R.id.btnLogin);
final EditText edtEmail = (EditText)
findViewById(R.id.editTxtEmail);
```

```
final EditText edtPassword = (EditText);
findViewById(R.id.editTxtPassword);
double latitude=41.026106;
double longitude=28.939018;
btnLogin.setOnClickListener(new OnClickListener() {
@Override
public void onClick(View arg0) {
final String email = edtEmail.getText().toString();
final String password = edtPassword.getText().toString();
if (RestServices.login(email,password,getApplicationContext())) {
String phoneNumber = getMyPhoneNumber();
String IMEI=getMyPhoneIMEI();
int phoneStatus=RestServices.getPhoneStatus(email, phoneNumber).getStatus();
if (phoneStatus == -1) {
Toast.makeText(LoginActivity.this, "The Phone Number is not Defined!!! ",
Toast.LENGTH_SHORT).show();
}
else if (phoneStatus == 0) {
if(RestServices.registerPhone(phoneNumber, IMEI))
{
Intent intent = new Intent(LoginActivity.this, MainActivity.class);
intent.putExtra("EMAIL", email);
intent.putExtra("PHONENUMBER", phoneNumber);
intent.putExtra("IMEI", IMEI);
startActivity(intent);
finish();
}
else
{
Toast.makeText(LoginActivity.this, "There is an Error While Activating Phone",
Toast.LENGTH_SHORT).show();
}
}
```

```
else if (phoneStatus == 1) {
Intent intent = new Intent(LoginActivity.this, MainActivity.class);
intent.putExtra("EMAIL", email);
intent.putExtra("PHONENUMBER", phoneNumber);
intent.putExtra("IMEI", IMEI);
startActivity(intent);
finish();
}
else {
}
} else {
Toast.makeText(LoginActivity.this, "Email or Password is Worng!!!",
Toast.LENGTH_SHORT).show();
}
}
});
}
private String getMyPhoneNumber() {
TelephonyManager mTelephonyMgr;
mTelephonyMgr=(TelephonyManager)getSystemService
(Context.TELEPHONY_SERVICE);
return mTelephonyMgr.getLine1Number();}
private String getMyPhoneIMEI()
{
TelephonyManager telephonyManager = (TelephonyManager)
this.getSystemService(Context.TELEPHONY_SERVICE);
String IMEI_Number = telephonyManager.getDeviceId();
return IMEI_Number;
}
}
```

MainActivity.java

```
package com.bayiroglu.kidstracking;
import android.app.Activity;
import android.app.ActivityManager;
import android.app.ActivityManager.RunningServiceInfo;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import com.bayiroglu.restclient.RestServices;
import com.bayiroglu.restmodel.KidsInfo;
public class MainActivity extends Activity {

@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
final TextView txtViewName=(TextView)findViewById(R.id.txtKidsName);
final TextView
txtViewFamilyEmail=(TextView)findViewById(R.id.txtFamilyEmail);
final TextView
txtViewPhoneNumber=(TextView)findViewById(R.id.txtPhoneNumber);
final TextView txtViewIMEI=(TextView)findViewById(R.id.txtPhoneIMEI);
final Button btnToggle =(Button)findViewById(R.id.btnToggle);
Bundle bundle= getIntent().getExtras();
String email=bundle.getString("EMAIL");
String phoneNumber=bundle.getString("PHONENUMBER");
String IMEI=bundle.getString("IMEI");
```

```
KidsInfo kidsInfo = RestServices.getKidsInfo(phoneNumber, IMEI);

txtViewFamilyEmail.setText(kidsInfo.getFamilyEmail());
txtViewIMEI.setText(kidsInfo.getIMEI());
txtViewName.setText(kidsInfo.getName());
txtViewPhoneNumber.setText(kidsInfo.getPhoneNumber());
if(isServiceRunning())
{
btnToggle.setText("Disable");
}
else
{
btnToggle.setText("Enable");
}

btnToggle.setOnClickListener(new OnClickListener()
{
@Override
public void onClick(View arg0) {
// TODO Auto-generated method stub
if(isServiceRunning())
{
stopService(new Intent(MainActivity.this, TimerServices.class));
btnToggle.setText("Enable");
}
else
{
startService(new Intent(MainActivity.this, TimerServices.class));
btnToggle.setText("Disable");
}
}
});
}
```

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.main, menu);
return true;
}
private boolean isServiceRunning()
{
ActivityManager activityManager = (ActivityManager)
getSystemService(ACTIVITY_SERVICE);
for(RunningServiceInfo runningServiceInfo :
activityManager.getRunningServices(Integer.MAX_VALUE))
{
if
(getApplication().getPackageName().equals(runningServiceInfo.service.getPackage
Name()))
{
return true;
}
}
return false;
}
}
```

GPSTracker.java

```
package com.bayiroglu.kidstracking;
import android.app.AlertDialog;
import android.app.Service;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.location.Location;
import android.location.LocationListener;
```

```
import android.location.LocationManager;
import android.os.Bundle;
import android.os.IBinder;
import android.provider.Settings;
import android.util.Log;

public class GPSTracker extends Service implements LocationListener {
private final Context mContext;
boolean isGPSEnabled = false;
boolean isNetworkEnabled = false;
boolean canGetLocation = false;
Location location;
double latitude;
double longitude;
private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES = 10;
private static final long MIN_TIME_BW_UPDATES = 1000 * 60 * 1;
protected LocationManager locationManager;
public GPSTracker(Context context) {
this.mContext = context;
getLocation();
}
public Location getLocation() {
try {
locationManager = (LocationManager) mContext
.getSystemService(LOCATION_SERVICE);
isGPSEnabled = locationManager
.isProviderEnabled(LocationManager.GPS_PROVIDER);
isNetworkEnabled = locationManager
.isProviderEnabled(LocationManager.NETWORK_PROVIDER);
if (!isGPSEnabled && !isNetworkEnabled) {
this.canGetLocation = false;
} else {
this.canGetLocation = true;
```

```
if (isNetworkEnabled) {
locationManager.requestLocationUpdates(
LocationManager.NETWORK_PROVIDER,
MIN_TIME_BW_UPDATES,
MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
Log.d("Network", "Network");
if (locationManager != null) {
location = locationManager
.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
if (location != null) {
latitude = location.getLatitude();
longitude = location.getLongitude();
}
}
}
if (isGPSEnabled) {
if (location == null) {
locationManager.requestLocationUpdates(
LocationManager.GPS_PROVIDER,
MIN_TIME_BW_UPDATES,
MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
Log.d("GPS Enabled", "GPS Enabled");
if (locationManager != null) {
location = locationManager
.getLastKnownLocation(LocationManager.GPS_PROVIDER);
if (location != null) {
latitude = location.getLatitude();
longitude = location.getLongitude();
}
}
}
}
}
```

```
} catch (Exception e) {
e.printStackTrace();
}
return location;
}
public void stopUsingGPS() {
if (locationManager != null) {
locationManager.removeUpdates(GPSTracker.this);
}
}
public double getLatitude() {
if (location != null) {
latitude = location.getLatitude();
}
return latitude;
}
public double getLongitude() {
if (location != null) {
longitude = location.getLongitude();
}
return longitude;
}
public boolean canGetLocation() {
return this.canGetLocation;
}
public void showSettingsAlert() {
AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);
alertDialog.setTitle("GPS is settings");
alertDialog
.setMessage("GPS is not enabled. Do you want to go to settings menu?");
alertDialog.setPositiveButton("Settings",
new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int which) {
```

```
Intent intent = new Intent(
Settings.ACTION_LOCATION_SOURCE_SETTINGS);
mContext.startActivity(intent);
}
});
alertDialog.setNegativeButton("Cancel",
new DialogInterface.OnClickListener() {
public void onClick(DialogInterface dialog, int which) {
dialog.cancel();
}
});
alertDialog.show();
}
@Override
public void onLocationChanged(Location location) {
}
@Override
public void onProviderDisabled(String provider) {
}
@Override
public void onProviderEnabled(String provider) {
}
@Override
public void onStatusChanged(String provider, int status, Bundle extras) {
}
@Override
public IBinder onBind(Intent arg0) {
return null;
}
}
```

SMSListener.java

```
package com.bayiroglu.kidstracking;
```

```
import com.bayiroglu.restclient.RestServices;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.telephony.SmsManager;
import android.telephony.SmsMessage;
import android.telephony.TelephonyManager;
import android.telephony.gsm.GsmCellLocation;
import android.util.Log;

public class SMSListener extends BroadcastReceiver {
public static final String SMS_RECEIVED =
"android.provider.Telephony.SMS_RECEIVED";
private static GPSTracker gps;
public SMSListener(GPSTracker gpsTracker) {
super();
gps=gpsTracker;
}
@Override
public void onReceive(Context context, Intent intent) {
if (intent.getAction().equals(SMS_RECEIVED)) {
Bundle bundle = intent.getExtras();
SmsMessage[] msgs = null;
String msg_from;
if (bundle != null) {
try {
Object[] pdus = (Object[]) bundle.get("pdus");
msgs = new SmsMessage[pdus.length];
for (int i = 0; i < msgs.length; i++) {
msgs[i] = SmsMessage.createFromPdu((byte[]) pdus[i]);
msg_from = msgs[i].getOriginatingAddress();
final String msgBody = msgs[i].getMessageBody();
```

```
if (msgBody.toLowerCase().contains("nerdesin")) {
String smsText;
try {
if(gps.isGPSEnabled && gps.isNetworkEnabled)
{
smsText=RestServices.getAddress(gps.getLatitude(), gps.getLongitude());
}
else if(gps.isNetworkEnabled)
{
smsText=RestServices.getAddress(gps.getLatitude(), gps.getLongitude());
}
else if(gps.isGPSEnabled)
{
smsText="Latitude :"+gps.getLatitude()+", Longitude"+ gps.getLongitude();
}else
{
smsText=getSMSText(context);
}
sendSMS(msg_from, smsText);
} catch (Exception e) {
sendSMS(msg_from, getSMSText(context));
}
}
}
} catch (Exception e) {
Log.d("Exception caught", e.getMessage());
}
}
}
}
private void sendSMS(String phoneNumber, String text) {
SmsManager smsManager = SmsManager.getDefault();
smsManager.sendTextMessage(phoneNumber, null, text, null, null);
```

```
}
private String getSMSText(Context context) {
TelephonyManager telephonyManager = (TelephonyManager) context
.getSystemService(Context.TELEPHONY_SERVICE);
GsmCellLocation cellLocation = (GsmCellLocation) telephonyManager
.getCellLocation();
String networkOperator = telephonyManager.getNetworkOperator();
String mcc = networkOperator.substring(0, 3);
String mnc = networkOperator.substring(3);
int cid = cellLocation.getCid();
int lac = cellLocation.getLac();
String text="Base Station Information mcc:"+mcc +", mnc:"+mnc+", cid:"+cid+",
lac:"+lac;
return text;
}
}
```

TimerServices.java

```
package com.bayiroglu.kidstracking;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.Date;
import java.util.Timer;
import java.util.TimerTask;
import com.bayiroglu.restclient.RestServices;
import android.app.Service;
import android.content.BroadcastReceiver;
import android.content.Context;
```

```
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Bundle;
import android.os.Handler;
import android.os.IBinder;
import android.os.Looper;
import android.telephony.SmsManager;
import android.telephony.SmsMessage;
import android.telephony.TelephonyManager;
import android.util.Log;
import android.widget.Toast;
public class TimerServices extends Service {
private static final String FILENAME="location";
private Timer timer;
private Handler handler;
private Double latitude = 40.743243;
private Double longitude = 30.331433;
private String address = "";
GPSTracker gps;
private SMSListener smsReciver;
public static final String SMS_RECEIVED =
"android.provider.Telephony.SMS_RECEIVED";
final static long TIME = 10000;
@Override
public IBinder onBind(Intent intent) {
return null;
}
@Override
public void onCreate() {
super.onCreate();
timer = new Timer();
handler = new Handler(Looper.getMainLooper());
gps = new GPSTracker(this);
```

```
final IntentFilter theFilter = new IntentFilter();
theFilter.addAction(SMS_RECEIVED);
smsReciver= new SMSListener(gps);
registerReceiver(smsReciver, theFilter);
timer.scheduleAtFixedRate(new TimerTask() {
@Override
public void run() {
bilgiVer();
}
}, 0, TIME);
}
private void bilgiVer() {
handler.post(new Runnable() {
@Override
public void run() {
try {
gps.getLocation();
if (gps.canGetLocation()) {
latitude = gps.getLatitude();
longitude = gps.getLongitude();
address = RestServices.getAddress(latitude, longitude);
if (RestServices.postLocation(null,getMyPhoneNumber(),longitude, latitude)) {
submitSavedLocation();
} else {
logLocation();
}
Toast.makeText(getApplicationContext(),"Your Location is - \nLat: " + latitude+
"\nLong: " + longitude,
Toast.LENGTH_LONG).show();
} else {
gps.showSettingsAlert();
}
} catch (Exception e) {
```

```
}
}
});
}
@Override
public void onDestroy() {
unregisterReceiver(smsReciver);
timer.cancel();
super.onDestroy();
}
private String getMyPhoneNumber() {
TelephonyManager mTelephonyMgr;
mTelephonyMgr = (TelephonyManager)
getSystemService(Context.TELEPHONY_SERVICE);
return mTelephonyMgr.getLine1Number();
}
private void logLocation()
{
try {
FileOutputStream fos = openFileOutput(FILENAME, this.MODE_APPEND);
Date dt = new Date();
String data=dt.getTime()+";"+longitude.toString()+";"+latitude.toString()+";"+"\n";
fos.write(data.getBytes());
fos.close();
} catch (IOException e) {
Log.e("Controller", e.getMessage() + e.getLocalizedMessage() + e.getCause());
Toast.makeText(TimerServices.this,"Error Occured While Location submitting or
saving location",
Toast.LENGTH_SHORT).show();
}
}
private void submitSavedLocation()
{
```

```
String line=null;
try {
InputStream in = openFileInput(FILENAME);
File file = new File(FILENAME);
if (in != null & file.exists()) {
InputStreamReader input = new InputStreamReader(in);
BufferedReader buffreader = new BufferedReader(input);
while (( line = buffreader.readLine()) != null) {
String[] locationComponent=line.split(";");
if(locationComponent.length==3)
{
RestServices.postLocation(locationComponent[0].trim(),getMyPhoneNumber(),Dou
ble.parseDouble(locationComponent[1].trim()),
Double.parseDouble(locationComponent[2].trim()));
}
}
in.close();
}
}catch (Exception e) {
}
FileOutputStream fos;
try {
fos = openFileOutput(FILENAME, this.MODE_PRIVATE);
String data=" ";
fos.write(data.getBytes());
fos.close();
} catch (FileNotFoundException e) {
e.printStackTrace();
} catch (IOException e) {
e.printStackTrace();
}
Date dt = new Date();}
}
```

RestServices.java

```
package com.bayiroglu.restclient;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.StatusLine;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import com.bayiroglu.kidstracking.LoginActivity;
import com.bayiroglu.restmodel.KidsInfo;
import com.bayiroglu.restmodel.PhoneStatus;
import android.content.Context;
import android.telephony.TelephonyManager;
import android.util.Log;
import android.widget.Toast;

public class RestServices {
private static String username = "sometyx";
private static String passwordAuthentication = "kfcx5.mnopq";
private static DefaultHttpClient httpclient = new DefaultHttpClient();
public static Boolean postLocation(String date,String phoneNumber, Double
longitude,
Double Latitude) {
String link = "http://192.168.43.156:8080
/KidsTrackingServices/rest/LocationSubmit/"+ date+ "/"+ phoneNumber+ "/"+
longitude.toString()+ "/"+ Latitude.toString() + "";
```

```
HttpGet httpget = new HttpGet(link);
httpget.addHeader("usrnm", username);
httpget.addHeader("pswdx", passwordAuthentication);
HttpResponse response;
try {
response = httpclient.execute(httpget);
StatusLine requestSuccess = response.getStatusLine();
if (requestSuccess.getStatusCode() == 200) {
HttpEntity entity = response.getEntity();
if (entity != null) {
return true;
}
}
} catch (Exception ex) {
Log.e("REST_ERR", ex.getMessage());
}
return false;
}
public static Boolean login(String email, String password,Context context) {
String link = "http://192.168.43.156:8080 /KidsTrackingServices/rest/Login/"
+ email + "/" + password;
Toast.makeText(context,link,
Toast.LENGTH_SHORT).show();
HttpGet httpget = new HttpGet(link);
httpget.addHeader("usrnm", username);
httpget.addHeader("pswdx", passwordAuthentication);
HttpResponse response;
try {
response = httpclient.execute(httpget);
StatusLine requestSuccess = response.getStatusLine();
if (requestSuccess.getStatusCode() == 200) {
HttpEntity entity = response.getEntity();
if (entity != null) {
```

```
InputStream instream = entity.getContent();
String requestResult = convertStreamToString(instream);
try {
JSONObject jsonResult = new JSONObject(requestResult);
return jsonResult.getBoolean("result");
} catch (JSONException e) {
Log.e("REST_ERR", e.getMessage());
}
instream.close();
}
}
} catch (Exception ex) {
Log.e("REST_ERR", ex.getMessage());
Toast.makeText(context, "There is an Error While Login",
Toast.LENGTH_SHORT).show();
}
return false;
}
public static PhoneStatus getPhoneStatus(String email, String phoneNumber) {
String link = "http://192.168.43.156:8080 /KidsTrackingServices/rest/PhoneStatus/"
+ phoneNumber + "/" + email;
HttpGet httpget = new HttpGet(link);
httpget.addHeader("usrnm", username);
httpget.addHeader("pswdx", passwordAuthentication);
HttpResponse response;
PhoneStatus ps = new PhoneStatus();
ps.setStatus(-1);
try {
response = httpclient.execute(httpget);
StatusLine requestSuccess = response.getStatusLine();
if (requestSuccess.getStatusCode() == 200) {
HttpEntity entity = response.getEntity();
if (entity != null) {
```

```
InputStream instream = entity.getContent();
String requestResult = convertStreamToString(instream);
try {
JSONObject jsonResult = new JSONObject(requestResult);
ps.setStatus(jsonResult.getInt("status"));
} catch (JSONException e) {
Log.e("REST_ERR", e.getMessage());
}
instream.close();
}
}
} catch (Exception ex) {
Log.e("REST_ERR", ex.getMessage());
}
return ps;
}
public static Boolean registerPhone(String phoneNumber, String IMEI) {
String link = "http://192.168.43.156:8080
/KidsTrackingServices/rest/RegisterPhone/"
+ phoneNumber + "/" + IMEI;
HttpGet httpget = new HttpGet(link);
httpget.addHeader("usrnm", username);
httpget.addHeader("pswdx", passwordAuthentication);
HttpResponse response;
PhoneStatus ps = new PhoneStatus();
ps.setStatus(-1);
try {
response = httpclient.execute(httpget);
StatusLine requestSuccess = response.getStatusLine();
if (requestSuccess.getStatusCode() == 200) {
HttpEntity entity = response.getEntity();
if (entity != null) {
InputStream instream = entity.getContent();
```

```
String requestResult = convertStreamToString(instream);
try {
JSONObject jsonResult = new JSONObject(requestResult);
return jsonResult.getBoolean("result");
} catch (JSONException e) {
// TODO: handle exception
Log.e("REST_ERR", e.getMessage());}
instream.close();
}
}
} catch (Exception ex) {
Log.e("REST_ERR", ex.getMessage());
}
return false;
}
public static KidsInfo getKidsInfo(String phoneNumber, String IMEI) {
String link = "http://192.168.43.156:8080 /KidsTrackingServices/rest/KidsInfo/"
+ phoneNumber + "/" + IMEI;
HttpGet httpget = new HttpGet(link);
httpget.addHeader("usrnm", username);
httpget.addHeader("pswdx", passwordAuthentication);
HttpResponse response;
PhoneStatus ps = new PhoneStatus();
ps.setStatus(-1);
try {
response = httpclient.execute(httpget);
StatusLine requestSuccess = response.getStatusLine();
if (requestSuccess.getStatusCode() == 200) {
HttpEntity entity = response.getEntity();
if (entity != null) {
InputStream instream = entity.getContent();
String requestResult = convertStreamToString(instream);
try {
```

```
JSONObject jsonResult = new JSONObject(requestResult);
KidsInfo kidsInfo = new KidsInfo();
kidsInfo.setFamilyEmail(jsonResult
.getString("familyEmail"));
kidsInfo.setIMEI(jsonResult.getString("imei"));
kidsInfo.setName(jsonResult.getString("name"));
kidsInfo.setPhoneNumber(jsonResult
.getString("phoneNumber"));
return kidsInfo;
} catch (JSONException e) {
Log.e("REST_ERR", e.getMessage());
}
instream.close();
}
}
} catch (Exception ex) {
Log.e("REST_ERR", ex.getMessage());
}
return null;
}
public static String getAddress(Double latitude, Double longitude) {
String link = "http://maps.googleapis.com/maps/api/geocode/json?latlng=" +
latitude.toString() + "," + longitude.toString() + "&sensor=false";
HttpGet httpget = new HttpGet(link);
HttpResponse response;
try {
response = httpclient.execute(httpget);
StatusLine requestSuccess = response.getStatusLine();
if (requestSuccess.getStatusCode() == 200) {
HttpEntity entity = response.getEntity();
if (entity != null) {
InputStream instream = entity.getContent();
String requestResult = convertStreamToString(instream);
```

```
try {
JSONObject json = new JSONObject(requestResult);
JSONArray jsonArray = json.getJSONArray("results");
JSONObject addressInfo = (JSONObject) jsonArray.get(0);
String address = addressInfo.getString("formatted_address");
return address;
} catch (JSONException e) {
Log.e("REST_ERR", e.getMessage());
}
instream.close();
}
}
} catch (Exception ex) {
Log.e("REST_ERR", ex.getMessage());
}
return null;
}
private static String convertStreamToString(InputStream is) {
BufferedReader reader = new BufferedReader(new InputStreamReader(is));
StringBuilder sb = new StringBuilder();
String line = null;
try {
while ((line = reader.readLine()) != null) {
sb.append(line + "\n");
}
} catch (IOException e) {
e.printStackTrace();
} finally {
try {
is.close();
} catch (IOException e) {
e.printStackTrace();
}
```

```
}
return sb.toString();
}
}
```

KidsInfo.java

```
package com.bayiroglu.restmodel;
public class KidsInfo {
private String name;
private String phoneNumber;
private String IMEI;
private String familyEmail;
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public String getPhoneNumber() {
return phoneNumber;
}
public void setPhoneNumber(String phoneNumber) {
this.phoneNumber = phoneNumber;
}
public String getIMEI() {
return IMEI;
}
public void setIMEI(String iMEI) {
IMEI = iMEI;
}
public String getFamilyEmail() {
return familyEmail;
```

```
}
public void setFamilyEmail(String familyEmail) {
this.familyEmail = familyEmail;
}
}
```

PhoneStatus.java

```
package com.bayiroglu.restmodel;
public class PhoneStatus {
private int status;
public int getStatus() {
return status;
}
public void setStatus(int status) {
this.status = status;
}
}
```

Result.java

```
package com.bayiroglu.restmodel;
public class Result {
private Boolean result;
public Boolean getResult() {
return result;
}
public void setResult(Boolean result) {
this.result = result;
}
}
```

EK- B

Web Servis Uygulaması

KidsInfoServices.java

```
package com.bayiroglu.service.rest;
import java.io.IOException;
import javax.servlet.http.HttpServletRequest;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Context;
import javax.ws.rs.core.MediaType;
import com.bayiroglu.service.model.KidsInfo;
import com.bayiroglu.utils.AuthenticateUser;
import com.bayiroglu.utils.DBOperation;
@Path("/KidsInfo")
public class KidsInfoServices {
@Context
private HttpServletRequest servletRequest;
@GET
@Path("{phoneNumber}/{imei}")
@Produces(MediaType.APPLICATION_JSON)
public KidsInfo getKidsInfo(@PathParam("phoneNumber") String phoneNumber,
@PathParam("imei") String imei) throws IOException {
String usrname = servletRequest.getHeader("usrnm");
String psw = servletRequest.getHeader("pswdx");
if (AuthenticateUser.authenticate(usrname, psw)) {
KidsInfo kidsInfo = new KidsInfo();
kidsInfo = DBOperation.getKidsInfo(phoneNumber, imei);
return kidsInfo;
}
```

```
return null;
}
}
```

LocationSubmitServices.java

```
package com.bayiroglu.service.rest;
import java.io.IOException;
import javax.servlet.http.HttpServletRequest;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Context;
import javax.ws.rs.core.MediaType;
import com.bayiroglu.service.model.Result;
import com.bayiroglu.utils.AuthenticateUser;
import com.bayiroglu.utils.DBOperation;
import com.bayiroglu.utils.KidsLimit;
import com.bayiroglu.utils.LocationOperation;
import com.bayiroglu.utils.Mail;
import com.bayiroglu.utils.Mailer;
@Path("/LocationSubmit")
public class LocationSubmitServices {
@Context
private HttpServletRequest servletRequest;
@GET
@Path("{date}/{phoneNumber}/{longitude}/{latitude}")
@Produces(MediaType.APPLICATION_JSON)
public Result setKidsLocation(@PathParam("date") String
date,@PathParam("phoneNumber") String phoneNumber, @PathParam("longitude")
String longitude, @PathParam("latitude") String latitude) throws IOException {

String usrname = servletRequest.getHeader("usrnm");
```

```
String psw = servletRequest.getHeader("pswdx");
Mailer mailer = new Mailer();
Result lr= new Result();
lr.setResult(false);
if (AuthenticateUser.authenticate(usrname, psw)) {
if(DBOperation.insertLocation(date,phoneNumber, longitude, latitude))
{
KidsLimit kidsLimit= DBOperation.getKidsLimitForCurrentTime(phoneNumber);
if(kidsLimit!=null)
{
if(LocationOperation.isViolate(longitude, latitude, kidsLimit.getLatitude(),
kidsLimit.getLatitude(), kidsLimit.getDistance()))
{
if(!DBOperation.isViolationHandle(kidsLimit.getKidsId(),kidsLimit.getKidsLimitId(
)))
{
String address = LocationOperation.getAddresInfo(longitude, latitude);
String familyEmail = DBOperation.getFamilyEmail(phoneNumber);
if(familyEmail!=null)
{
String content="Your kids out of your limit, your kids current addres: "+address;
byte[] bs=content.getBytes();
Mail mail = new Mail("kidstrackinginfo@gmail.com", familyEmail, "Kids Tracking
Info", new String(bs,"UTF-8"));
mailer.sendMail(mail);
}
}
}
}
lr.setResult(true);
return lr;
}
}
```

```
return null;
}
}
```

LoginServices.java

```
package com.bayiroglu.service.rest;
import java.io.IOException;
import javax.servlet.http.HttpServletRequest;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Context;
import javax.ws.rs.core.MediaType;
import com.bayiroglu.service.model.Result;
import com.bayiroglu.utils.AuthenticateUser;
import com.bayiroglu.utils.DBOperation;
@Path("/Login")
public class LoginServices {
@Context
private HttpServletRequest servletRequest;
@GET
@Path("{email}/{password}")
@Produces(MediaType.APPLICATION_JSON)
public Result getLoginResult(@PathParam("email") String email,
@PathParam("password") String password) throws IOException {
System.out.println("login basladı");
System.out.println(email);
System.out.println(password);
String usrname = servletRequest.getHeader("usrnm");
String psw = servletRequest.getHeader("pswdx");
Result lr= new Result();
lr.setResult(false);
```

```
if (AuthenticateUser.authenticate(usrname, psw)) {
if(DBOperation.login(email, password))
{
lr.setResult(true);
return lr;
}

}
return null;
}
}
```

PhoneStatusServices.java

```
package com.bayiroglu.service.rest;
import java.io.IOException;
import javax.servlet.http.HttpServletRequest;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Context;
import javax.ws.rs.core.MediaType;
import com.bayiroglu.utils.AuthenticateUser;
import com.bayiroglu.utils.DBOperation;
import com.bayiroglu.service.model.PhoneStatus;
@Path("/PhoneStatus")
public class PhoneStatusServices {
@Context
private HttpServletRequest servletRequest;
@GET
@Path("{phoneNumber}/{email}")
@Produces(MediaType.APPLICATION_JSON)
```

```
public PhoneStatus getPhoneStatus(@PathParam("phoneNumber") String
phoneNumber, @PathParam("email") String email) throws IOException {
String usrname = servletRequest.getHeader("usrnm");
String psw = servletRequest.getHeader("pswdx");
if (AuthenticateUser.authenticate(usrname, psw)) {
return DBOperation.getPhoneStatus(email, phoneNumber);
}
return null;
}
}
```

RegisterPhoneServices.java

```
package com.bayiroglu.service.rest;
import java.io.IOException;
import javax.servlet.http.HttpServletRequest;
import javax.ws.rs.GET;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.Context;
import javax.ws.rs.core.MediaType;
import com.bayiroglu.service.model.Result;
import com.bayiroglu.utils.AuthenticateUser;
import com.bayiroglu.utils.DBOperation;
@Path("/RegisterPhone")
public class RegisterPhoneServices {
@Context
private HttpServletRequest servletRequest;
@GET
@Path("{phoneNumber}/{imei}")
@Produces(MediaType.APPLICATION_JSON)
public Result getPhoneStatus(@PathParam("phoneNumber") String phoneNumber,
@PathParam("imei") String imei) throws IOException {
```

```
String usrname = servletRequest.getHeader("usrnm");
String psw = servletRequest.getHeader("pswdx");
if (AuthenticateUser.authenticate(usrname, psw)) {
Result rs = new Result();
rs.setResult(DBOperation.registerPhone(phoneNumber, imei));
return rs;
}
return null;
}
}
```

AuthenticateUser.java

```
package com.bayiroglu.utils;
public class AuthenticateUser {
static String u = "sometyx";
static String p = "kfcx5.mnopq";
public static boolean authenticate(String username, String password) {
if (username.toLowerCase().equals(u) && password.equals(p)) {
return true;
}
return false;
}
}
```

DBOperation.java

```
package com.bayiroglu.utils;
import java.sql.Connection;
import java.sql.Date;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
```

```
import java.util.Calendar;
import com.bayiroglu.service.model.KidsInfo;
import com.bayiroglu.service.model.PhoneStatus;
import com.mysql.jdbc.ResultSetMetaData;
public class DBOperation {
private static String url = "jdbc:mysql://localhost:3306/";
private static String dbName = "KidsTrackingDB";
private static String driver = "com.mysql.jdbc.Driver";
private static String userName = "KidsTrackingDBA";
private static String password = "12qw34er";
public static boolean insertLocation(String date, String phoneNumber, String
longitude, String latitude) {
try {
java.util.Date dt = null;
if (date == null) {
dt = new java.util.Date();
}
else
{
long timestamp = Long.valueOf(date);
dt = new java.util.Date(timestamp);
}
String query = "insert into kids_location
(phone_number,log_date,longitude,latitude)" + "values(?,?,?,?)";
Class.forName(driver);
Connection conn = DriverManager.getConnection(url + dbName, userName,
password);
PreparedStatement pst = conn.prepareStatement(query);
pst.setString(1, phoneNumber);
pst.setDate(2, new java.sql.Date( dt.getTime() ));
pst.setString(4, longitude);
pst.setString(3, latitude);
int val = pst.executeUpdate();
```

```
if (val == 1)
return true;
conn.close();
} catch (Exception e) {
e.printStackTrace();
}
return false;
}
public static boolean login(String email, String password) {
try {
String query = "select * from family_info where password= ? and email= ?";
Class.forName(driver);
Connection conn = DriverManager.getConnection(url + dbName, userName,
password);
PreparedStatement ps = conn.prepareStatement(query);
ps.setString(1, password);
ps.setString(2, email);
ResultSet rs = ps.executeQuery();
while (rs.next())
return true;
conn.close();
} catch (Exception e) {
e.printStackTrace();
}
return false;
}
public static PhoneStatus getPhoneStatus(String email, String phoneNumber) {
PhoneStatus ps = new PhoneStatus();
ps.setStatus(-1);
try {
String query = "select * from kids_info A, family_info B where
A.family_id=B.family_id and A.phone_number='" + phoneNumber + "' and
B.email='" + email + "'";
```

```
Class.forName(driver);
Connection conn = DriverManager.getConnection(url + dbName, userName,
password);
java.sql.Statement st = conn.createStatement();
ResultSet rs = st.executeQuery(query);
while (rs.next()) {
int isActive = rs.getInt("is_active");
ps.setStatus(isActive);
}
conn.close();
} catch (Exception e) {
}
return ps;
}
public static boolean registerPhone(String phoneNumber, String imei) {
try {
String query = "update kids_info A set A.is_active=  ?,A.phone_imei= ? where
A.phone_number= ?";
Class.forName(driver);
Connection conn = DriverManager.getConnection(url + dbName, userName,
password);
PreparedStatement ps = conn.prepareStatement(query);
ps.setInt(1, 1);
ps.setString(2, imei);
ps.setString(3, phoneNumber);
if (ps.executeUpdate() == 1)
return true;
} catch (Exception e) {
}
return false;
}
public static KidsInfo getKidsInfo(String phoneNumber, String imei) {
try {
```

```
String query = "select * from kids_info A, family_info B where
A.family_id=B.family_id and A.phone_number= ? and A.phone_imei = ?";
Class.forName(driver);
Connection conn = DriverManager.getConnection(url + dbName, userName,
password);
PreparedStatement ps = conn.prepareStatement(query);
ps.setString(1, phoneNumber);
ps.setString(2, imei);
ResultSet rs = ps.executeQuery();
KidsInfo kidsInfo = new KidsInfo();
kidsInfo.setIMEI(imei);
kidsInfo.setPhoneNumber(phoneNumber);
while (rs.next()) {
kidsInfo.setName(rs.getString("kids_name"));
kidsInfo.setFamilyEmail(rs.getString("email"));
return kidsInfo;
}
} catch (Exception e) {
}
return null;
}
public static KidsLimit getKidsLimitForCurrentTime(String kidsPhone) {
KidsLimit kidsLimit = null;
Calendar c = Calendar.getInstance();
int dayOfWeek = c.get(Calendar.DAY_OF_WEEK);
int minute = c.get(Calendar.MINUTE);
int hour = c.get(Calendar.HOUR);
int minuteOfDay = hour * 60 + minute;
try {
String query = "select B.* from kids_info A, kids_limits B where A.phone_number=
? and A.kids_id=B.Kids_id and day_of_week=? and start_time < ? and end_time >
?";
Class.forName(driver);
```

```
Connection conn = DriverManager.getConnection(url + dbName, userName,
password);
PreparedStatement ps = conn.prepareStatement(query);
ps.setString(1, kidsPhone);
ps.setInt(2, dayOfWeek);
ps.setInt(3, minuteOfDay);
ps.setInt(4, minuteOfDay);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
kidsLimit = new KidsLimit();
kidsLimit.setKidsLimitId(rs.getInt("id"));
kidsLimit.setDayOfWeek(rs.getInt("day_of_week"));
kidsLimit.setStartTime(rs.getInt("start_time"));
kidsLimit.setEndTime(rs.getInt("end_time"));
kidsLimit.setDistance(rs.getInt("distance"));
kidsLimit.setLatitude(rs.getString("latitude"));
kidsLimit.setLongitude(rs.getString("longitude"));
kidsLimit.setKidsId(rs.getInt("kids_id"));
return kidsLimit;
}
} catch (Exception e) {
}
return kidsLimit;
}
public static String getFamilyEmail(String kidsPhoneNumber) {
try {
String query = "select * from kids_info A, family_info B where
A.family_id=B.family_id and A.phone_number= ? ";
Class.forName(driver);
Connection conn = DriverManager.getConnection(url + dbName, userName,
password);
PreparedStatement ps = conn.prepareStatement(query);
ps.setString(1, kidsPhoneNumber);
```

```
ResultSet rs = ps.executeQuery();
while (rs.next()) {
return rs.getString("email");
}
} catch (Exception e) {
}
return null;
}
public static boolean isViolationHandle(int kidsId, int kidsLimitId) {
try {
String query = "select * from violations_info A where A.kids_id= ? and
A.kids_limits_id=? and
date_format(sysdate(),'%d%m%Y%H')=date_format(date,'%d%m%Y%H') and
is_handle=1";
System.out.println(query);
Class.forName(driver);
Connection conn = DriverManager.getConnection(url + dbName, userName,
password);
PreparedStatement ps = conn.prepareStatement(query);
ps.setInt(1, kidsId);
ps.setInt(2, kidsLimitId);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
return true;
}
query = "insert into violations_info (kids_id,kids_limits_id,date,is_handle)
values(?,?,sysdate(),1)";
PreparedStatement ps1 = conn.prepareStatement(query);
ps1.setInt(1, kidsId);
ps1.setInt(2, kidsLimitId);
int i = ps1.executeUpdate();
} catch (Exception e) {
}
```

```
return false;
}
}
```

KidsLimit.java

```
package com.bayiroglu.utils;
public class KidsLimit {
private int kidsLimitId;
private int kidsId;
private String longitude;
private String latitude;
private int distance;
private int dayOfWeek;
private int startTime;
private int endTime;
private String Addres;
public int getKidsId() {
return kidsId;
}
public void setKidsId(int kidsId) {
this.kidsId = kidsId;
}
public String getLongitude() {
return longitude;
}
public void setLongitude(String longitude) {
this.longitude = longitude;
}
public String getLatitude() {
return latitude;
}
public void setLatitude(String latitude) {
```

```
this.latitude = latitude;
}
public int getDistance() {
return distance;
}
public void setDistance(int distance) {
this.distance = distance;
}
public int getDayOfWeek() {
return dayOfWeek;
}
public void setDayOfWeek(int dayOfWeek) {
this.dayOfWeek = dayOfWeek;
}
public int getStartTime() {
return startTime;
}
public void setStartTime(int startTime) {
this.startTime = startTime;
}
public int getEndTime() {
return endTime;
}
public void setEndTime(int endTime) {
this.endTime = endTime;
}
public String getAddres() {
return Addres;
}
public void setAddres(String addres) {
Addres = addres;
}
public int getKidsLimitId() {
```

```
return kidsLimitId;
}
public void setKidsLimitId(int kidsLimitId) {
this.kidsLimitId = kidsLimitId;
}
}
```

LocationOperation.java

```
package com.bayiroglu.utils;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import org.codehaus.jettison.json.JSONArray;
import org.codehaus.jettison.json.JSONException;
import org.codehaus.jettison.json.JSONObject;
public class LocationOperation {
public static String getAddresInfo(String longitude, String latitude) throws
IOException {
String link = "http://maps.googleapis.com/maps/api/geocode/json?latlng=" +
latitude.toString() + "," + longitude.toString() + "&sensor=false";
URL url;
try {
url = new URL(link);
BufferedReader in = new BufferedReader(new
InputStreamReader(url.openStream()));
String inputLine;
String jsonResult = "";
while ((inputLine = in.readLine()) != null) {
jsonResult += inputLine + " ";
}
```

```
JSONObject json = new JSONObject(jsonResult);
JSONArray jsonArray = json.getJSONArray("results");
JSONObject addressInfo = (JSONObject) jsonArray.get(0);
String address = addressInfo.getString("formatted_address");
return address;
} catch (MalformedURLException e) {
e.printStackTrace();
} catch (JSONException e) {
e.printStackTrace();
}
return null;
}
public static Boolean isViolate(String x1, String y1, String x2, String y2, int
distance) {
Double result = Math.sqrt(Math.pow((Double.parseDouble(x1)  -
Double.parseDouble(x2)), 2) + Math.pow((Double.parseDouble(y1) -
Double.parseDouble(y2)), 2));
if (result * 100000 > distance) {
return true;
}
return false;
}
}
```

Mail.java

```
package com.bayiroglu.utils;
public class Mail {
String from;
String to;
String title;
String body;
public Mail(String from, String to, String title, String content) {
```

```
super();
this.from = from;
this.to = to;
this.title = title;
this.body = content;
}
public void setFrom(String from) {
this.from = from;
}
public String getFrom() {
return from;
}
public void setTo(String to) {
this.to = to;
}
public String getTo() {
return to;
}
public void setTitle(String title) {
this.title = title;
}
public String getTitle() {
return title;
}
public void setBody(String content) {
this.body = content;
}
public String getBody() {
return body;
}
}
```

Mailer.java

```
package com.bayiroglu.utils;
import java.util.Properties;
import javax.mail.Message;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;
public class Mailer {
String mailSunucu = "smtp.gmail.com";
String mailUser = "kidstrackinginfo@gmail.com";
String mailPassword = "******";
String varsayilanMailSunucu = "smtp.gmail.com";
int mailAuth = 1;
String mesaj = "";
int hataKod = 0;
int duzOrHtml = 1; // 1 duz 2 html
public boolean sendMail(Mail mail) {
try {
Properties props = System.getProperties();
if (mailAuth == 1) {
String javaHome = System.getProperty("java.home");
String keyStore = javaHome + "/lib/security/cacerts";
System.setProperty("javax.net.ssl.trustStore", keyStore);
System.setProperty("javax.net.ssl.trustStorePassword", "changeit");
System.setProperty("javax.net.ssl.keyStore", keyStore);
System.setProperty("javax.net.ssl.keyStorePassword", "changeit");
props.put("mail.smtp.auth", "true");
props.put("mail.smtp.host", mailSunucu);
props.put("mail.smtp.starttls.enable", "true");
} else {
props.put("mail.smtp.host", varsayilanMailSunucu);
```

```
}
Session session = Session.getDefaultInstance(props, null);
MimeMessage message = new MimeMessage(session);
message.setFrom(new InternetAddress(mailUser));
message.addRecipient(Message.RecipientType.TO, new
InternetAddress(mail.getTo()));
message.setSubject(mail.getTitle());
if (duzOrHtml == 1) {
message.setText(mail.getBody());
}
if (mailAuth == 1) {
Transport transport = session.getTransport("smtp");
transport.connect(mailSunucu, 587, mailUser, mailPassword);
message.saveChanges();
transport.sendMessage(message, message.getAllRecipients());
transport.close();
} else {
Transport.send(message);
}
mesaj = "";
return true;
} catch (Exception e) {
mesaj = e.toString();
e.printStackTrace();
return false;
}
}
}
```

EK- C

FamilyInfo.java

```
package com.bayiroglu.model;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import com.bayiroglu.utils.DBConnection;
public class FamilyInfo {
private int familyId;
private String email="";
private String password="";
private String parentName="";
private String parentSurname="";
private String parentAddress="";
public FamilyInfo() {
}
public int getFamilyId() {
return familyId;
}
public void setFamilyId(int familyId) {
this.familyId = familyId;
}
public String getEmail() {
return email;
}
public void setEmail(String email) {
this.email = email;
}
public String getPassword() {
return password;
}
```

```
public void setPassword(String password) {
this.password = password;
}
public String getParentName() {
return parentName;
}
public void setParentName(String parentName) {
this.parentName = parentName;
}
public String getParentSurname() {
return parentSurname;
}
public void setParentSurname(String parentSurname) {
this.parentSurname = parentSurname;
}
public String getParentAddress() {
return parentAddress;
}
public void setParentAddress(String parentAddress) {
this.parentAddress = parentAddress;
}
public FamilyInfo(int familyId, String email, String password, String parentName,
String parentSurname, String parentAddress) {
super();
this.familyId = familyId;
this.email = email;
this.password = password;
this.parentName = parentName;
this.parentSurname = parentSurname;
this.parentAddress = parentAddress;
}
@Override
public String toString() {
```

```
return "FamilyInfo [familyId=" + familyId + ", email=" + email + ", password=" +
password + ", parentName=" + parentName + ", parentSurname=" + parentSurname
+ ", parentAddress=" + parentAddress + "]";
}
public static FamilyInfo getFamilyByUnameAndPass(String username, String
password) throws Exception {
String sql = "SELECT * FROM FAMILY_INFO WHERE EMAIL = '" + username
+ "' AND PASSWORD = '" + password +"'";
FamilyInfo family =null;
Connection conn = DBConnection.DBConnect();
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
if (rs.next()) {
family= new FamilyInfo();
do {
family.setEmail(rs.getString(2));
family.setFamilyId(rs.getInt(1));
family.setPassword(rs.getString(3));
family.setParentName(rs.getString(4));
family.setParentSurname(rs.getString(5));
family.setParentAddress(rs.getString(6));
} while(rs.next());
} else {
}
rs.close();
return family;
}
public  Boolean insert() throws Exception {
Connection conn = DBConnection.DBConnect();
String sql = "insert into family_info
(email,password,parent_name,parent_surname,parent_address) values (?,?,?,?,?)";
PreparedStatement ps = conn.prepareStatement(sql);
ps.setString(1, email.trim());
```

```
ps.setString(2, password.trim());
ps.setString(3, parentName.trim());
ps.setString(4, parentSurname.trim());
ps.setString(5, parentAddress.trim());
int rs = ps.executeUpdate();
if (rs==1)
{
return true;
}
return false;
}
}
```

KidsInfo.java

```
package com.bayiroglu.model;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
import com.bayiroglu.utils.DBConnection;
public class KidsInfo {
private int id;
private int familyId;
private String name;
private String surname;
private String phoneNumber;
private String imei;
private int isActive;
public KidsInfo() {
}
public int getId() {
return id;
```

```
}
public void setId(int id) {
this.id = id;
}
public int getFamilyId() {
return familyId;
}
public void setFamilyId(int familyId) {
this.familyId = familyId;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public String getSurname() {
return surname;
}
public void setSurname(String surname) {
this.surname = surname;
}
public String getPhoneNumber() {
return phoneNumber;
}
public void setPhoneNumber(String phoneNumber) {
this.phoneNumber = phoneNumber;
}
public String getImei() {
return imei;
}
public void setImei(String imei) {
this.imei = imei;
```

```
}
public int getIsActive() {
return isActive;
}
public void setIsActive(int isActive) {
this.isActive = isActive;
}
@Override
public String toString() {
return "KidsInfo [id=" + id + ", familyId=" + familyId + ", name=" + name + ",
surname=" + surname + ", phoneNumber=" + phoneNumber + ", imei=" + imei + ",
isActive=" + isActive + "]";
}
public KidsInfo(int id, int familyId, String name, String surname, String
phoneNumber, String imei, Byte isActive) {
super();
this.id = id;
this.familyId = familyId;
this.name = name;
this.surname = surname;
this.phoneNumber = phoneNumber;
this.imei = imei;
this.isActive = isActive;
}
public void insert() throws Exception {
Connection conn = DBConnection.DBConnect();
Integer executed = 0;
String sql = "insert into kids_info
(family_id,kids_name,kids_surname,phone_number) values (" + familyId + ",'" +
name + "','" + surname + "','" + phoneNumber + "')";
System.out.println("insert new kid : " + sql);
PreparedStatement ps = conn.prepareStatement(sql);
executed = ps.executeUpdate();
```

```
ps.close();
}
public void update() throws Exception {
Connection conn = DBConnection.DBConnect();
Integer executed = 0;
String sql = "update kids_info set kids_name ='" + name + "',kids_surname ='" +
surname + "',phone_number ='" + phoneNumber + "' where kids_id = " + id;
System.out.println("update new kid : " + sql);
PreparedStatement ps = conn.prepareStatement(sql);
executed = ps.executeUpdate();
ps.close();
}
public ArrayList<KidsInfo> getKids(int familyId) throws Exception {
ArrayList<KidsInfo> kids = new ArrayList<KidsInfo>();
String sql = "select * from kids_info where family_id = " + familyId;
System.out.println("kids sql : " + sql);
Connection conn = DBConnection.DBConnect();
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
KidsInfo k = new KidsInfo();
k.setFamilyId(rs.getInt("family_id"));
k.setId(rs.getInt("kids_id"));
k.setName(rs.getString("kids_name"));
k.setSurname(rs.getString("kids_surname"));
k.setPhoneNumber(rs.getString("phone_number"));
k.setImei(rs.getString("phone_imei"));
k.setIsActive(rs.getByte("is_active"));
kids.add(k);
}
ps.close();
conn.close();
return kids;
```

```
}
public KidsInfo getKidById(Integer kidId) throws Exception {
KidsInfo kid = new KidsInfo();
Connection conn = DBConnection.DBConnect();
String sql = "select * from kids_info where kids_id = " + kidId;
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
kid.setFamilyId(rs.getInt(2));
kid.setId(rs.getInt(1));
kid.setName(rs.getString(3));
kid.setSurname(rs.getString(4));
kid.setPhoneNumber(rs.getString(5));
kid.setImei(rs.getString(6));
kid.setIsActive(rs.getByte(7));
}
rs.close();
return kid;
}
}
```

KidsLimit.java

```
package com.bayiroglu.model;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.ArrayList;
```

```
import org.apache.commons.lang.StringEscapeUtils;
import org.apache.taglibs.standard.lang.jstl.parser.ParseException;
import org.codehaus.jettison.json.JSONArray;
import org.codehaus.jettison.json.JSONException;
import org.codehaus.jettison.json.JSONObject;
import com.bayiroglu.utils.DBConnection;
public class KidsLimit {
private Integer id;
private Integer kidsId;
private String longitude;
private String latitude;
private Integer distance;
private Integer dayOfWeek;
private Integer startTime;
private Integer endTime;
private String address;
public KidsLimit() {
}
public Integer getId() {
return id;
}
public void setId(Integer id) {
this.id = id;
}
public Integer getKidsId() {
return kidsId;
}
public void setKidsId(Integer kidsId) {
this.kidsId = kidsId;
}
public String getLongitude() {
return longitude;
}
```

```
public void setLongitude(String longitude) {
this.longitude = longitude;
}
public String getLatitude() {
return latitude;
}
public void setLatitude(String latitude) {
this.latitude = latitude;
}
public Integer getDistance() {
return distance;
}
public void setDistance(Integer distance) {
this.distance = distance;
}
public Integer getDayOfWeek() {
return dayOfWeek;
}
public void setDayOfWeek(Integer dayOfWeek) {
this.dayOfWeek = dayOfWeek;
}
public Integer getStartTime() {
return startTime;
}
public void setStartTime(Integer startTime) {
this.startTime = startTime;
}
public Integer getEndTime() {
return endTime;
}
public void setEndTime(Integer endTime) {
this.endTime = endTime;
}
```

```
public String getAddress() {
return address;
}
public void setAddress(String address) {
this.address = address;
}
public KidsLimit(Integer id, Integer kidsId, String longitude, String latitude,
Integer distance, Integer dayOfWeek, Integer startTime,
Integer endTime, String address) {
super();
this.id = id;
this.kidsId = kidsId;
this.longitude = longitude;
this.latitude = latitude;
this.distance = distance;
this.dayOfWeek = dayOfWeek;
this.startTime = startTime;
this.endTime = endTime;
this.address = address;
}
@Override
public String toString() {
return "KidsLimit [id=" + id + ", kidsId=" + kidsId + ", longitude="
+ longitude + ", latitude=" + latitude + ", distance="
+ distance + ", dayOfWeek=" + dayOfWeek + ", startTime="
+ startTime + ", endTime=" + endTime + ", address=" + address
+ "]";
}
public void insert() throws Exception {
Connection conn = DBConnection.DBConnect();
Integer executed = 0;
String sql = "insert into kids_limits
(kids_id,longitude,latitude,distance,day_of_week,start_time,end_time,address) "
```

```
+ "values ("
+ kidsId
+ ","
+ longitude
+ ","
+ latitude
+ ","
+ distance
+ ","
+ dayOfWeek
+ ","
+ startTime
+ ","
+ endTime
+ ",'" + address + "')";
System.out.println("insert limit : " + sql);
PreparedStatement ps = conn.prepareStatement(sql);
executed = ps.executeUpdate();
ps.close();
}
public ArrayList<KidsLimit> getLimit(Long kId) throws Exception {
ArrayList<KidsLimit> limit = new ArrayList<KidsLimit>();
Connection conn = DBConnection.DBConnect();
String sql = "select * from kids_limits where kids_id = " + kId;
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
KidsLimit l = new KidsLimit();
l.setKidsId(rs.getInt(2));
l.setId(rs.getInt(1));
l.setLongitude(rs.getString(3));
l.setLatitude(rs.getString(4));
l.setDistance(rs.getInt(5));
```

```
l.setDayOfWeek(rs.getInt(6));
l.setStartTime(rs.getInt(7));
l.setEndTime(rs.getInt(8));
String address = getAddresInfo(rs.getString(3), rs.getString(4));
l.setAddress(address);
limit.add(l);
}
rs.close();
return limit;
}
public String getAddresInfo(String longitude, String latitude)
throws IOException, ParseException {
String link = "http://maps.googleapis.com/maps/api/geocode/json?latlng="
+ latitude.toString()
+ ","
+ longitude.toString()
+ "&sensor=false";
System.out.println(link);
URL url;
try {
url = new URL(link);
BufferedReader in = new BufferedReader(new InputStreamReader(
url.openStream()));
String inputLine;
String jsonResult = "";
while ((inputLine = in.readLine()) != null) {
jsonResult += inputLine + " ";
}
JSONObject json = new JSONObject(jsonResult);
JSONArray jsonArray = json.getJSONArray("results");
JSONObject addressInfo = (JSONObject) jsonArray.get(0);
String address = addressInfo.getString("formatted_address");
byte[] ba= address.getBytes();
```

```
return new String(ba,"UTF-8");
} catch (MalformedURLException e) {
e.printStackTrace();
} catch (JSONException e) {
e.printStackTrace();
}
return null;
}
}
```

KidsLocation.java

```
package com.bayiroglu.model;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.ParseException;
import java.util.ArrayList;
import java.util.Date;
import org.codehaus.jettison.json.JSONArray;
import org.codehaus.jettison.json.JSONException;
import org.codehaus.jettison.json.JSONObject;
import com.bayiroglu.utils.DBConnection;
public class KidsLocation {
Integer id;
String phoneNumber;
Date logDate;
String longitude;
```

```
String latitude;
String address;
public KidsLocation() {
}
public String getAddress() {
return address;
}
public void setAddress(String address) {
this.address = address;
}
public Integer getId() {
return id;
}
public void setId(Integer id) {
this.id = id;
}
public String getPhoneNumber() {
return phoneNumber;
}
public void setPhoneNumber(String phoneNumber) {
this.phoneNumber = phoneNumber;
}
public Date getLogDate() {
return logDate;
}
public void setLogDate(Date logDate) {
this.logDate = logDate;
}
public String getLongitude() {
return longitude;
}
public void setLongitude(String longitude) {
this.longitude = longitude;
```

```
}
public String getLatitude() {
return latitude;
}
public void setLatitude(String latitude) {
this.latitude = latitude;
}
@Override
public String toString() {
return "KidsLocation [id=" + id + ", phoneNumber=" + phoneNumber
+ ", logDate=" + logDate + ", longitude=" + longitude
+ ", latitude=" + latitude + "]";
}
public KidsLocation(Integer id, String phoneNumber, Date logDate,
String longitude, String latitude) {
super();
this.id = id;
this.phoneNumber = phoneNumber;
this.logDate = logDate;
this.longitude = longitude;
this.latitude = latitude;
}
public ArrayList<KidsLocation> getKidsLoc(String kidId, String date,
String startHour, String endHour) throws Exception {
ArrayList<KidsLocation> loc = new ArrayList<KidsLocation>();
String a = date + " " + startHour;
String b = date + " " + endHour;
String sql = "select l.* from kids_location l left outer join kids_info k on
k.phone_number = l.phone_number "
+ "where l.log_date between '"
+ a
+ "'  and '"
+ b
```

```
+ "' and k.kids_id = " + kidId + " order by 1 desc;";
Connection conn = DBConnection.DBConnect();
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
KidsLocation lc = new KidsLocation();
lc.setId(rs.getInt(1));
lc.setLogDate(rs.getDate(3));
lc.setPhoneNumber(rs.getString(2));
lc.setLatitude(rs.getString(5));
lc.setLongitude(rs.getString(4));
loc.add(lc);
}
rs.close();
return loc;
}
public ArrayList<KidsLocation> getLastLocation(String kidId)
throws Exception {
ArrayList<KidsLocation> list = new ArrayList<KidsLocation>();
KidsLocation loc = new KidsLocation();
String sql = "select l.* from kids_location l left outer join kids_info k on
k.phone_number = l.phone_number "
+ "where k.kids_id = " + kidId + " order by id desc limit 1;";
System.out.println("kids  loc enes : " + sql);
Connection conn = DBConnection.DBConnect();
PreparedStatement ps = conn.prepareStatement(sql);
ResultSet rs = ps.executeQuery();
while (rs.next()) {
String lat = rs.getString(5);
String lng = rs.getString(4);
String address = loc.getAddresInfo(lng, lat);
loc.setLatitude(lat);
loc.setLongitude(lng);
```

```
loc.setAddress(address);
list.add(loc);
}
rs.close();
return list;
}
public String getAddresInfo(String longitude, String latitude)
throws IOException, ParseException {
String link = "http://maps.googleapis.com/maps/api/geocode/json?latlng="
+ latitude.toString()
+ ","
+ longitude.toString()
+ "&sensor=false";
System.out.println(link);
URL url;
try {
url = new URL(link);
BufferedReader in = new BufferedReader(new InputStreamReader(
url.openStream()));
String inputLine;
String jsonResult = "";
while ((inputLine = in.readLine()) != null) {
jsonResult += inputLine + " ";
}
JSONObject json = new JSONObject(jsonResult);
JSONArray jsonArray = json.getJSONArray("results");
JSONObject addressInfo = (JSONObject) jsonArray.get(0);
String address = addressInfo.getString("formatted_address");
return address;
} catch (MalformedURLException e) {
e.printStackTrace();
} catch (JSONException e) {
e.printStackTrace();
```

```
}
return null;
}
}
```

Violation.java

```
package com.bayiroglu.model;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.MalformedURLException;
import java.net.URL;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.ParseException;
import java.util.ArrayList;
import java.util.Date;
import org.codehaus.jettison.json.JSONArray;
import org.codehaus.jettison.json.JSONException;
import org.codehaus.jettison.json.JSONObject;
import com.bayiroglu.utils.DBConnection;
public class Violation {
Integer id;
Integer kidsId;
Integer kidsLimitId;
String inst;
Date dateTime;
Integer isHandle;
String longitude;
String latitude;
String address;
```

```
String kidsName;
String kidsSurname;
public Violation() {
}
public String getKidsName() {
return kidsName;
}
public void setKidsName(String kidsName) {
this.kidsName = kidsName;
}
public String getKidsSurname() {
return kidsSurname;
}
public void setKidsSurname(String kidsSurname) {
this.kidsSurname = kidsSurname;
}
public Integer getId() {
return id;
}
public String getAddress() {
return address;
}
public void setAddress(String address) {
this.address = address;
}
public void setId(Integer id) {
this.id = id;
}
public Integer getKidsId() {
return kidsId;
}
public void setKidsId(Integer kidsId) {
this.kidsId = kidsId;
```

```
}
public Integer getKidsLimitId() {
return kidsLimitId;
}
public void setKidsLimitId(Integer kidsLimitId) {
this.kidsLimitId = kidsLimitId;
}
public String getInst() {
return inst;
}
public void setInst(String inst) {
this.inst = inst;
}
public Date getDateTime() {
return dateTime;
}
public void setDateTime(Date dateTime) {
this.dateTime = dateTime;
}
public Integer getIsHandle() {
return isHandle;
}
public void setIsHandle(Integer isHandle) {
this.isHandle = isHandle;
}
public String getLongitude() {
return longitude;
}
public void setLongitude(String longitude) {
this.longitude = longitude;
}
public String getLatitude() {
return latitude;
```

```
}
public void setLatitude(String latitude) {
this.latitude = latitude;
}
public Violation(Integer id, Integer kidsId, Integer kidsLimitId,
String inst, Date dateTime, Integer isHandle) {
super();
this.id = id;
this.kidsId = kidsId;
this.kidsLimitId = kidsLimitId;
this.inst = inst;
this.dateTime = dateTime;
this.isHandle = isHandle;
}
@Override
public String toString() {
return "Violation [id=" + id + ", kidsId=" + kidsId + ", kidsLimitId="
+ kidsLimitId + ", inst=" + inst + ", dateTime=" + dateTime
+ ", isHandle=" + isHandle + "]";
}
public ArrayList<Violation> getViolaitonByKidsId(Integer kidsId)
throws Exception {
ArrayList<Violation> vList = new ArrayList<Violation>();
String sql = "SELECT v.*,l.longitude,l.latitude ,k.kids_name,k.kids_surname"
+ " FROM violations_info v,kids_location l,kids_info k WHERE"
+ " k.phone_number = l.phone_number and "
+ " k.kids_id = v.kids_id and "
+ " date_format(l.log_date,'%d%m%Y%H')=date_format(v.date,'%d%m%Y%H')
and"
+ " v.KIDS_ID = ? limit 20";
Connection conn = DBConnection.DBConnect();
PreparedStatement ps = conn.prepareStatement(sql);
ps.setInt(1, kidsId);
```

```
ResultSet rs = ps.executeQuery();
while (rs.next()) {
Violation v = new Violation();
v.setId(rs.getInt(1));
v.setKidsId(rs.getInt(2));
v.setKidsLimitId(rs.getInt(3));
v.setDateTime(rs.getDate(4));
v.setInst(rs.getString(5));
v.setIsHandle(rs.getInt(6));
v.setLongitude(rs.getString(7));
v.setLatitude(rs.getString(8));
v.setKidsName(rs.getString(9));
v.setKidsSurname(rs.getString(10));
String address = v.getAddresInfo(rs.getString(7), rs.getString(8));
v.setAddress(address);
vList.add(v);
}
rs.close();
return vList;
}
public String getAddresInfo(String longitude, String latitude)
throws IOException, ParseException {
String link = "http://maps.googleapis.com/maps/api/geocode/json?latlng="
+ latitude.toString()
+ ","
+ longitude.toString()
+ "&sensor=false";
System.out.println(link);
URL url;
try {
url = new URL(link);
BufferedReader in = new BufferedReader(new InputStreamReader(
url.openStream()));
```

```
String inputLine;
String jsonResult = "";
while ((inputLine = in.readLine()) != null) {
jsonResult += inputLine + " ";
}
JSONObject json = new JSONObject(jsonResult);
JSONArray jsonArray = json.getJSONArray("results");
JSONObject addressInfo = (JSONObject) jsonArray.get(0);
String address = addressInfo.getString("formatted_address");
return address;
} catch (MalformedURLException e) {
e.printStackTrace();
} catch (JSONException e) {
e.printStackTrace();
}
return null;
}
}
```

DBConnection.java

```
package com.bayiroglu.utils;
import java.sql.Connection;
import java.sql.DriverManager;
public class DBConnection {
private static String url = "jdbc:mysql://localhost:3306/";
private static String dbName = "KidsTrackingDB";
private static String driver = "com.mysql.jdbc.Driver";
private static String userName = "KidsTrackingDBA";
private static String password = "12qw34er";
public static Connection DBConnect() throws Exception {
Class.forName(driver);
```

```
Connection conn = DriverManager.getConnection(url + dbName, userName,
password);
return conn;
}
}
```

addlimit.jsp

```
<%@page import="org.apache.log4j.helpers.NullEnumeration"%>
<%@page import="com.bayiroglu.model.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
<link rel="stylesheet" href="static/bootstrap/css/bootstrap.css">
<link rel="stylesheet" href="css/map.css">
<script type="text/javascript"
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDrDXJH98sT*********
*******&sensor=true">
</script>
</head>
<body>
<script>
var geocoder;
var map;
function initialize() {
geocoder = new google.maps.Geocoder();
var latlng = new google.maps.LatLng(41, 29);
var mapOptions = {
zoom : 8,
center : latlng,
```

```
mapTypeId : google.maps.MapTypeId.ROADMAP
}
map = new google.maps.Map(document.getElementById('map-canvas'),
mapOptions);
}
function codeAddress() {
var address = document.getElementById('address').value;
geocoder.geocode(
{
'address' : address
},
function(results, status) {
if (status == google.maps.GeocoderStatus.OK) {
map.setCenter(results[0].geometry.location);
var marker = new google.maps.Marker({
map : map,
position : results[0].geometry.location
});
var lat1 = results[0].geometry.location
.lat();
var lng1 = results[0].geometry.location
.lng();
alert(lat1);
alert(lng1);
document.getElementById('latitude').value = lat1;
document.getElementById('longitude').value = lng1;
} else {
alert('Geocode was not successful for the following reason: '
+ status);
}
});
}
google.maps.event.addDomListener(window, 'load', initialize);
```

```
</script>
<%
String method = request.getParameter("method");
String kidId = (String) session.getAttribute("kidsId");
if (method == null) {
} else if (method.equals("Save")) {
String latitude = request.getParameter("latitude");
String longitude = request.getParameter("longitude");
String address = request.getParameter("address");
Integer distance = Integer.parseInt(request
.getParameter("distance"));
String startHour = request.getParameter("startTime");
String endHour = request.getParameter("endTime");
Integer startTime = Integer.parseInt(startHour.split(":")[0])
* 60 + Integer.parseInt(startHour.split(":")[1]);
Integer endTime = Integer.parseInt(endHour.split(":")[0]) * 60
+ Integer.parseInt(endHour.split(":")[1]);
Integer dayOfWeek = Integer.parseInt(request
.getParameter("day"));
if (distance != null && longitude != null && latitude != null
&& startTime != null && endTime != null
&& dayOfWeek != null) {
KidsLimit limit = new KidsLimit();
limit.setKidsId(Integer.parseInt(kidId));
limit.setAddress(address);
limit.setLatitude(latitude);
limit.setLongitude(longitude);
limit.setDistance(distance);
limit.setStartTime(startTime);
limit.setEndTime(endTime);
limit.setDayOfWeek(dayOfWeek);
limit.insert();
} else {
```

```
%>
<div class="message">
Error!<br /> <font color="red">Fill all fields!</font>
</div>
<%
}
}
%>
<div class="container">
<form action="/KidsTrackingWebApp/addlimit.jsp">
<div id="panel">
<input id="address" type="text" value="" name="address"> <input
type="button" value="Find Address" onclick="codeAddress()"
class="btn">
</div>
<div id="map-canvas"
style="position: relative; background-color: rgb(229, 227, 223); overflow: hidden;">
</div>
<div class="limit">
<table>
<caption>Kids Limit</caption>
<tr>
<td>LONGITUDE</td>
<td><input type="text" id="longitude" name="longitude"
style="padding: 0" /></td>
</tr>
<tr>
<td>LATITUDE</td>
<td><input type="text" id="latitude" name="latitude"
style="padding: 0" /></td>
</tr>
<tr>
<td>DISTANCE</td>
```

```
<td><input type="text" name="distance" style="padding: 0" /></td>
</tr>
<tr>
<td>START TIME</td>
<td><input type="text" name="startTime" style="padding: 0" /></td>
<td><label>Time format should be HH:mm </label></td>
</tr>
<tr>
<td>END TIME</td>
<td><input type="text" name="endTime" style="padding: 0" /></td>
<td><label>Time format should be HH:mm </label></td>
</tr>
<tr>
<td>DAY OF WEEK</td>
<td><select name="day" onclick="">
<option value="1">SUNDAY</option>
<option value="2">MONDAY</option>
<option value="3">TUESDAY</option>
<option value="4">WENDESDAY</option>
<option value="5">THURSDAY</option>
<option value="6">FRIDAY</option>
<option value="7">SATURDAY</option>
</select></td>
</tr>
</table>
<input type="submit" name="method" value="Save" class="btn" />
</div>
</form>
</div>
</body>
</html>
```

kidsedit.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib uri="http://displaytag.sf.net" prefix="display"%>
<%@page import="java.util.*"%>
<%@page import="com.bayiroglu.model.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<link rel="stylesheet" href="static/bootstrap/css/bootstrap.css">
</head>
<body>
<script src="http://code.jquery.com/jquery.js"></script>
<script src="static/bootstrap/js/bootstrap.min.js"></script>
<div class="navbar navbar-fixed-top">
<div class="navbar-inner">
<div class="container container-fluid">
<a class="brand">KidsTrackingService</a>
<div class="nav nav-pills pull-right">
<ul class="nav">
<li>
<%
FamilyInfo family = (FamilyInfo) session.getAttribute("UserInfo");
%> <%
if (family == null) {
%> <c:redirect url="../Login.jsp">
</c:redirect> <%
} else {
%> <b>Welcome <br /> <font color='white'> <%=family.getParentName() + "-" +
family.getParentSurname()%></b></font>
<%
```

```
}
%>
</li>
</ul>
</div>
</div>
</div>
</div>
<hr>
<%
String method = request.getParameter("method");
System.out.println("edit metot : " + method);
String kidsId = request.getParameter("id");
System.out.println("edit kidsid : " + kidsId);
KidsInfo kid = new KidsInfo();
if (method == null) {
session.setAttribute("kidId", kidsId);
System.out.println("metot bos ");
session.setAttribute("kid", null);
if (kidsId != null) {
session.setAttribute("kidId", kidsId);
String b = (String) session.getAttribute("kidId");
System.out.println("edit b : " + b);
session.setAttribute("kid", kid.getKidById(Integer.parseInt(kidsId)));
}
} else {
String a = (String) session.getAttribute("kidId");
String kidName = request.getParameter("kidsName");
String kidSurname = request.getParameter("kidsSurname");
String phoneNumber = request.getParameter("phoneNumber");
if (method.equals("Save") && a == null) {
if (kidName != null && kidSurname != null && phoneNumber != null) {
kid.setFamilyId(family.getFamilyId());
```

```
kid.setName(kidName);
kid.setSurname(kidSurname);
kid.setPhoneNumber(phoneNumber);
kid.insert();
%>
<c:redirect url="/kidsinfo.jsp"></c:redirect>
<%
} else {
%>
<div class="message">
All field required!<br /> <font color="red"> <%
}
} else if (a != null && method.equals("Save")) {
if (kidName != null && kidSurname != null && phoneNumber != null) {
kid.setFamilyId(family.getFamilyId());
kid.setName(kidName);
kid.setSurname(kidSurname);
kid.setPhoneNumber(kidSurname);
kid.setId(Integer.parseInt(a));
kid.update();
%> <c:redirect url="/kidsinfo.jsp"></c:redirect> <%
} else {
%>
<div class="message">
All field required!<br /> <font color="red"> <%
}
}
}
%>
<form action="">
<table>
<caption>Kids Information</caption>
<c:choose>
```

```
<c:when test="${not empty kid }">
<tr>
<td>KIDS NAME</td>
<td><input type="text" name="kidsName" style="padding: 0"
value="<c:out value="${kid.name}"/>" /></td>
<tr>
<td>KIDS SURNAME</td>
<td><input type="text" name="kidsSurname"
style="padding: 0" value="<c:out value="${kid.surname}"/>" /></td>
</tr>
<tr>
<td>KIDS PHONE NUMBER</td>
<td><input type="text" name="phoneNumber"
style="padding: 0"
value="<c:out value="${kid.phoneNumber}"/>" /></td>
</tr>
</c:when>
<c:otherwise>
<tr>
<td>KIDS NAME</td>
<td><input type="text" name="kidsName" style="padding: 0" /></td>
<tr>
<td>KIDS SURNAME</td>
<td><input type="text" name="kidsSurname"
style="padding: 0" /></td>
</tr>
<tr>
<td>KIDS PHONE NUMBER</td>
<td><input type="text" name="phoneNumber"
style="padding: 0" /></td>
</tr>
</c:otherwise>
</c:choose>
```

```
</table>
<input type="submit" name="method" value="Save" class="button" />
</form>
</body>
</html>
```

kidsinfo.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib uri="http://displaytag.sf.net" prefix="display"%>
<%@page import="java.util.*"%>
<%@page import="com.bayiroglu.model.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<link rel="stylesheet" href="static/bootstrap/css/bootstrap.css">
</head>
<body>
<script src="http://code.jquery.com/jquery.js"></script>
<script src="static/bootstrap/js/bootstrap.min.js"></script>
<div class="navbar navbar-fixed-top">
<div class="navbar-inner">
<div class="container container-fluid">
<a class="brand">KidsTrackingService</a>
<div class="nav nav-pills pull-right">
<ul class="nav">
<li>
<%
FamilyInfo family = (FamilyInfo) session.getAttribute("UserInfo");
%> <%
if (family == null) {
```

```
%> <c:redirect url="/login.jsp">
</c:redirect> <%
} else {
%> <b>Welcome <br /> <font color='white'> <%=family.getParentName() + "-" +
family.getParentSurname()%></b></font>
<%
}
%>
</li>
</ul>
</div>
</div>
</div>
</div>
<hr>
<%
System.out.println("kid family : " + family.getFamilyId());
String method = request.getParameter("method");
KidsInfo k = new KidsInfo();
ArrayList<KidsInfo> kids = new ArrayList<KidsInfo>();
if (method == null) {
if (family != null) {
kids = k.getKids(family.getFamilyId());
session.setAttribute("kid", kids);
} else {
//kids = k.getKidsWithLimit();
//session.setAttribute("kids", kids);
}
} else if (method.equals("Add")) {
%>
<c:redirect url="/kidsedit.jsp"></c:redirect>
<%
}
```

```
%>
<form action="">
<input type="submit" name="method" value="Add" class="btn" />
<display:table name="sessionScope.kid" id="kid"
class="table table-striped table-bordered">
<display:column title="Kids Name" href="kidsedit.jsp" paramId="id"
paramProperty="id">
<c:out value="${kid.name}" />
</display:column>
<display:column title="Kids Surname">
<c:out value="${kid.surname}" />
</display:column>
<display:column title="Phone Number">
<c:out value="${kid.phoneNumber}" />
</display:column>
<display:column title="IMEI">
<c:out value="${kid.imei}" />
</display:column>
<display:column paramProperty="id" paramId="id"
href="/KidsTrackingWebApp/violation.jsp" value="violation"></display:column>
<display:column>
<a href="kidslimit.jsp?id=${kid.id}">Add Limit</a>
</display:column>
<display:column>
<a href="location.jsp?id=${kid.id}">Where is my kid?</a>
</display:column>
</display:table>
</form>
</body>
</html>
```

kidslimit.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib uri="http://displaytag.sf.net" prefix="display"%>
<%@page import="java.util.*"%>
<%@page import="com.bayiroglu.model.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<link rel="stylesheet" href="static/bootstrap/css/bootstrap.css">
</head>
<body>
<script src="http://code.jquery.com/jquery.js"></script>
<script src="static/bootstrap/js/bootstrap.min.js"></script>
<div class="navbar navbar-fixed-top">
<div class="navbar-inner">
<div class="container container-fluid">
<a class="brand">KidsTrackingService</a>
<div class="nav nav-pills pull-right">
<ul class="nav">
<li>
<%
FamilyInfo family = (FamilyInfo) session.getAttribute("UserInfo");
%> <%
if (family == null) {
%> <c:redirect url="../Login.jsp">
</c:redirect> <%
} else {
%> <b>Welcome <br /> <font color='white'> <%=family.getParentName() + "-" +
family.getParentSurname()%></b></font>
<%
```

```
}
%>
</li>
</ul>
</div>
</div>
</div>
</div>
<hr>
<%
String method = request.getParameter("method");
System.out.println("metod : " + method);
String kidsId = request.getParameter("id");
KidsLimit l = new KidsLimit();
ArrayList<KidsLimit> kids = new ArrayList<KidsLimit>();
if (method == null) {
kids = l.getLimit(Long.parseLong(kidsId));
session.setAttribute("limit", kids);
session.setAttribute("kidsId", kidsId);
} else if (method.equals("Add")) {
%>
<c:redirect url="/addlimit.jsp">
<c:param name="id" value="${kidsId }"></c:param>
</c:redirect>
<%
}
%>
<form action="">
<input type="submit" name="method" value="Add" class="btn" />
<display:table name="sessionScope.limit" id="a"
class="table table-striped table-bordered">
<display:column title="Address">
<c:out value="${a.address}" />
```

```
</display:column>
<display:column title="Distance">
<c:out value="${a.distance}" />
</display:column>
<display:column title="Day of Week">
<c:out value="${a.dayOfWeek}" />
</display:column>
<display:column title="Start Time">
<c:out value="${a.startTime}" />
</display:column>
<display:column title="End Time">
<c:out value="${a.endTime}" />
</display:column>
</display:table>
</form>
</body>
</html>
```

location.jsp

```
<%@page import="java.util.ArrayList"%>
<%@page import="java.util.*"%>
<%@page import="com.bayiroglu.model.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
<link rel="stylesheet" href="static/bootstrap/css/bootstrap.css">
<link rel="stylesheet" href="css/map.css">
<script type="text/javascript"
```

```
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDrDXJH98**********
***&sensor=true">
</script>
<link rel="stylesheet"
href="http://code.jquery.com/ui/1.10.3/themes/smoothness/jquery-ui.css" />
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="http://code.jquery.com/ui/1.10.3/jquery-ui.js"></script>
</head>
<body>
<script src="http://code.jquery.com/jquery.js"></script>
<script src="static/bootstrap/js/bootstrap.min.js"></script>
<script type="text/javascript">
$(document).ready(
function() {
var lats = [];
var lngs = [];
var points = [];
var loclat1 = [];
var loclng1 = [];
$('.loclng').each(function() {
var lat = '';
$(this).find('li').each(function() {
var current = $(this);
if (current.children().size() > 0) {
return true;
}
lat += $(this).text();
});
loclng1.push(lat);
});
$('.loclat').each(function() {
var lat = '';
$(this).find('li').each(function() {
```

```
var current = $(this);
if (current.children().size() > 0) {
return true;
}
lat += $(this).text();
});
loclat1.push(lat);
});
$('.lng').each(function() {
var lat = '';
$(this).find('li').each(function() {
var current = $(this);
if (current.children().size() > 0) {
return true;
}
lat += $(this).text();
});
lngs.push(lat);
});
$('.lat').each(function() {
var lat = '';
$(this).find('li').each(function() {
var current = $(this);
if (current.children().size() > 0) {
return true;
}
lat += $(this).text();
});
lats.push(lat);
});
var mycenter = new google.maps.LatLng(loclat1[0],
loclng1[0]);
for ( var i = 0; i < lats.length; i++) {
```

```
var lat1 = lats[i];
var lng1 = lngs[i];
points.push(new google.maps.LatLng(lat1, lng1));
}
function initialize() {
var mapProp = {
center : mycenter,
zoom : 5,
mapTypeId : google.maps.MapTypeId.ROADMAP
};
var map = new google.maps.Map(document
.getElementById("map-canvas"), mapProp);
var flightPath = new google.maps.Polyline({
path : points,
strokeColor : '#FF0000',
strokeOpacity : 1.0,
strokeWeight : 2
});
var marker = new google.maps.Marker({
position : mycenter,
map : map,
title : 'Hello World!'
});
flightPath.setMap(map);
}
google.maps.event
.addDomListener(window, 'load', initialize);
});
</script>
<div class="navbar navbar-fixed-top">
<div class="navbar-inner">
<div class="container container-fluid">
<a class="brand">KidsTrackingService</a>
```

```
<div class="nav nav-pills pull-right">
<ul class="nav">
<li>
<%
FamilyInfo family = (FamilyInfo) session.getAttribute("UserInfo");
%> <%
if (family == null) {
%> <c:redirect url="/login.jsp">
</c:redirect> <%
} else {
%> <b>Welcome <br /> <font color='white'> <%=family.getParentName() + "-"
+ family.getParentSurname()%></b></font> <%
}
%>
</li>
</ul>
</div>
</div>
</div>
</div>
<hr>
<%
String method = request.getParameter("method");
String kidsId = request.getParameter("id");
System.out.println("metot : " + method);
KidsLocation l = new KidsLocation();
ArrayList<KidsLocation> loc = new ArrayList<KidsLocation>();
if (method == null) {
ArrayList<KidsLocation> kl;
kl = l.getLastLocation(kidsId);
System.out.println("lati :"+kl.get(0).getLatitude());
System.out.println("long :"+kl.get(0).getLongitude());
session.setAttribute("lat", kl.get(0).getLatitude());
```

```
session.setAttribute("lng", kl.get(0).getLongitude());
session.setAttribute("kidId", kidsId);
session.setAttribute("list", null);
} else {
String date = request.getParameter("date");
String stime = request.getParameter("startTime");
String etime = request.getParameter("endTime");
String kidId = (String) session.getAttribute("kidId");
System.out.println("kidid : " + Integer.parseInt(kidId));
loc = l.getKidsLoc(kidId, date, stime, etime);
System.out.println("kidid : " + loc.get(0).getLongitude());
session.setAttribute("list", loc);
ArrayList<KidsLocation> kl = new ArrayList<KidsLocation>();
kl = l.getLastLocation(kidId);
session.setAttribute("lat", kl.get(0).getLatitude());
session.setAttribute("lng", kl.get(0).getLongitude());
}
ArrayList<KidsLocation> kl = new ArrayList<KidsLocation>();
%>
<div id="map-canvas"></div>
<div class="modal hide fade">
<div id="testLng" class="loclng" style="visibility: hidden;">
<ul class="items">
<li>${lng }</li>
</ul>
</div>
<div id="testLat" class="loclat" style="visibility: hidden;">
<ul class="items">
<li>${lat }</li>
</ul>
</div>
<c:forEach var="i" items="${ list}">
<div id="testLng" class="lng" style="visibility: hidden;">
```

```
<ul class="items">
<li><c:out value="${i.longitude }"></c:out></li>
</ul>
</div>
<div id="testLng" class="lat" style="visibility: hidden;">
<ul class="items">
<li><c:out value="${i.latitude }"></c:out></li>
</ul>
</div>
</c:forEach>
</div>
<form action="/KidsTrackingWebApp/location.jsp">
<table>
<tr>
<td>DATE</td>
<td><input type="text" name="date" style="padding: 0" /></td>
<td><label>Date format should be YYYY-MM-DD </label></td>
</tr>
<tr>
<td>START TIME</td>
<td><input type="text" name="startTime" style="padding: 0" /></td>
<td><label>Time format should be HH:mm </label></td>
</tr>
<tr>
<td>END TIME</td>
<td><input type="text" name="endTime" style="padding: 0" /></td>
<td><label>Time format should be HH:mm </label></td>
</tr>
</table>
<input type="submit" name="method" value="Save" class="btn" />
</form>
</body>
</html>
```

login.jsp

```
<%@page import="com.bayiroglu.model.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib uri="http://displaytag.sf.net" prefix="display"%>
<%@page import="java.util.*"%>
<%@page import="com.bayiroglu.model.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<meta name="apple-mobile-web-app-capable" content="yes" />
<meta name="apple-mobile-web-app-status-bar-style" content="black" />
<meta name="viewport"
content="width=device-width,initial-scale=1,user-scalable=no,maximum-scale=1" />
<link rel="stylesheet" href="static/bootstrap/css/bootstrap.css">
</head>
<body id="login">
<script src="http://code.jquery.com/jquery.js"></script>
<script src="static/bootstrap/js/bootstrap.min.js"></script>
<div class="navbar navbar-fixed-top">
<div class="navbar-inner">
<div class="container-fluid">
<button class="btn btn-navbar" data-target=".nav-collapse"
data-toggle="collapse" type="button">
<span class="icon-bar"></span> <span class="icon-bar"></span> <span
class="icon-bar"></span>
</button>
<a class="brand" href="#">KidsTrackingService</a>
</div>
</div>
</div>
```

```
<hr>
<%
String method = request.getParameter("method");
FamilyInfo f = null;
f = (FamilyInfo) session.getAttribute("user");
if (f == null) {
System.out.println("metod : " + method);
if (method != null && method.equals("Save")) {
FamilyInfo family = new FamilyInfo();
family.setEmail(request.getParameter("email"));
family.setParentName(request.getParameter("name"));
family.setParentSurname(request.getParameter("surname"));
family.setPassword(request.getParameter("pass"));
family.setParentAddress(request.getParameter("address"));
family.insert();
} else if (method != null && method.equals("Login")) {
String username = request.getParameter("username");
String password = request.getParameter("password");
System.out.println(username+password);
FamilyInfo family = new FamilyInfo();
family = FamilyInfo.getFamilyByUnameAndPass(username, password);
System.out.println("parent name" + family.getParentName());
if (family != null) {
session.setAttribute("UserInfo", family);
%>
<c:redirect url="/kidsinfo.jsp">
</c:redirect>
<%
} else {
%>
<div class="message">
Error!<br /> <font color="red">Kullanici adi veya sifre
yanlistir!</font>
```

```
</div>
<%
}
} else if (method != null && method.equals("Register")) {
%>
<c:redirect url="/register.jsp">
</c:redirect>
<%
}
} else {
%>
<c:redirect url="/kidsinfo.jsp">
</c:redirect>
<%
}
%>
<div class="container">
<form class="form-signin" method="post"
action="/KidsTrackingWebApp/login.jsp">
<h2 class="form-signin-heading">Please sign in</h2>
<input class="input-block-level" type="text" name="username"
id="username"> <input class="input-block-level"
type="password" name="password" id="password"> <label
class="checkbox"> <input type="checkbox" value="remember-me">
Remember me
</label> <input type="submit" name="method" value="Login" class="btn" />
<a href="#myModal" role="button" class="btn" data-toggle="modal">Register</a>
<!-- Modal -->
<div id="myModal" class="modal hide fade" tabindex="-1" role="dialog"
aria-labelledby="myModalLabel" aria-hidden="true">
<div class="modal-header">
<button type="button" class="close" data-dismiss="modal"
aria-hidden="true">×</button>
```

```
<h3 id="myModalLabel">Modal header</h3>
</div>
<div class="modal-body">
<table>
<tr>
<td>Email</td>
<td><input type="text" name="email" style="padding: 0" /></td>
</tr>
<tr>
<td>Password</td>
<td><input type="password" name="pass" style="padding: 0" /></td>
</tr>
<tr>
<td>Name</td>
<td><input type="text" name="name" style="padding: 0" /></td>
</tr>
<tr>
<td>Surname</td>
<td><input type="text" name="surname" style="padding: 0" /></td>
</tr>
<tr>
<td>Address</td>
<td><input type="text" name="address" style="padding: 0" /></td>
</tr>
</table>
</div>
<div class="modal-footer">
<button class="btn" data-dismiss="modal" aria-hidden="true">Close</button>
<input type="submit" name="method" value="Save" class="button" />
</div>
</div>
</form>
</div>
```

```
</body>
</html>
```

register.jsp

```
<%@page import="com.bayiroglu.model.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib uri="http://displaytag.sf.net" prefix="display"%>
<%@page import="com.bayiroglu.model.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<meta name="apple-mobile-web-app-capable" content="yes" />
<meta name="apple-mobile-web-app-status-bar-style" content="black" />
<meta name="viewport"
content="width=device-width,initial-scale=1,user-scalable=no,maximum-scale=1" />
<title>Insert title here</title>
<link rel="stylesheet" href="static/bootstrap/css/bootstrap.css">
</head>
<body>
<script src="http://code.jquery.com/jquery.js"></script>
<script src="static/bootstrap/js/bootstrap.min.js"></script>
<%
String method = request.getParameter("method");
FamilyInfo f = new FamilyInfo();
if (method == null) {
} else if (method != null && method.equals("Save")) {
FamilyInfo family = new FamilyInfo();
family.setEmail(request.getParameter("email"));
family.setParentName(request.getParameter("name"));
family.setParentSurname(request.getParameter("surname"));
```

```
family.setPassword(request.getParameter("pass"));
family.setParentAddress(request.getParameter("address"));
family.insert();
%>
<c:redirect url="login.jsp"></c:redirect>
<%
}
%>
<form action="">
<table>
<tr>
<td>Email</td>
<td><input type="text" name="email" style="padding: 0" /></td>
</tr>
<tr>
<td>Password</td>
<td><input type="password" name="pass" style="padding: 0" /></td>
</tr>
<tr>
<td>Name</td>
<td><input type="text" name="name" style="padding: 0" /></td>
</tr>
<tr>
<td>Surname</td>
<td><input type="text" name="surname" style="padding: 0" /></td>
</tr>
<tr>
<td>Address</td>
<td><input type="text" name="address" style="padding: 0" /></td>
</tr>
</table>
<input type="submit" name="method" value="Save" class="button" />
</form>
```

```
</body>
</html>
```

violation.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<%@taglib uri="http://displaytag.sf.net" prefix="display"%>
<%@page import="java.util.*"%>
<%@page import="com.bayiroglu.model.*"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<link rel="stylesheet" href="static/bootstrap/css/bootstrap.css">
<link rel="stylesheet" href="css/map.css">
<script type="text/javascript"
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDrDXJH98sT5********
***&sensor=true">
</script>
</head>
<body>
<script src="http://code.jquery.com/jquery.js"></script>
<script src="static/bootstrap/js/bootstrap.min.js"></script>
<script type="text/javascript">
var QueryString = function() {
var query_string = {};
var query = window.location.search.substring(1);
var vars = query.split("&");
for ( var i = 0; i < vars.length; i++) {
var pair = vars[i].split("=");
if (typeof query_string[pair[0]] === "undefined") {
query_string[pair[0]] = pair[1];
```

```
} else if (typeof query_string[pair[0]] === "string") {
var arr = [ query_string[pair[0]], pair[1] ];
query_string[pair[0]] = arr;
} else {
query_string[pair[0]].push(pair[1]);
}
}
return query_string;
}();
var lat1 = QueryString.lat;
var lng1 = QueryString.lng;
if (lat1 != null) {
var myCenter = new google.maps.LatLng(lat1, lng1);
} else {
var myCenter = new google.maps.LatLng(40.77, 30.40);
}
function initialize() {
var mapProp = {
center : myCenter,
zoom : 10,
mapTypeId : google.maps.MapTypeId.ROADMAP
};
var map = new google.maps.Map(
document.getElementById("map-canvas"), mapProp);
var marker = new google.maps.Marker({
position : myCenter,
});
marker.setMap(map);
}
google.maps.event.addDomListener(window, 'load', initialize);
</script>
<div class="navbar navbar-fixed-top">
<div class="navbar-inner">
```

```
<div class="container container-fluid">
<a class="brand">KidsTrackingService</a>
<div class="nav nav-pills pull-right">
<ul class="nav">
<li>
<%
FamilyInfo family = (FamilyInfo) session.getAttribute("UserInfo");
%> <%
if (family == null) {
%> <c:redirect url="../Login.jsp">
</c:redirect> <%
} else {
%> <b>Welcome <br /> <font color='white'> <%=family.getParentName() + "-" +
family.getParentSurname()%></b></font>
<%
}
%>
</li>
</ul>
</div>
</div>
</div>
</div>
<hr>
<%
String kidsId = request.getParameter("id");
ArrayList<Violation> vList = new ArrayList<Violation>();
Violation v = new Violation();
vList = v.getViolaitonByKidsId(Integer.parseInt(kidsId));
session.setAttribute("vList", vList);
%>
<div id="map-canvas"
style="position: relative; background-color: rgb(229, 227, 223); overflow: hidden;">
```

```
</div>
<form action="">
<table class="table">
<c:forEach var="i" items="${vList }">
<tr>
<td><c:out value="${i.kidsLimitId }"></c:out></td>
<td><c:out value="${i.kidsName }"></c:out></td>
<td><c:out value="${i.kidsSurname }"></c:out></td>
<td><c:out value="${i.dateTime }"></c:out></td>
<td><c:out value="${i.address }"></c:out></td>
<c:choose>
<c:when test="${i.isHandle == 1 }">
<td><c:out value="is handled"></c:out></td>
</c:when>
<c:otherwise>
<td><c:out value="is not handled"></c:out></td>
</c:otherwise>
</c:choose>
<td><a
href="/KidsTrackingWebApp/violation.jsp?id=${i.kidsId}&lat=${i.latitude}&lng=$
{i.longitude}">Click
Me</a></td>
</tr>
</c:forEach>
</table>
</form>
</body>
</html>
```

ÖZGEÇMİŞ

Hacer BAYIROĞLU, 26.03.1985'de Ünye'de doğdu. İlk, orta ve lise eğitimini Ünye'de tamamladı. 2003 yılında Ünye Anadolu İmam Hatip Lisesi'nden mezun oldu. 2004 yılında başladığı Yakın Doğu Üniversitesi Bilgisayar Mühendisliği (İngilizce) bölümünü 2009 yılında bitirdi. 2010 yılında Sakarya Üniversitesi Bilgisayar ve Bilişim Mühendisliği bölümünde yüksek lisansa başladı. Aynı zamanda Eylül 2010 tarihinden Şubat 2012 tarihine kadar Kilis 7 Aralık Üniversitesi Meslek Yüksek Okul 'da Öğretim Görevlisi Yetiştirme Programı kapsamında çalıştı. Şubat 2013 tarihinden itibaren Düzce Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği'nde Araştırma Görevlisi olarak çalışmaktadır.