T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ANDROİD İŞLETİM SİSTEMİ İÇİN MEDİKAL İZLEME YAZILIMI GELİŞTİRME

YÜKSEK LİSANS TEZİ

Faruk AKSOY
(111131103)
Anabilim Dalı: Elektronik ve Bilgisayar Eğitimi
Programı: Bilgisayar Sistemleri

Danışman: Prof. Dr. İbrahim TÜRKOĞLU

NİSAN-2014

T.C.
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ANDROİD İŞLETİM SİSTEMİ İÇİN MEDİKAL İZLEME YAZILIMI GELİŞTİRME

YÜKSEK LİSANS TEZİ

Faruk AKSOY
(111131103)

Tezin Enstitüye Verildiği Tarih: 18 NİSAN 2014
Tezin Savunulduğu Tarih:

Diğer Jüri Üyeleri :
Tez Danışman : Prof.Dr. İbrahim TÜRKOĞLU
Yrd. Doç. Dr. Resul DAŞ
Yrd. Doç. Dr. Oğuzhan ÖZDEMİR

NİSAN-2014

İÇİNDEKİLER

ÖZET

ANDROİD İŞLETİM SİSTEMİ İÇİN MEDİKAL İZLEME YAZILIMI GELİŞTİRME

Sabit platformlara kıyasla mobil sistemleri kullanmak hız, maliyet, erişim kolaylığı vb. alanlarda daha avantajlı görünmektedir. Bununla beraber sağlık bilgi sistemleri alanındaki işlemler her geçen gün yeni uygulamalarla dijital ortama katılmaktadır. Bu çalışmada mobil platform üzerinde çalışacak bir medikal takip sistemi geliştirilmesi hedeflenmiştir. Bu sistem sayesinde bebek aşıları, gebe kontrolleri ve kronik hastaların izlenmesi ve ilgili taraflara bu bilgilerin ulaştırılmasına çalışılmıştır. Geliştirilen sistem Eclipse, Java ve XML kullanılarak Android işletim sistemi üzerinde çalışacak şekilde tasarlanmıştır. Özellikle aşıların ve hastaların takibi açısından doktorlara büyük kolaylıklar getirmesi ve geniş kitlelere daha kolay ulaşılabilmesi ve uyarma sistemini de içermesi gibi özellikleri sayesinde sistemin tercih edilme oranının artması öngörülmektedir.

Anahtar Kelimeler - Mobil Cihazlar, Sağlık Bilgi Sistemleri, Medikal İzleme Sistemi, Android

SUMMARY

DEVELOPMENT OF MEDICAL MONITORING SOFTWARE FOR ANDROID OPERATING SYSTEMS

The use of mobile systems is more advantageous compared to the fixed platforms in terms of speed, cost, and ease of access etc. However, operations in the field of health information systems participate in digital environment by new applications every day. In this study aimed to develop a medical monitoring system work on mobile platform. With this system, it would be possible to monitor the baby's vaccinations, pregnancy checks and monitoring of chronically ill patients and to transport these information to interested parties. The system developed is designed to be run on the Android operating system by using Eclipse, Java and XML. The using rate of the program is expected to be increased since it comes with great conveniences especially for the doctors to follow up their patients and vaccines, has a potential of reaching out a wider population and includes a warning.

Keywords – Mobile devices, Health information systems, Medical monitoring system, Android

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

KISALTMALAR VE TANIMLAR

3G	**:**	"3. Generation". Geniş Alan devre anahtarlamalı ve hücresel paket mobil şebekesi için III. kuşak standart ve teknolojisi.
4G	**:**	"4. Generation". IP temelli, paket anahtarlamalı, geniş kapasiteli, ses, veri ve çoklu ortam servislerinin istenilen yerde ve zamanda sunulabildiği en son mobil teknolojisi
API		Application Programming Interface (Uygulama Programlama Arayüzü
ADT	**:**	Android development tools
BLUETOOTH	**:**	Kısa mesafede sabit ve mobil cihazlardan veri değişimi için geliştirilmiş bir telsiz iletişim protokolü ve bu özelliği destekleyen cihazların teknolojik özelliği.
GPRS	**:**	"General Packet Radio System". II. ve III. Kuşak Mobil İletişim sistemlerinde kullanılan paket temelli orta hızda mobil veri servisi sağlayan, IP, PPP ve X.25 bağlantılarını destekleyen servis
GSM	**:**	"Global System for Mobile Communications". İşaretleme ve konuşma kanallarının sayısal olan, farklı operatörler arasında dolaşım imkanı sağlayan, veri iletişimine imkan sağlayan mobil telefon operatörleri tarafından kabul edilen Uluslararası "Global System for Mobile Communication" II. Kuşak standardı.
İNTERNET	**:**	Dünya üzerinde sahipsiz, bilgi paylaşımı amacı ile ortaya çıkmış TCP/IP temelli bilgisayar ağı.
JAVA ME veya J2ME		Java Micro Edition
JDK	**:**	Java development kit
KAYNAK KODU	**:**	Bilgisayar makine diline çevrilmemiş ve derlenmemiş bilgisayar programı.
OS	**:**	Operation system=işletim sistemi
SDK	**:**	Software development kit
SMS	**:**	"Short Message Service". GSM Mobil iletişim sistemlerinde kullanılan, mobil telefon cihazları arasında 160 adet karakter ile iletişimin sağlandığı Mobil Kısa Mesaj Servisi.

VTYS : Veri Tabanı Yönetim Sistemi.

WAP : "Wireless Application Protocol". Telsiz Uygulama Protokolü. Düşük bant genişliği ve yüksek gecikme süreli telsiz cihazların iletişiminde kullanılan uygulama katmanı iletişim protokolü WAP-1 ve WAP-2 türleri vardır.

WEB : "World Wide Web".

WEP : "Wired Equivalent Privacy". Kablosuz ağ bağlantılarında çalışan şifreleme yöntemi.

WI-FI : "Wireless Fidelity". IEEE 802.11 telsiz teknolojilerini kullanan, farklı telsiz cihazların birlikte çalışabilmesi için oluşturulmuş akreditasyon standardı ve bu teknolojideki telsiz iletişim ağı

1. GİRİŞ

Günümüzde sabit sistemlerden mobil sistemlere doğru hızlı bir geçiş yaşanmaktadır.[1] Mobil platformda da yazılım ve yazılımcı ihtiyacı artmaktadır. Mobil sistemler içerisinde, Android işletim sistemi üzerinde çalışacak bir sağlık izleme sistemi geliştirilmeye üzerine çalışılmıştır.

Günümüzde kullanılan bazı mobil işletim sistemleri listelenecek olursa:

- Google: Android
- Nokia: Symbian
- Apple: iPhone OS
- Microsoft: Windows Mobile
- RIM: Blackberry OS
- Palm: WebOS
- Bada: Samsung

işletim sistemleri yaygın olarak kullanılmaktadır.

Bilişim dünyasında her cihazın mobil olmaya başladığı ortamda işletim sistemleri ve onların içerikleri büyük önem arzetmektedir. Çünkü yazılımlar ne kadar başarılı olursa olsun koşturuldukları alanlar işletim sistemleridir. Ayrıca tablo 1.1 ve tablo1.2'de görüldüğü gibi StatCounter tarafından yapılan araştırmalardaki kullanım oranlarına bakıldığında bu sistemler içerisinde dünyada ve ülkemizde en çok kullanım oranına sahip mobil işletim sisteminin Android olduğu görülmektedir.

Tablo 1.1. Türkiye'de ekim 2012-2013 mobil işletim sistemi kullanim oranlari

Android	59.34
iOS	16.6
SymbianOS	9.71
Series 40	8.95
Windows phone	1.69
Samsung	0.98
Bada	0.96

Tablo 1.2. Dünyada ekim 2012-2013 mobil işletim sistemi kullanim orani

Android	37.12
iOS	24.41
Series 40	13.79
SymbianOS	8.01
Samsung	5.1
Unknown	4.4
Blackberry OS	3.61
Windows Phone	1.31
Others	2.26

Dünyada [3] ve Türkiye'de [2] yapılan istatistik çalışmalarının sonuçlarından anlaşıldığı üzere mobil platformlarda Android işletim sisteminin kullanım oranına bakıldığında, sonraki yıllarda daha üst sıralara gelmesi tahmin edilmektedir.

1.1. Tezin Amacı

Mobil platformda çalışacak, sağlık alanında kullanılabilecek bir izleme sistemi geliştirilmesi hedeflenmiştir. Geliştirilecek medikal izleme sistemi ile bir aile hekiminin takip etmesi gereken;

- Bebek aşılarının
- Gebe kontrollerinin
- Kronik hastaların ve bunların kullandığı ilaçların izlenmesi için gerekli ortam oluşturularak gerekli bilgilendirme akışının doktor ve hasta tarafına verimli bir şekilde ulaştırılması ve bilgilerin sorgulanması esas alınmıştır.[4]

1.2. Tezin Önemi

Gelişen teknolojinin internetle beraber dinamizminin artmasıyla ve teknolojiye yeni kabiliyetler kazandırmasıyla mobil sistemler, sadece iletişim için kullanılan teknolojiler olmaktan , farklı platformları bir araya getirebilen bir merkez haline dönüştü. Sağlık [5], finans, robotik vs. sektörlerde gezgin sistemler, sabit sistemlere hâkim duruma geçtiler. Bu

gelişmelerle beraber günlük hayatı etkileyen her alana mobil sistemler entegre edilmeye ve hayata daha fazla dahil edilmeye başlandı. Bunun için gerekli olan yazılım ve donanım ürünleri daha cazip olmaya başladı. Artık ulaşmak istedinilen bilgilere, yapılması gereken işlere daha fazla mobilite kazandırdıkça yapılan işlere daha fazla hâkim olunmaktadır.

İnsanlar için geliştirilen teknolojilere bakıldığında insanın konforu için geliştirilen teknolojiler daima daha çok tercih edilir ve daha çok pazarlanabilir olmuştur. Bu günümüzde yapılan gözlemlerde rahatlıkla görülebilmektedir. Özellikle mobil sistemler içinde insanlara en çok lazım olabilecek ya da devamlı başvurulabilecek kaynakların bulundurulması kullanıcılara büyük kolaylıklar sağlamaktadır.

Bu bilgiler ışığında özellikle bebeklerin aşı zamanlarının takibi için çok önemli olan ihtiyacın mobil yazılım aracılığıyla veli ve doktor tarafından kullanılabilmesi önem arz etmektedir. Bununla beraber gebe aşıları hakkında fazla bilgiye ulaşamayan kişiler için gebe aşısı takvimi de bu yazılım sayesinde takip edilebilmektedir.

1.3. Literatürdeki Yeri

Mobil yazılım alanında Türkiye'de ve dünyada yapılan çalışmalara bakıldığında bilgilendirme [7], oyun[6], mobil öğrenme[21] [27], konum bilgisi [11], yabancı dil öğrenme [46], araç tanıma[34] ve araç takip[29], sıcaklık kontrolü [47] amaçlı yazılımların yanı sıra, teknik bilgi gerektiren işlemlerde, görüntü işleme çalışmaları [48], kaza tesbiti [49] veya hasta-doktor [24], [38] ilişkisini kolaylaştırıcı, mobil astım takibi [39] ve çok farklı alanlarda [10][18] yazılımlar hazırlanmış ve hazırlanmaktadır. Aşağıda mobil sistemler üzerinde geliştirilmiş bazı sağlık uygulamaları ile ilgili çalışmalar verilmiştir.

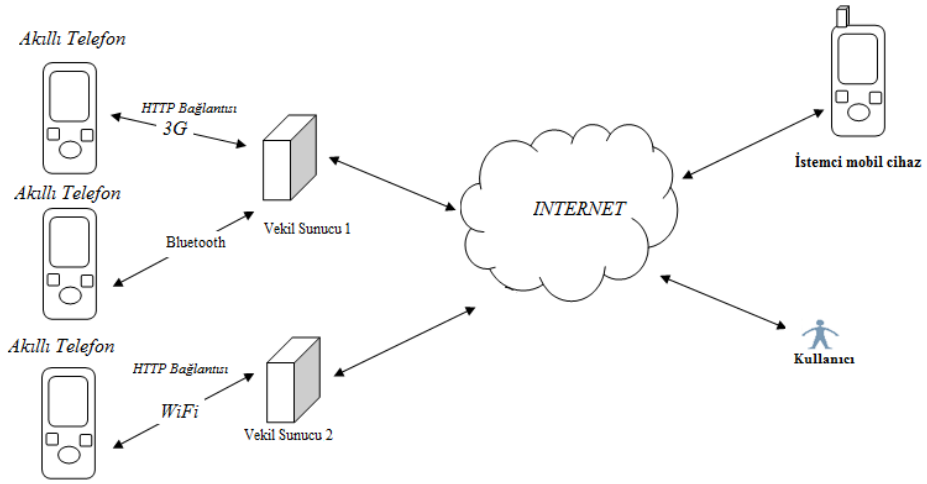
1.3.1. Tele-Sağlık için Mobil Servis Platformu

Tele tıp hastaların iyileştirilmesi amacı ile tıbbi bilginin [16] elektronik ortamda değişimini gerektirir. Bu bilgilerin içerisinde tele tıp ve tele sağlık için hazırlanmış konferans görüntüleri, e-sağlık hasta portalları, vital (yaşamsal) bulguların uzaktan izlenmesi ve sürekli tıp eğitimi vs. konular yer almaktadır.[15]

Normal tedavi usulleri düzenli ve uzun süreli tedavi gerektiren hastalarda, maliyet açısından ekstra bir yük getirir. Uzak ve kırsal alanlarda yaşayan hastalar için bu daha da sıkıntılı bir yoldur. Bunun yanında spor hekimliği ve koçluk gibi uzaktan izleme gerektiren

sistemler için de Tele tıp en uygun çözüm noktasını oluşturmaktadır. Senaryo özellikleri olarak kalp hastalarının izlenmesi(kalp hızı, duruş, nefes alma oranı), diyabet takibinin yapılması(kan şekeri seviyesi), atlet eğitimi(deri sıcaklığı solunum, kalp hızı, dehidratasyon, sporcunun konumu, yol, arazi ve yükseklik belirlenmesi) gibi durumlarda çalışabilecek bir sistem tasarlanmıştır. Ayrıca aşağıda verilen özelliklere sahip bir sistem olması gerekmektedir. [38]

a) **Bağlantı**: Taraflar arasında devamlı üst düzey bir bağlantı gerektirmesi.
b) **Gerçek zamanlı etkileşim**: Gerçek zamanlı bilgi alışverişinin sağlanması. Koç'un sporcusunu tam olarak yönlendirebilmesi için gerçek zamanlı veri taşınması gereklidir.
c) **Çift yönlü iletişim**: Hastanın cihazının veya örnekleme frekansının, doktorun verileri takip edebilmesi için doktorun isteklerine göre ayarlanması, bunun içinde çift yönlü iletişim hattının olması.
d) **Bilgi değiştirme**: Yaşamsal sinyallerin devamlı olarak servis talepleri ile güncellenmesi.
e) **İşleme**: Yoğun algoritmaların analiz ve potansiyel olarak yaptığı işlemlerin hesaplamalarını yapması.
f) **Güvenlik**: Alınıp gönderilen verilerin, yalnız hastalara gösterilmesi. Başkalarının bu bilgilere ulaşamaması.



Şekil 1.1. Odin platformu'nun genel mimarisi

Sistem internet üzerinde vekil sunuculara yüklenen veritabanlarına bağlanabilecek istemci sistemlerin yapılarında ve bağlantı şekillerinde kullanıcılara esneklik sunmaktadır.

Hem mobil hem de sabit kullanıcılar, 3G, Bluetooth ve WiFi üzerinden sisteme bağlantı kurabilmektedir. Odin mobil hizmetlerin geliştirilmesi ile uygulama geliştiricilere yardımcı olmak için zengin bir API sağlar.

Uygulama bir akıllı telefon ve kalp hızı, solunum hızı, durgun cilt ısısı ölçümlerini alabilen bir bluetooth vücut sensörü içermektedir.

Oluşturulan sistem pil kullanımı, bant genişliği tüketimi ve veri iletim gecikmesi gibi başlıklarda bazı ölçümlere tabi tutularak incelenmiştir. Deneyler şekil 1.3.'de görülen Android 2.1 işletim sistemi çalıştıran bir HTC akıllı telefon kullanılarak yapıldı. Windows Server 2008 işletim sistemine, 4 GB Ram, Intel Xeon X5570 2.96 Ghz işlemci ve 54 Mbps 802.11g kablosuz ağ ve 3G özelliklerine sahip bir sistem vekil host olarak çalıştırılmıştır.

Tablo 1.3. Akıllı telefon işletim koşullari

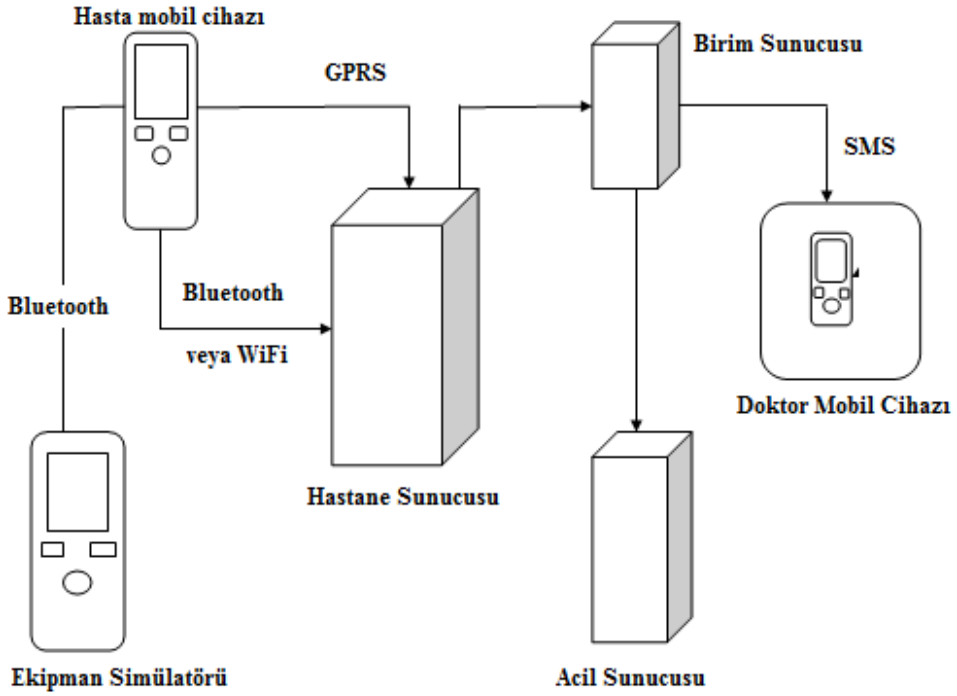
Akıllı Telefon Çalışma Koşulları	İşlem süresi
Servis konuşlandırılmadı	13 saat 55 dk.
Servis boşta Vekil Sunucuya WiFi Bağlantısı	9 saat 35 dk.
Servis boşta Vekil Sunucuya 3G Bağlantısı	9 saat 5 dk.
WiFi üzerinde çalışan WiFi	6 saat 26 dk.
3G üzerinde çalışan WiFi	6 saat 1 dk.

Sağlık alanında süregelen değişimler göz önüne alındığında tele tıp çözümleri ilgi artan düzeyde artıyor. Bu uygulamada önemli vücut sinyallerini izleme sensörü ile mobil izleme uygulaması geliştirildi. Kullanıcı etkileşimini çift yönlü kolaylaştırıcı olduğu gözlenmiş bir çalışmadır. Bununla tele-sağlık çözümleri noktasında birçok problem çözülebilir.

1.3.2. Uzaktan Güvenli Hasta İzleme Sistemi

Düşük maliyetli taşınabilir sistemlerin tasarımı kronik hastalıkların uzaktan izlenmesi için önemli alanlardandır. Bu çalışmada[24] düşük maliyetli ve güvenli bir taşınabilir bir sistem sunulmaktadır. Bu sistemde, kalp hızı, O2 seviyesi gibi hayati parametrelerin verileri arşivlenir ve bir cep telefonu üzerinde ve merkezi bir sunucu üzerinde tutulmaya ve ihtiyaç anında gönderilmeye çalışılır.

Mobil tele tıp sistemleri uzak hastane ve hastaları uzaktan izleme için giderek büyüyor. Bu sistemler düşük maliyet, düşük güç tüketimi ve küçük/hafif cihazlara gömülü kullanıcı dostu ara yüze sahip sistemler olmalıdır. Bu sistemler Bluetooth, GPRS, GSM veya Wİ-Fi gibi teknolojiler içeren hastane veya acil merkezlere kablosuz iletim sağlar. Bugüne kadar kazanılmış verileri örneklemek, kısa bir süre için saklamak ve istenen hedefe iletmek için bir çok tele tıp sistemleri mevcuttur. Bu sistemlerin çoğu tıbbi ekibi veya hastanın bir yakınını ciddi durumlar için uyarma yeteneğinden yoksundur. Örneğin EKG'yi sadece uzaktan izleme için bazı örnekler mevcuttur. Sinyali alıp örnekleme için bir veritabanına kaydeder. Bu çalışmada şekil 1.4.'de görüldüğü gibi, mobil olarak bir dizi sinyali toplayan bir modül, ayrıca kablosuz iletim özelliklerini içinde barındıran (Bluetooth, Wi-Fi, GSM/GPRS) güvenli veri gönderme aracı da mevcuttur. Bir sunucu verileri arşivlemek için kullanılır. Acil durumlarda da veri sunucusuna dayalı olarak çalışan bir acil sunucu, hasta için uygun eylemin ne olduğunu bildirmek için kullanılır.



Şekil 1.2. Hasta izleme sistemi mimarisi

Bu sistem 5 modülden oluşuyor.

1- Veri Toplama: Kan basıncı, kalp hızı, vücut ısısı gibi hayati verileri el ile veya Bluetooth ile cep telefonuna aktarabilir.

2- Modül Gönderme: Cep telefonu (terminal) üzerinde çalışan Bluetooth, Wi-Fi ile donatılmış Java tabanlı sanal bir makine. Bu adımda veri güvenliğini sağlamak için dahili bir kodlama işlemi vardır.

3- Modülü Alma: Hastane Sunucu Bluetooth, Wi-Fi, GPRS üzerinden cep telefonundan verileri güvenli bir şekilde verileri alma gerçekleştirilir. Bu adımda da kod çözme işlemi vardır.

4- Arşivleme Modülü: Hastanede iç ağdaki departman sunucusu alınan bilgileri hastane veritabanına kaydeder. Hastanın durumuna göre acil sunucuya ve doktorun cep telefonuna da SMS olarak bilgileri gönderir.

5- Uyarma Modülü: Acil durumlarda sunucu, kurtarma ekibini uyarır.

Önerilen sistemin üç alanda entegrasyonu vardır. Mobil iletişim, PC uygulamaları ve güvenlik yazılımı. Geliştirme Ortamı olarak J2ME, Netbeans-6.7.1 versiyonu kullanıldı. Visual studio 2008 PC uygulaması geliştirmek ve test etmek için kullanıldı. SQL Server 2005 veritabanı yapısı oluşturmak için kullanıldı.J2ME dili tarafından BouncyCastel

Kütüphanesi kullanıldı. Sony Ericsson W810i modeli cep telefonu mobil uygulama çalıştırmak için kullanıldı RFCOMM protokole Bluetooth iletişim kullanılarak programlandı (JSR 82). RFCOMM güvenilir ve iki yönlü bağlantı sağladığından Bluetooth teknolojisi etkin olan herhangi bir tıbbi cihaza bağlanmak için kullanılmaktadır.

Güvenlik Yazılımı iki bölüme ayrılmıştır; Bilgiyi kodlayarak gönderen (hasta/hemşire mobil uygulama) taraf ile kodu çözme tarafı. Hasta verilerini, gizli bir anahtar ile SHA1 algoritması ile şifreler. Gönderenin MAC değeri ile birleştirilip karşı tarafa ulaştırılır. Bu gizli anahtar tekrarlanmaz çünkü üç faktöre bağlıdır. Hastanın ID'si, adı ve doğum tarihinin birleşiminden oluşur. Hastane sunucusuna hastanın hayati sinyallerinin aktarılması başarıldı. Bluetooth özellikli PC alınan sinyalleri hasta bilgi girişi olarak alıyor, güvenli bir şekilde veritabanına kaydediyor. Acil durumlarda bilgiler acil sunucuya aktarıldı. Her yerde her zaman izleme sistemi kullanılabilmesi için cihazların yaygın olarak kullanılması gerekir. Bu izleme sistemleri temel olarak cep telefonlarının kullanılmasını tavsiye eder. Bu çalışmada düşük maliyetle, güvenlikli bir sistemle hayati verileri toplama ve görselleştirme gerçekleştirilmiştir.

Android platformunda yapılmış uygulamalara bakıldığında; alışveriş, eğitim, eğlence, sağlık, finans, oyun, video ve ses uygulamaları gibi çok farklı amaçlar için hazırlanmış örneklere rastlamak mümkündür. Burada özellikle sağlık alanındaki uygulamalar dikkate alındığında; spor, günlük egzersiz programlarından, kalori takibi, şeker takibi gibi programlara kadar farklı yapısal özelliklere sahip programlar geliştirilmiş ve geliştirilmektedir. Bunlara benzer olarak, Sağlık Bakanlığının Android Randevu Sistemi (MHRS) Mobil ile tüm Türkiye'de Sağlık Bakanlığına bağlı Devlet Hastaneleri, Ağız ve Diş Sağlığı Hastaneleri ve Merkezleri (ADSH, ADSM) için vatandaşların, istedikleri hekimden ve belirledikleri uygun olan tarihe randevu almaları sağlanır. MHRS Mobil uygulaması yardımı ile kolay ve hızlıca muayene randevusu alınabilir ve alınan randevular buradan takip edilebilmektedir. Randevu gününü, saatini ve ilgili hekimin o gün hastanede görev yapıp yapmadığını öğrenilebilir, gidilemeyecek randevular buradan iptal edilebilmektedir.

Medikal Klinik Asistan Programları

- Aile Hekimliği Bilgi Yönetim Sistemi(AHBS) Programları
- Hastane Bilgi Yönetim Sistemleri (HBYS)
- Hastalık, İlaç ve Teşhis Rehber Programları (Epocrates, DrDrugs, Tarascon vs.)

- Tıbbi Hesaplama Programları (Medcalc, Medmath, Archimedes, Pregcalcpro vs.) gibi programlar sağlık alanında geliştirilmiş programlardandır ve şu an için genellikle sabit sistemler olmak üzere farklı ihtiyaçlara cevap olarak hizmet vermektedirler. Fakat diğer taraftan mobil platforma taşınmasına da önem verilmektedir. Alınan kararlara destek oluşturması amacıyla üretilen uygulamalar tüm kullanıcılar açısından pozitif etkileri barındırmaktadır.[5]

2. ANDROİD TABANLI SİSTEMLER İÇİN YAZILIM GELİŞTİRME

2.1. Android İşletim Sistemi

Bireysel olarak kullanımı olan, kitap okuma, yabancı dil öğrenme, sosyal ağlardan kurumsal kullanıma ait programlara, banka işlemlerine, sağlık, adalet, ulaşım gibi farklı devlet kurumlarının programlarına kadar birçok alanda mobil yazılımlar bulunmaktadır.

Mobil yazılım alanında üzerinde en çok mobil yazılım üretilen Android işletim sistemi; Google önderliğinde, 5 Kasım 2007 tarihinde 34 şirket ile birlik olarak çalışmalarına başlanılan bir mobil işletim sistemi projesidir. Bu birliğin adı Open Handset Alliance'dır ve web sitesi de http://www.openhandsetalliance.com adresidir. Google, Android Inc adlı şirketi satın aldı ve Open Handset Alliance (OHA) birliğini kurdu. Open Handset Alliance (OHA), mobil teknolojiler sektöründe büyük önem taşıyan oyuncuların Google önderliğinde bir araya gelmesiyle kurulmuş bir birliktir. Bu birlikte yarı iletken teknolojisi kuruluşları Android'in gelişimi OHA eliyle yönetilmeye başlandı. Gsm operatörleri, cep telefonu üreticileri ve ticarileştirme kuruluşları bulunuyor.

Üye şirketler birbirlerinden teknoloji transferlerinde bulunmakta ve üretilen sistemlerin açık olmasına çalışmaktadırlar. Android sistemi başlangıçta Open Handset Alliance sitesi üzerinden sunulurken şimdi www.android.com adresinden sunulmaktadır. Open Handset Alliance'in büyük üyelerinden bazıları Samsung, Motorola, LG, HTC, SonyEricsson, Acer, Vodafone, Alcatell, Dell, Lenovo, Huawei, Sharp, Nec, Toshiba, Ebay, Asus, Fujitsu, Kyocera, Arm, Atheros, Broadcom, Intel, Marvell, Mips, Texas Instruments, Andago, Access şirketleridir.

Android, inşası için linux çekirdeği kullanılmış bir mobil işletim sistemidir, bu sistemde ara katman yazılımı, kütüphaneler ve API C diliyle yazılmıştır. Uygulama yazılımları ise, Apache harmony üzerine kurulu java-uyumlu kütüphaneler ihtiva eden uygulama iskeleti üzerinden çalışır. Derlenmiş java kodunu çalıştırmak için dinamik çevirmeli (JIT) dalvik sanal makinasını kullanır, cihazların kabiliyetlerini artıran uygulamaların geliştirilmesi için çalışan geniş bir programcı-geliştirici çevresine sahiptir. Android farklı çok sayıda donanım, sensör, proje, uygulama, geliştirme ortamı ve teknolojileri içeren büyük bir ekolojik yapıyı ieçrisinde barındırmaktadır. Android üzerinde çalışan uygulamaların sayısı her geçen gün daha da artmaktadır.

Android işletim sistemi beş kısımdan oluşur.

1. **Çekirdek:** Linux kernelidir. Güvenlik, hafıza yönetimi, süreç yönetimi, ağ yığınları ve sürücü modellerini içerir.
2. **Android Runtime:** Sanal makinedir. Dalvik Sanal Makinesini de içerir.
3. **Kütüphaneler:** Veritabanı kütüphaneleri, web tarayıcı kütüphaneleri, grafik ve arayüz kütüphanelerini içerir.
4. **Uygulama Çatısı:** Uygulama geliştiricilere geniş bir platform sunan kısımdır.
5. **Uygulama Katmanı:** Doğrudan Java programlama diliyle geliştirilmiş uygulamaları içerir.

Android mimari olarak Linux çekirdeği üzerine Android runtime ve kütüphanelerin bulunduğu bir yapı ile beraber bunların üzerinde çalıştırılan uygulama çatısı ve uygulamalardan oluşan bir yapıya sahiptir.

Android' in tarihçesine bakıldığında Google'ın 2005 yılında Android İnc adlı şirketi satın alması ve Open Handset Alliance' nin kurulması Android için iki önemli adımdır.

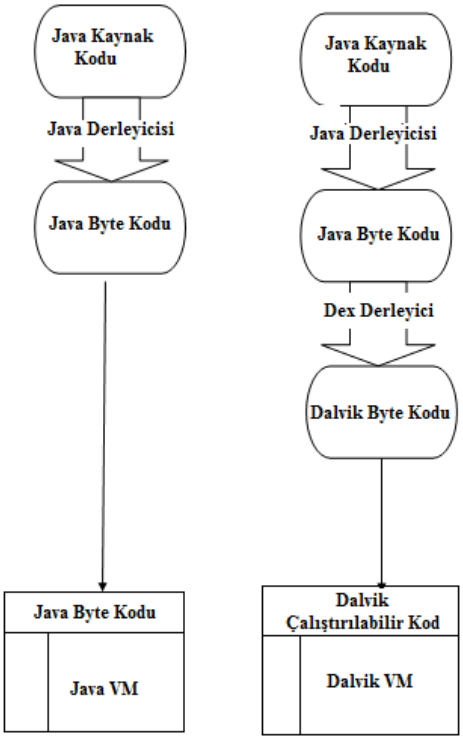
Open Handset Alliance, mobil teknolojiler sektöründe büyük önem taşıyan oyuncuların Google önderliğinde bir araya gelmesiyle kurulmuş bir birliktir. 5 Kasım 2007'de 34 üye kuruluş ile duyurulan bu birliğin üye sayısı sürekli artmaktadır. Bu birlikte yarı iletken teknolojisi kuruluşları, yatılım kuruluşları, GSM operatör cep telefonu üreticileri ve ticarileştirme kuruluşları bulunmaktadır. Üye şirketler birbirleriyle teknoloji transferinde bulunmakta ve üretilen sistemlerin açık olmasına çalışmaktadırlar. Android sistemi başlangıçta Open Handset Alliance sitesi üzerinden sunulurken şimdi http://www.android.com/ adresinden sunulmaktadır. Open Handset Alliance' nin büyük üyelerinden bazıları Samsung, Motorola, LG, HTC, Sony Ericsson, Acer, Vodafone, Alcatell, Dell, Lenovo, Huawei, Sharp, Nec, Toshiba, Ebay sayılabilir.

Android açık kaynak kodludur ve açık kaynak lisansı kullanan çok sayıda yazılımı içinde barındırmaktadır. Zaten linux çekirdeğini ihtiva ettiği için doğal olarak açık kaynak kodludur. Bir yazılım, açık kodlu yazılımlar kullanıyorsa kodunu açmak durumundadır. Ama bu yazılım üzerinde çalışan her yazılımın açık kaynak kodlu olması zorunlu değildir. Örnek vermek gerekirse Android Market uygulaması açık kodlu değildir. Google'in Android üzerinde çalışan tüm yazılımları da kapalı kodlu değildir. Android İşletim Sistemi internetten indirilip cihaza veya sanal makineye kurulursa içerisinde pekçok önemli yazılımın olmadığı görülecektir. Android Market, Google Maps, Youtube, Gmail uygulaması gibi. Google bu uygulamaları telefon üreticilerine satmaktadır.[32] Bir mobil cihaz üretici

şirket hiç kimseye danışma veya sorma gereği duymadan Android tabanlı cihaz üretip satabilir fakat bu cihaza yüklenecek yazılımlar için Google ile anlaşmadıysa cihazda bu yazılımları bulundurmayacaktır. Kaynak kodları ile ilgili bilgi ve indirme için http://source.android.com/source/index.html adresi kullanılabilmektedir.

Android içerisinde Linux çekirdeğinin bulunma sebebi çok sayıda donanım desteğini sağlaması ve Google tarafından yoğun olarak kullanılmasından kaynaklanıyor. Linux topluluğunun android'in kazandığı başarının arkasında desteği vardır.

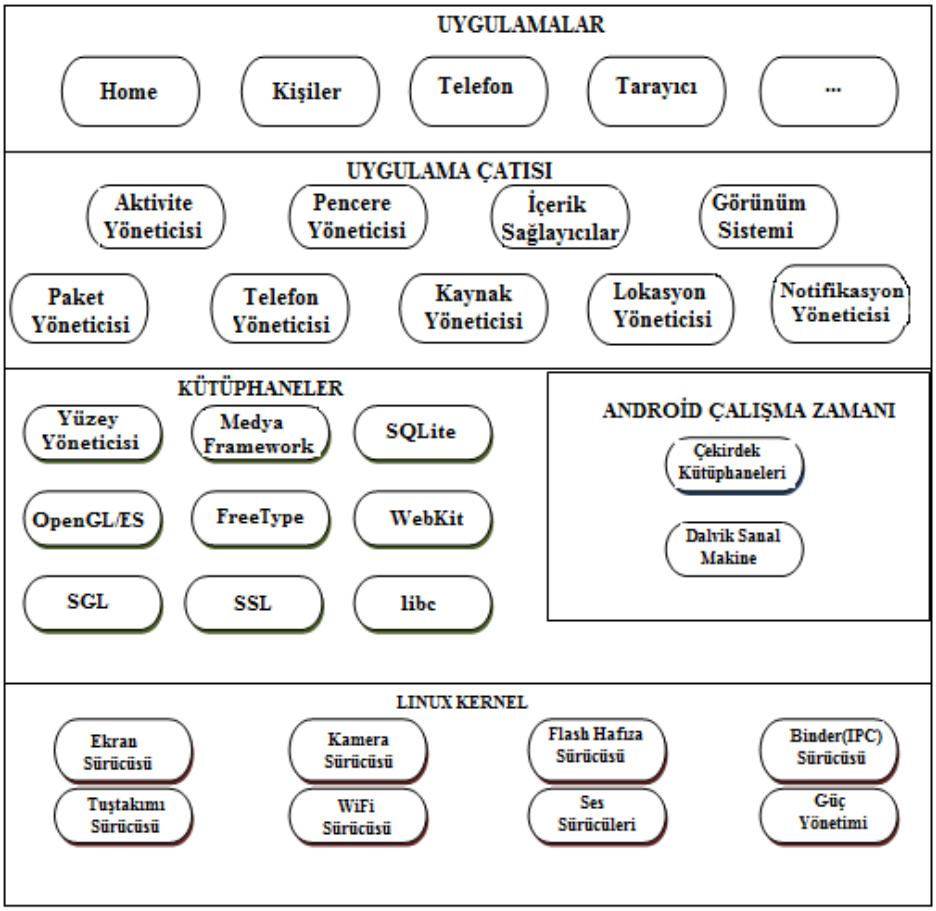
Java 1995 yılında Sun Microsystems tarafından duyurulan bir programlama dili ve yazılım altyapısıdır.Java'nın diğer dillerden ayrıldığı önemli bir özellik sanal makine (virtual machine) kavramını başarılı şekilde uygulamasıdır. Çünkü java programlama dili ile yazılan uygulamalar doğrudan işletim sistemi üzerinde çalışmazlar. İşletim sistemi üzerinde bulunan bir sanal makine tarafından çalıştırılırlar. Google'ın geliştirdiği Dalvik adlı sanal makine ise Sun'ın geliştirdiği sanal makineden daha performanslıdır. Burada Android runtime birimi içerisindeki DVM'nin sistem içerisinde önemli bir yeri bulunmaktadır. Android uygulamalarını compile ederken Android için özel olarak yazılmış Dalvik Virtual Machine kullanılır. Uygulamalar genellikle Java ile geliştirildiğinden, şekil 2.11'de görüldüğü gibi kodlar önce JVM (Java Virtual Machine) ile bytecode'a compile edilir [41]. Bu işlem sonrası JVM ile uyumlu hale gelen .class dosyaları uygulama çalıştırılmaya başlamadan önce bir de Android'in anlayabileceği ve çalıştırabileceği Dalvik compiled executable file olan .dex formatına çevrilir.



Şekil 2.1. DVM ile JVM'nin çalışma biçimleri

Google, Java ve Python dilini sıkça kullanmaktadır. Android projesinde ise yazılım geliştirme dili olarak yazılım dünyasında en çok kullanılan ve en çok yazılımcıya sahip olan Java dilini tercih etmiştir. Java dili ile mobil cihazlar için yazılım geliştirme daha önceden de yapılmaktaydı[8], önce J2ME daha sonra Java ME ile yapılan uygulamalar, Android'in gelişimiyle beraber çoğunlukla Android'de geliştirilmeye başlandı. Android mimarisinde şekil 2.12'de görüldüğü gibi en alt kısımda Linux çekirdeği, onun üzerinde Linux'un da uzun suredir kullandığı çeşitli kütüphaneler ve Android çalışma zamanı, onun üzerinde uygulama çatısı ve en üstte uygulamalar bulunmaktadır[40]. En üstte yer alan uygulamalar uygulama çatısı sayesinde işlemleri gerçekleştiriyor.

Uygulama çatısının altındaki kütüphanelere ve Linux çekirdeğine doğrudan erişen uygulamalar da olabilir ve bu uygulamalar genelde C ve C++ dilleriyle yazılır.



Şekil 2.2. Android işletim sisteminin mimari yapisi

Android kullanan cihazlar şöyle gruplandırılabilir. Akıllı telefonlar, tablet bilgisayarlar, google tv ve samsung televizyonları, gps cihazları, netbooklar, ortam oynatıcıları, oyun konsolları, kablosuz ev telefonları, e-kitap okuyucuları, beyaz eşyalar.

Eskiden sadece telefon etmek, sms göndermek için kullanılan telefonlar 90'ların sonları itibariyle cep bilgisayarı konseptine geçiş yapmaya başladı. PDA ya da Palm denilen cihazlarda kurumsal uygulamalar kullanılmaya başlandı. Zamanla fotoğraf makinesi, GPS, media özellikleri WiFi gibi özellikler kazandırılmaya başlandı. Cep bilgisayarlarının küçülmesi ve telefonlara ait özellikleri içinde bulundurmaya hazır hale getirilmesiyle akıllı telefonlar konsepti ortaya çıkmış oldu.

Android kullanarak sıradan bir telefon üretebileceğiniz gibi çok sayıda donanıma ve sensöre destek veren çekirdeğini kullanarak aşağıdaki donanımlara ve sensörlere sahip cihazlar da üretilebilir. Dokunmatik ve çoklu dokunmatik (multitouch) ekranlar, sanal veya gerçek klavyeli ekranlar, GPS, çok sayıda kamera, pusula, metal dedektörü, ısı dedektörü,

ışık dedektörü, jiroskop, ivmeölçer, bluetooth, yakınlık sensörü, wifi,2g (gprs, edge) ,3g,4g. Android cihazlarda yazılımsal yetenekler de son derece gelişmiştir.

HTML 5 destekli internet tarayıcıları (Browser, Firefox, Skyfire) , Gmail, Google Maps gibi önemli Google servisleriyle senkronize çalışabilmesi, ana ekranda çok sayıda masaüstü desteklenmesi, widget adı verilen küçük yazılımsal eklentilerin masaüstüne eklenebilmesi, çok sayıda başlatıcı (launcher) ile farklı görünümler kazanabilmesi ve özelleşebilmesi, uygulamaların hangi yetkilerle çalışacağının daha uygulama kurulurken kullanıcıya sorması ve bilgi vermesi gibi özellikler sayılabilir.

2.2. Android Yazilim Geliştirme Platformu

Android çok sayıda güncel yazılım dili ve yazılım teknolojisine destek verir, Fakat Java dili ve Android' in kendi yazılım altyapısı ön planda görünmektedir. Android platformunda geliştirilen uygulamalar boyut olarak küçük olduğundan internetten kolayca indirilebilmekte ve mobil cihazlara yüklenebilmektedir. Önce Google tarafından Android Market adıyla açılan bu uygulama mağazaları sayesinde yazılımcılar programlarını çok geniş bir pazara sunabilmekte ve gelir elde edebilmektedirler.

Android platformu bir işletim sistemidir ve bu platform üzerinde çeşitli programlama dilleri ile yazılımlar geliştirilebilir. Google tarafından sunulan developer.androld. com sitesinde Android için ağırlıklı olarak Java teknolojisi ile yazılım geliştirme imkanları sunulmuştur.Google Android platformu için Java dilini zorunlu tutmamıştır. http://developer.android.com/tools/sdk/ndk/index.html sayfasında C Ayrıca Microsoft'un. Net teknolojileri ile yazılım geliştirmeyi amaçlayan monodroid adında bir topluluk da bulunmakta.Python,perl,JRuby,Lua,BeanShell,JavaScript,Tcl dillerine de destek verilmektedir. http://code.google.com/p/android-scripting/ adresinden bu projeye de ulaşılabilir.

Java ile applet ve uygulamaları yazmak için JDK gibi geliştirme araçları gereklidir. JDK'nın içerisinde Java Runtime Environment, Java derleyicisi ve Java API'leri vardır. Android SDK kurulumuna geçmeden önce JDK kurulması gereklidir. Bunun için gerekli olan versiyonu http://www.oracle.com/technetwork/java/javase/downloads/index.html adresinden temin edilebilir. Java programlamada kullanılan yapı incelendiğinde, kodların herhangi bir metin editörüyle yazılması gereklidir. Yazılan kodlar .java uzantılı olarak kaydedilir. Bu .java uzantılı kodlar java derleyicisi ile derlenir.Kodların derlenmesinin

ardından aynı isimde .class uzantılı bytecode olarak isimlendirilen bir makine kodu oluşturulur. Her ne kadar bytecode makine kodu olarak ifade edilse de işlemcilerin anlayabileceği bir dile sahip değildir. Bu sebeple kodların tekrar derlemesi gerekir ki, bu aşamada duyulan ihtiyacı JVM adı verilen sanal makine karşılar.

Her işletim sisteminin kendisi için hazırlanmış bir sanal makinesi vardır. Bu JVM'ler aynı java kodlarını çalşıtırıp farkl platformlarda aynı kodlaırn çalışabilmesini temin ederler. Java tarafından iddia edilen platform bağımsız bir yapıya sahip olabilmesi bahsedilen Java Sanal makinesi sayesinde gerçekleştirilmiş olur. Bu bakımdan java platform bağımsızlığını JVM'ye bağlı olarak yürütmektedir.

Bireysel olarak java uygulaması oluşturmak istenildiği takdirde, JDK'ya adı verilen yapıya ihtiyaç duyulmaktadır. JDK (Java Development Kit)'nın içerisinde Java compiler, Java interpreter, geliştrici araçları, javaya ait API kütühanelerei, dökümantasyonlar, JVM ve JRE gibi bileşenler barındırlmaktadır. JDK 1,7 ile birlikte Oracle bytecode üzerinde değişiklikler yapmıştır. Dalvik VM olarak adlandırılan Android için hazırlanmış olan sanal makine henüz bu yeni bytecode yapısına uyumlu değildir. İşletim sistemlerine uygun olan 1.6 versiyonu listeden indirilebilmektedir. Kurulumu tamamladıktan sonra JAVA_HOME ortam değişkenini oluşturup JDK'nın kurulduğu adresi değer olarak verilmesi gerekmektedir.

Java ile programlama yapılırken diğer dillerde olduğu gibi gömülü fonksiyonları, tanımlanmış prosedür ve metotların kullanımını daha rahat bir şekilde gerçekleştirebilmek için kullanılan editörlerden faydalanılabilir. Bütün kodların tamamen elle yazılması yerine cümlelerin tamamlandığı fonksiyonların hazır olarak beklediği editörler yazılımcılar için kolaylıklar sağlamaktadır. Projenin büyüklüğü de göz önüne alındığında hazır bazı tool'lar çok gerek sözdizimi hatalarında, gerek hata tespitlerinde önemli görevleri üstlenmektedirler. Sadece işlevsel değil estetik olarak da yazılımı okumayı kolaylaştıran özelliklere sahip olabilirler.

Java ile yazılım geliştirme için kullanılan Eclipse, Netbeans, JCreator, Processing IDE, JBuilder, Intellij IDEA gibi farklı IDE(Integrated Development Environment)'ler bulunmaktadır. Eclipse ve Netbeans adlı editörler bu alanda çok tercih edilmektedirler. Özellikle Eclipse editörü kullandığı plug-in(eklenti) desteği sayesinde diğer editörlere gore avantaj sağlamaktadır. [28]

Geliştirme ortamı olarak Eclipse kullanmak zorunluluğu bulunmamaktadır. Java ile programlama yaparken farklı bir editörde kullanılabilir fakat Google Eclipse'i

önermektedir. Ayrıca uygulama geliştirmek için birçok eklenti sunduğundan dolayı google tarafında Eclipse üzerinde çalışılmış ve dökümantasyonlar Eclipse eksenli oluşturulmuştur. Eclipse kullanıcılarına açık kaynak kodlu bir geliştirme ortamı sunmaktadır. Daha çok java ile ilişkili görünmesinin yanında C ve Phyton dilleri için de kullanılmaktadır. IBM tarafından 2001 yılında geliştirilen Eclipse Android SDK ile bütünleşik çalışabilmesinin yanında eklentileri ile birçok farklı alanda kullanılabilmektedir.

Eclipse ile uygulama geliştirirken kullanılan eklentilerle beraber birçok menünün de editöre eklenmesi sayesinde kullanıcılar menüler içinde gezinmekte zorlanabilmektedir. Bu zorlukların önüne geçilebilmesi adına tablo 2.1.'de sık kullanılan kısayollar sunulmuştur.[36]

Tablo 2.1. Eclipse kısayolları

CTRL + Space:	Bir kısmı yazılan bir kod parçasını tamamlayacak önerileri gösteren bir pencere açar.
CTRL + Shift + O:	Yazılan kod içerisinde kontrol edilip import edilmemiş bileşenler varsa ekleme işini otomatik olarak yapar. Eğer kullanılmayan importlar varsa kaldırır. Aynı isimde birden fazla import seçeneği varsa seçenekleri sunan bir pencere açar.
Tab / CTRL + Tab:	Tab, seçili kısmı ileri ötelerken CTRL + Tab geri öteler.
CTRL + Shift + I:	Kod içerisinde daha önce belirlenmiş olan girintileri, belirlendiği şekilde düzenler.
CTRL + Shift + F:	Girintiler, parantezler, satır fazlaları, kodun görünüşünü Preferences'dan ayarlandığı şekilde düzenler. Ancak bu kısayolun çalışabilmesi için kod içinde herhangi bir hata olmaması gerekmektedir.
CTRL + Sağ/Sol/Backspace/Del:	Normalde çoğu editörde olduğu gibi Eclipse'in editöründe de CTRL + Sağ/Sol kelimleler arasında hızlıca gezinmeyi sağlar. Yalnız bu kelimeler sadece boşluk ile bölünüldüğünde değil, iki kelimeden oluşan bir methoddaki ikinci kelimenin büyük harfi gibi değişikliklere dikkat ederek gezinmeyi sağlar. CTRL + Backspace/Del ile bu kelimelerdeki gezinmeler gibi silme işlemleri yapılabilir.

Alt + Shift + Sag/Sol/Yukarı/Aşağı:	İmlecin durduğu yerden itibaren kırlangıç parantezlerin arasını kolay seçmeyi sağlar.
CTRL + L:	İmleç kullanıcı tarafından verilen satıra gider.
CTRL + Alt + Yukarı/Aşağı:	İmlecin bulunduğu satırı, bir üst veya bir alt satıra direk kopyalamayı sağlar.
Alt + Yukarı/Aşağı:	İmlecin bulunduğu satırı aşağı veya yukarı taşımayı sağlar.
CTRL + D:	İmlecin bulunduğu satırı siler.

Tablo 2.1. Eclipse kısayolları devamı

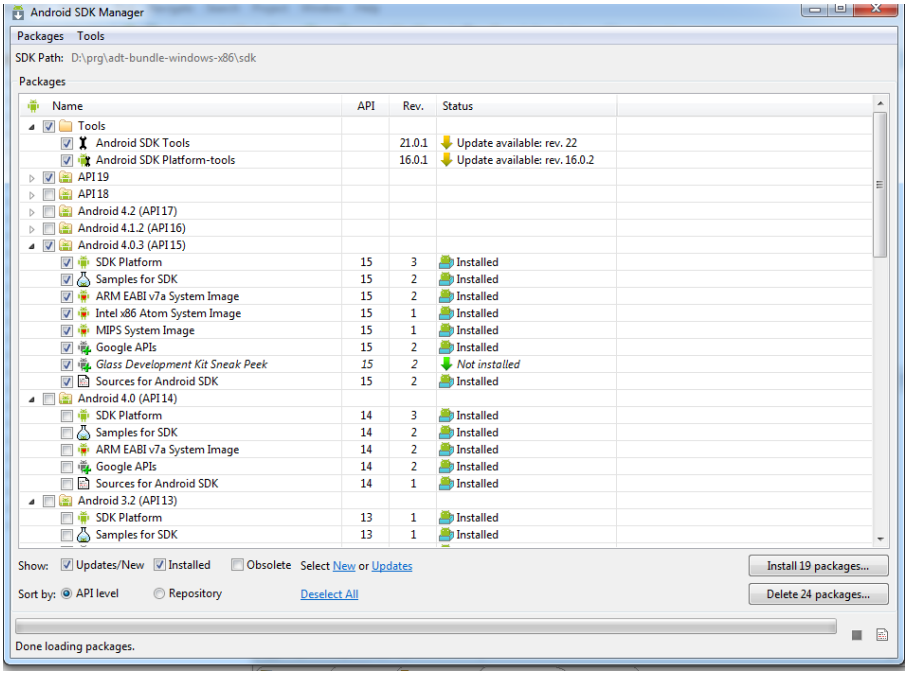
CTRL + Shift + Yukarı/Aşağı:	Kod içinde yazılmış methodlar arasında gezinmeyi sağlar.
CTRL + Shift + B:	İmlecin bulunduğu satıra breakpoint koyar veya kaldırır.
Alt + Shift + M:	Seçili kod parçasını yeni bir method oluşturup içine yazar.
Alt + Shift + J:	Methodun veya sınıfın üzerine Javadoc yorumu ekler.
CTRL + Shift + J:	Kodun seçili kısmını tek satır haline getirir.
F3:	İmlecin bulunduğu öğenin üzerindeyken basıldığında tanımlandığı kısma gider.
CTRL + 1:	Quick Fix penceresini açar.
CTRL + E:	Açık olan editörlerleri listeleyen bir pencere açar.
CTRL + Shift + L:	Bütün bu kısayolları bulunduğu ve kısayol tuşlarının değiştirilebildi bir pencere açar.

Eclipse http://www.eclipse.org/downloads/ adresinden Eclipse IDE for Java Developers kurulum versiyonu olarak indirilebilmektedir.

Android işletim sistemine sahip chazlar için yazılım üretme,test etme ve hata ayıklama gibi işlemleri gerçekleştirebilmek için içerisinde gerekli araçları ve API kütüphanlerei barındıran uygulama geliştirme aracı olan Android SDK'sına http:/developer.android.com/sdk/index.html üzerinden erişebilimektedir.Öncelikli olarak yapılması gereken JDK kurulumunun gerçekleştirilmesi ve IDE olarak Eclipse editörünün kurulumunun yapılmasının ardından Android SDK ve ADT(Android Development

Tools)'nin kurulması gereklidir.Android SDK adresinden işletim sistemine uygun versiyonu indirildikten sonra kuruluma geçilebilir. Zip dosyası uygun bir klasöre açılıp (Mümkünse doğrudan C veya D: sürücüsünün altında açılması tavsiye ediliyor. Örneğin;D:\android-sdk-windows) klasörün içindeki SDK Manager.exe dosyası çalıştırılmalıdır.

SDK kurulduktan sonra ekranda şekil 2.13.'deki pencere açılacaktır.



Şekil 2.3. Android SDK yöneticisi

Android SDK yöneticisini kullanarak SDK'nın son sürümünü Install düğmesi ile gereken paketlerin kurulumu tamamlanabilir. Kurulum tamamlandıktan sonra klasör içeriği tekrar incelendiğinde yeni elemanların eklendiği görülecektir. Eğer farklı android versiyonlarına hitap edecek uygulamalar üretilecekse istenilen versiyonlar buradan seçilip API'lerin ihtiyaca gore indirilmesi sağlanabilir. Bu pencerenin dışında Eclipse içerisinde iken Help -> Install New Software yolunu tıklayıp, Work with kısmına http://dl-ssl.google.com/android/eclipse/ linki yapıştırılarak, ardından Add düğmesine basılarak açılan pencereden Location sekmesine de aynı linki yapıştırarak aynı dosyaların indirilmesi sağlanabilir. Kaydedip Next'e tıklandıktan sonra yüklemelerin tamamlanması ile birlikte Eclipse ve Android'in bribirini tanıması gerçekleştirilmiştir.

SDK klasörü incelendiğinde farklı dizinler görülecektir. Androidin projelerinin takibi ve yapısının anlaşılması için klasör sisteminin de anlaşılması gereklidir. Bu klasörlerin içeriklerine bakıldığında platforms klasörü içerisinde kurulmuş olan farklı SDK

versiyonlarını barındırmaktadır. Her versiyon farklı bir klasör içerisine yerleştirilmiştir. Yeni SDK versiyonları yayınlandığında en baştan kurulum yapılmasına gerek yoktur. SDK Manager kullanılarak yukarıda gösterildiği gibi yeni versiyonu kurulabilmektedir. Kurulum tamamlandığında versiyon bu klasör içerisinde oluşturulacaktır.

Tools klasörü içinde emülatör, veritabanı yönetimi(sqlite), hata ayıklama(ddms) gibi komut satırından çalıştırılabilecek dosyaları barındırmaktadır. Samples klasörü içinde farklı SDK versiyonlarına göre gruplanmış durumda çalıştırılabilir örnekler bulabilirsiniz. Bu dosyalan genellikle komut satırından çalıştırılmıyor. Fakat ihtiyaç duyulan durumlarda her defasında aynı klasöre kadar gitmemek için tools klasörünün yolu (Örneğin; D:\android-sdk-windows\tools ve D:\android-sdk-wîndows\platform-toois) path ortam değişkenine eklenebilir. Böylece komut satırından doğrudan emulator yazarak emulatör başlatılabilir.

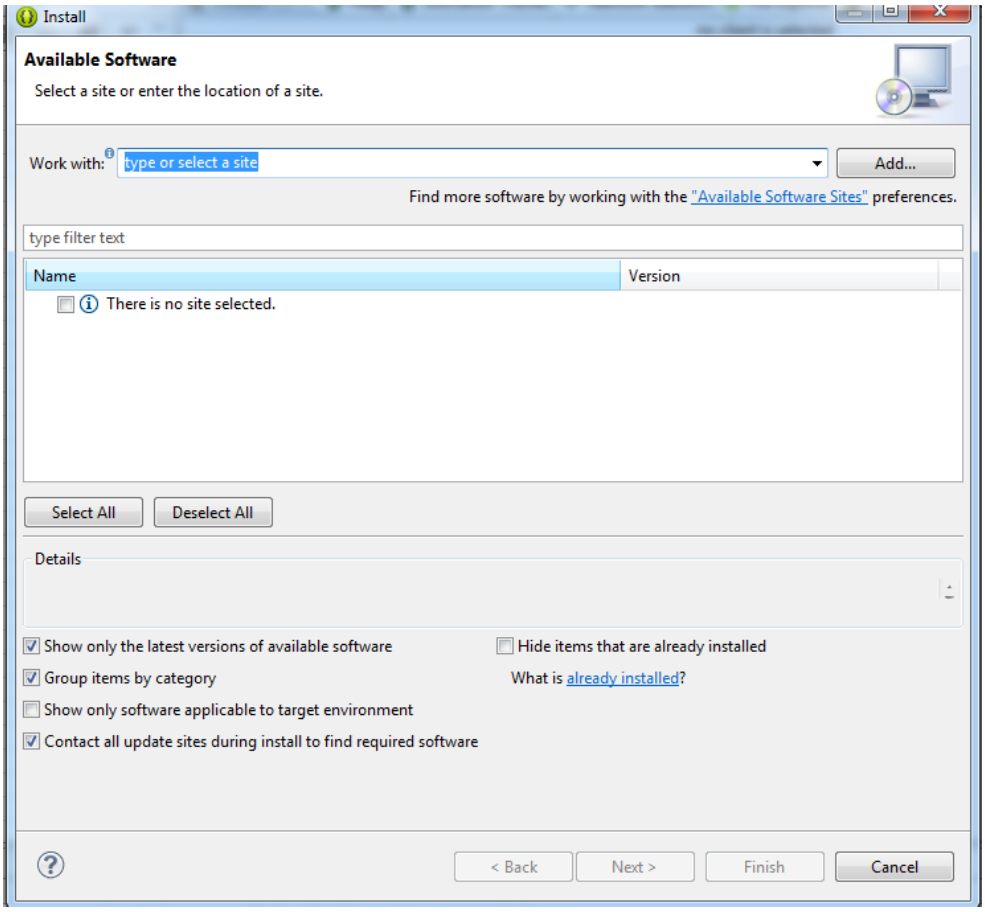
Samples klasörü içinde farklı SDK versiyonlarına göre gruplanmış durumda çalıştırılabilir örnekler bulunmaktadır. Bu örnekler nasıl uygulama geliştirileceği konusunda yazılım geliştirmek isteyenlere çok faydalı olmaktadır. Bunun dışında http:/developer.android.com adresi de yardımcı olacak bilgileri içinde bulundurmaktadır. Temel seviyede İngilizce bilgisi ile yeterli bilgiyi bu site sağlamaktadır. Ayrıca http://developer.android.com/resources/index.html içerisinde Samples klasörünüzdeki bazı uygulamalar hakkında da kısa açıklamalar ve ekran görüntüleri bulunmaktadır.

Google, Android uygulamaları için Android Development Toos (Android Geliştirme Araçları) adında bir eklenti geliştirmiştir. Bu eklenti sayesinde arayüz elemanları sürükle bırak şeklinde oluşturabilir, uygulama konfigürasyonu XML dokümanıyla uğraşmaksızın kolayca değiştirebilmektedir. Ayrıca ADT'ye uygulamayı debug etmek için yardımcı olacak özellikler eklenmiştir. ADT kurmak için önce Eclipse uygulamasının başlatılması gerekmektedir. Başlattıktan sonra Help menüsünden Install New Software adımı seçilmelidir. Karşımıza aşağıdaki şekil 2.14'te görüldüğü gibi bir pencere gelecektir.

Eklenti http:/dl-ssl.google.com/android/eclipse/ adresindedir. Eğer https üzerinden erişilemiyorsa, http olarak adresi değiştirilebilir. Developer Tools seçeneği işaretlenip Next butonu yardımıyla bir sonraki sayfaya geçilir ve açılan pencerede çıkan listeden bütün elemanlar seçilerek eklenti kurulabilir.

Eklentiyi kurduktan sonra Eclipse yeniden başlatılacaktır. Uygulama geliştirmeye başlamadan önce yapılması gereken son adımda ise; Eclipse'e SDK'nın yerinin

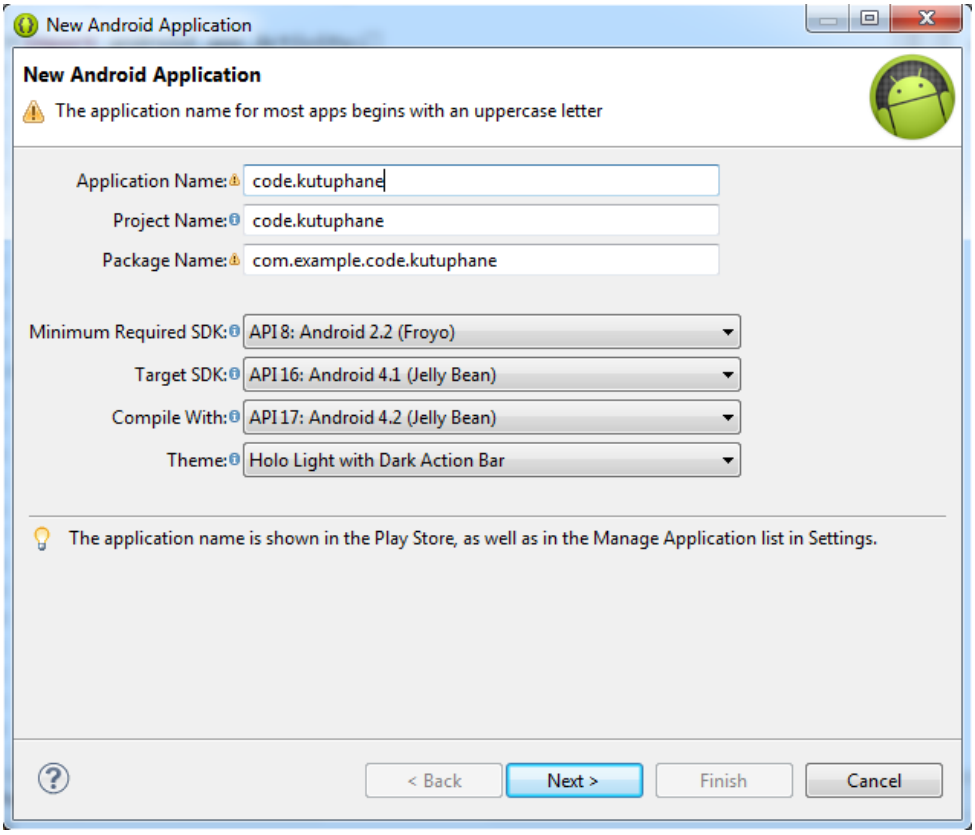
gösterilmesi gerekmektedir. Bunun için Window menüsünde Preferences adımı seçilir. Sol taraftaki Android menüsü seçilir. Sağdaki SDK Location bölümüne SDK'nın kurulduğu adres yazılır. Adresi seçildiğinde aşağıda kurulmuş olan SDK versiyonları listelenecektir. En sonuncusunu yani en yüksek



Şekil 2.4. Android eklenti indirme ekranı

seviyedeki API seviyesini seçip OK butonunu tıklayarak kurulum ayarları tamamlanır. Bu aşamadan sonra Android uygulaması geliştirmek için izlenmesi gereken yol şöyle olmalıdır.

Eclipse File > New > Android Application Project menusu seçilir. Açılan pencerede şekil 2.1.5'de görüldüğü gibi bir ekran açılacaktır



Şekil 2.5. Eclipse'de yeni android application oluşturma ekranı

Application Name olarak ekrana gelen seçeneğin içeriği, uygulamaların Google Play üzerinde ve dolayısıyla telefonlarda sahip olacağı isimdir.[26] Ayrıca telefonda veya emülatörlerde uygulamalar penceresi açıldığında uygulama logosuun altında görünecektir.

Project Name, projenin Eclipse workspace olarak kullanacağımız çalışma alanı üzerinde sahip olacağı isimdir. Application Name ile telefonlardaki uygulama ismi ile doğrudan bir bağlantısı bulunmamaktadır. İstenilen bir isim kullanılabilir.

Package Name, Java projelerinde klasik olarak belirlenen sınıflandırma amaçlı paket adıdır. Paket isimleri Java sınıflarını mantıksal gruplara ayırmak için kullanılır. Gruplandırma yaparken aralarda nokta kullanılır ve bu noktalar bir klasör hiyerarşisi içerisinde editör tarafından yorumlanarak proje yapısı oluşturulur.

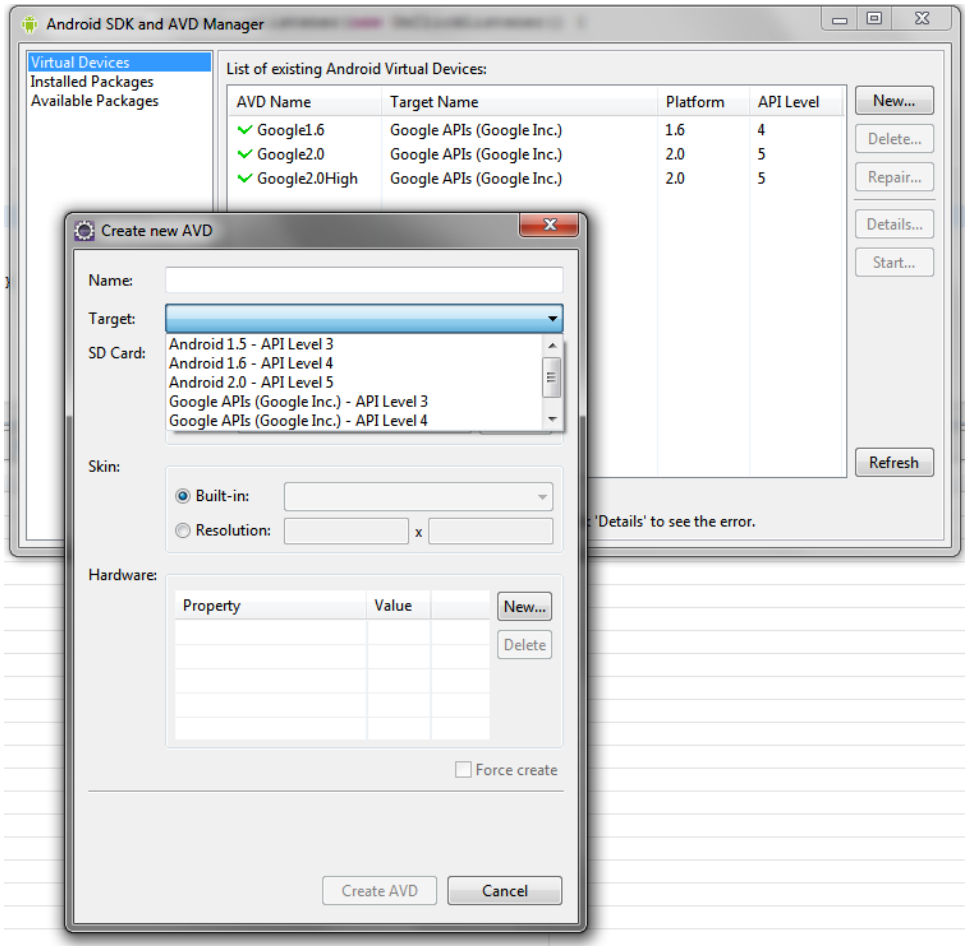
Build SDK, hangi Android SDK versiyonu üzerinde geliştirme yapılacağını belirlemeye yarar. Minimum Required SDK, uygulamanın çalışabileceği minimum SDK versiyonunu ifade eder. Bu geriye dönük uyumluluk açısından önemli bir konudur. Her yeni SDK yayınlandığında yeni uygulamaların, sizin bir yıl önce aldığınız telefonunuzda çalışmaması için bu konuya dikkat edilmesi gereklidir. Create Custom Launcher Icon proje için farklı bir simge kullanmak istediğinizde kullanılabilir. Burada aktivite oluştururken

boş bir ekran ile başlanılacaksa BlankActivity seçeneği seçilir. MasterDetailFlow, tablet bilgisayarlar için ekran oluşturulması içindir. BlankActivity seçilip, bir sonraki ekrana geçildiğinde karşımıza çıkan ekranda; Activity Name projede oluşturduğunuz ekranı temsil edecek olan sınıf için verilecek addır. Layout Name, ekran tasarımınızı içerecek olan xml dosyasına verilecek olan isimdir. Navigation Type, ekranın normal mi, sekmeli mi yoksa dropdovvn şeklinde mi olacağını belirtir. Title ekranın tepesinde görüntülenecek olan başlık değeridir.

Finish butonu tıklandığında Eclipse, Android projesini oluşturacaktır. Ekranın sol tarafında projede bulunan dosyalar görülebilir.

Varolan projeyi çalşıtırmak için bir emülatör yani sanal bir makine oluşturmamız gereklidir. Bu emulator Window>AVD Manager menüsü kullanılarak oluşturulabilir. Sağ taraftaki New butonuna basarak sanal makine oluşturulabilir. Name alanına sanal makine için uygun görülan bir isim girilebilir. Target bölümünde makinenin hangi Android SDK versiyonu üzerinde çalışacağı belirtilmeli. Build Target olarak belirlenmiş olan versiyon seçilir. SD Card, sanal makinenin ne kadar belleğe sahip olacağını, belirler. 9 MB'den aşağı olmamak şartıyla bir değer girilmesi gerekmektedir. Skin telefonunun ekran çözünürlüğünün belirlenmesini sağlar. Built-in çözünürlüklerden bir tanesi seçilebilir. Hardware alanı için herhangi bir değer girilmesine gerek yoktur. Built-in çözünürlükler için otomatik olarak doldurulacaktır. Create AVD butonu tıklandığında sanal telefon oluşturulacaktır. Debug konfigürasyonu yapmak için Run>Debug Configurations menü seçeneği tıklanır. Sol taraftan Android Application seçili durumdayken sol üst köşedeki + işaretine tıklayarak yeni bir konfigürasyon eklenir. Name alanına herhangi bir isim girip Browse butonu tıklanarak oluşturulan proje seçilir.

Launch Action bölmesindeki Launch Default Activity proje oluşturulurken, oluşturulan aktivitenin otomatik olarak çalıştırılmasını sağlar. Birden fazla aktivite olduğunda default aktivitenin yerine, başka aktivitelerin projenin açılışında çalıştırılması sağlanabilir. Target sekmesini tıklanır. Bu bölümde projenin hangi sanal makine üzerinde çalıştırılacağı seçilir. Daha önceden oluşturulan sanal telefonu işaretleyerek Close düğmesine tıklayıp,daha sonra Run>Debug menüsü seçildiğinde Eclipse emülatörü çalıştırmaya başlayacaktır. Emülatörün kurulumu ve ayarları şekil 2.16.'da gösterildiği gibi doğru olarak yapıldığında istenilen Android versiyonuna ait sanal cihaz olan emulator kurulmuş olup,uygulamayı bilgisayardaki emülatörde çalıştırma imkanı elde edilmiş olur.



Şekil 2.6. AVD ile sanal cihaz oluşturma ekranı

2.3. Uygulama Çeşitleri

Web ve native uygulama çeşitleri bulunmaktadır. Ağ üzerinden yüklenen , web tabanlı uygulamaların mobil cihazlara uygun hale getirilmesi için JQuery Mobile ve jQTouch gibi JavaScript uygulama çatıları geliştirilmiştir. Web tabanlı Android yazılımları HTML 5 ve CSS 3.0 alt yapısı kullanılarak geliştirilmektedirler. Tercihen MsSql veritabanı ile bağlantılı çalışan bu web tabanlı kodlama türleri uygulama şeklinde Google Play Developer üyeliği vasıtasıyla sisteme gönderilirler. Google Play uzmanları tarafından kontrol edilen uygulama yaklaşık bir-iki haftalık bir süre sonucunda onaylanarak Google Play Store' daki yerlerini alırlar.

Web tabanlı olduklarından web üzerindeki bir sunucu üzerinden çalışırlar ve admin paneli üzerinden istenilen sıklıkla kolaylıkla güncellenebilirler. Verilerin büyük çoğunluğu web üzerinde olduğu için uygulama kısa bir süre içinde indirilebilmektedir.

Offline olarak kullanılamazlar. Günümüzde Wi-Fi, Edge, 3G ve hatta 4G internet erişimine sahip olan kullanıcılar gittikçe artığı için bu dezavantaj da bu süreçle beraber önemini kaybetmektedir.

Native uygulamalar ise APK (Android Package) uzantısı ile cihazlara yüklenebilen uygulamalardır. Uygulama tabanlı yazılımlar tüm verileri ve veritabanını kendi içinde barındırmaktadır. Güncelleme gerektiğinde Android Developer' lar tarafından müdahale edilmeleri gerekir ve akabinde Google Play Store Developer Program' a gönderilirler. Uygulama yüklendikten sonra çalışmak için internet bağlantısına gerek duyulmaz. İstenildiği zamanlarda internet bağlantısı olmadan kullanılabilirler.

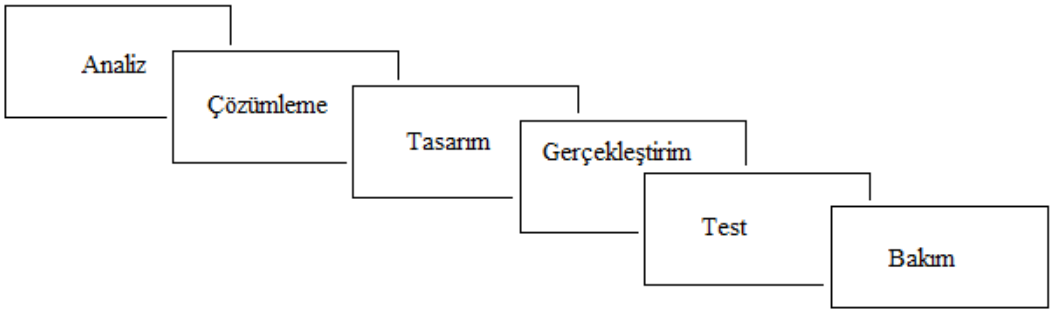
Çok fazla veri içeren uygulamaların dosya boyutları büyür ve uygulamanın indirme süresi bu nedenle veri miktarına bağlı olarak uzun olabilir. Güncelleme gerektiğinde Android Developer' lar tarafından güncellenebilir ve Google Play Store' a gönderilebilirler. Çok sık güncellenmeyecek uygulamalar için kullanılabilir.

Uygulamalar yüklenirken kullanıcıya hangi izinlerin istendiğini gösteren bir liste çıkar. Teorik olarak native uygulamalar cihazın tüm özelliklerini ve yeteneklerini kullanabilirler. Bu nedenle de web tabanlı uygulamalarda yapılamayan işlemlerin native uygulama olarak geliştirilmesi önerilir. Bir e-kitap okuyucu yazılımı internet tarayıcısı üzerinden sayfaların sunulması şeklinde de geliştirilebilir, PDF gibi doğal e-kitap formatlarının tamamını destekleyen ve bu kitapların satın alınabildiği bir mağaza gibi de sunulabilir.

Eğlence, iletişim, oyun, kurumsal uygulamalar, cihaz bakımı, sağlık, e-kitap okuyucuları, finans, kişiselleştirme, spor, medya, video tv gibibir çok alanda uygulama geliştirilmiştir. Ayrıca Android işetim sisteminin çok hızlı yayılması sebebiyle bir çok uygulamanın Android versiyonları çıkarılmıştır. Hem sosyal medyanın hem oyun üreticilerinin bir çok örneğinin Android platformuna taşıması gün geçtikçe artmaktadır.

3. MOBİL SAĞLIK YAZILIMI GELİŞTİRME

Yazılım geliştirme sürecinde aşağıda şekil 3.1.'de verilen sistematik yöntem kullanılmıştır.[43]



Şekil 3.1. Yazılım geliştirme yöntemi

Bu aşamalarda dikkat edilen noktalar şunlardır;

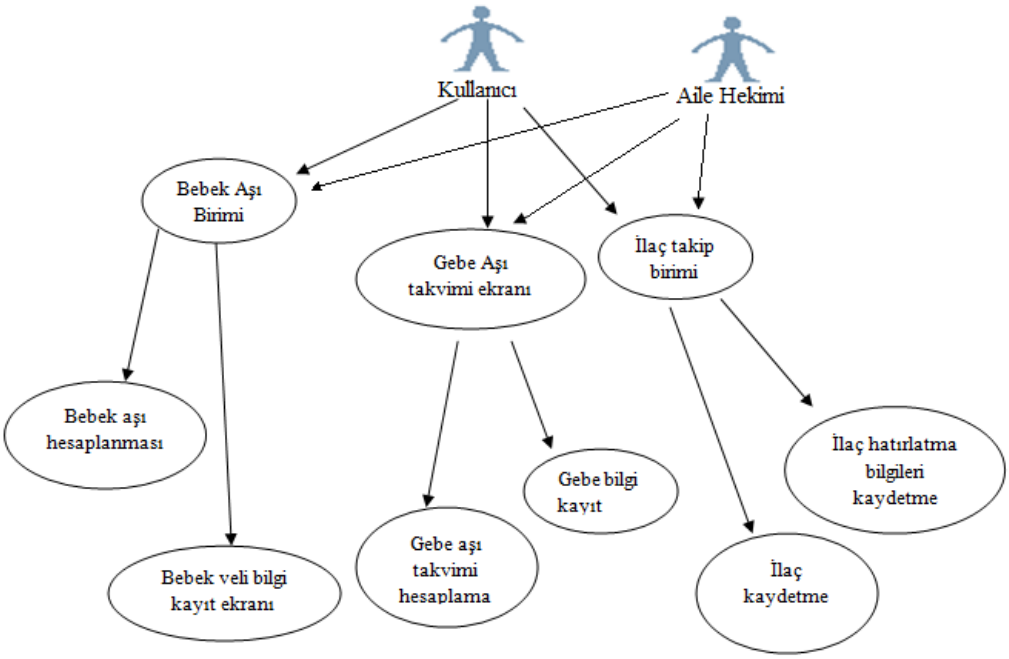
a) **Planlama**: Bu aşamada yapılacak çalşma için ilk ihtiyaçlar ve fizibilite çalışmaları yapılmaktadır. Yapılabilirlik, zaman ve maliyet konuları incelenip etüd yapılmalıdır.

b) **Yazılım İsterleri Çözümlemesi:** Önce bir kullanıcının, sonra aile hekiminin beklentileri ve gereksinimleri nin yazılım çözümlemelerinin oluşturulması. Bu aşamada sisteme dahil olacak her kademenin görüşleri alınıp bu paydaşlara sistem hakkında bilgiler sunulması ve sistemden beklentilerin, isteklerin net bir şekilde ortaya konulması gerçekleştirilmiştir.

c) **Yazılım Tasarımı:** Bu aşamada amaç, isterler çözümlendikten sonra geliştirilecek yazılım sisteminin modellemelerinin ortaya çıkarılması. Bunun için UML, SysML gibi araçlardan faydalanılmaktadır. Bunlarla beraber bir taraftan veritabanı işlemleri için veritabanı tasarımı ve yönetimi işlemleri gerçekleştirilmektedir. Veritabanı tasarımı ile ilgili olara tabloların kullanılacak sisteme ugun bir şekilde hazırlanması ve noralizasyon işlemlerinin gerçekleştirilmesi ile beraber kullanışlı bir yapıya kavuşturulması amaçlanmıştır. Tablolar arasında ilişkilerin ve alanların belirlenmesi ile beraber veritabanı sağlıklı bir yapıya kavuşmuş olmaktadır.

d) **Gerçekleştirme:** Bu aşamada medikal izleme sisteminin tasarlanan modellere göre kodlanması için Java dili ile, Eclipse ve Android SDK'dan oluşan bir platformda Android işletim sistemine uygun bir program geliştirilmesi.

e) **Test:** Geliştirilen yazılım bir aile hekimliği yönetim sisteminin işleyişi dikkate alınarak sanal ortamda test edilmektedir. Test aşamasında ortaya çıkan sorunların çözümlerine yönelik olarak sistemdeki modüllerin yapısının değiştirilmesi veya yeni modüller eklenmesi mümkün olmaktadır.



Şekil 3.2. Kullanıcı-Modül diyagramı

3.1. Tasarım ve Kodlama

Çalıştırılabilen kod kısımlarının içerisinde bulunduğu android sınıflarına activity adı verilir.çalışması istenilen java kodları aktivitelerin içerisinde bulunur. Her ekran içinbir activity uygulanır. Activity'ler kendi aralarında birbirlerini ihtiyaç anında çağırabilirler. Bununla beraber aralarında veri transferi yapabilirler. Belirli zamanlarda çalışıp kodun belirli kısımlarında kullanıcı ile ve sistemle etkileşime geçerek gerekli veriyi sağlayan ve kullanılmadıkları zaman sistem tarafından sonlandırılan parçalardır.

Activity uygulamalarda tek bir ekranı temsil eder. Uygulamalardaki ilk ekran **Default Activity**'dir.Sınıf tanımı aşağıdaki gibi yapılabilir.

public class MainActivity extends Activity

Görüldüğü üzere; bütün ekranlarda kullanılacak her bir ekran sınıfları için Activity sınıfından birer sınıf türetilmesi gerekmektedir

Aktivite kullanıcıların bir işlem yapmak için kullandığı ekranı oluşturan bir uygulama bileşenidir. Bu işlemler telefon çevirme, fotoğraf çekme, e-mail gönderme gibi

faaliyetleri içerir. Her aktivite aslında genellikle ekranı dolduran ve diğer pencerelerin üzerinde olan bir penceredir.

Bir uygulama birbiriyle bağlantılı birden fazla aktiviteden oluşur. Uygulama içindeki bir aktivite ana (main) aktivite olarak tanımlanır ve uygulama ilk çalıştırıldığında kullanıcı bu ekranı görür. Her aktivite farklı işlemler gerçekleştirmek için bir diğeri aktiviteyi çağırabilir. Yeni bir aktivite çalıştığında, önceki aktivite durdurulur, ancak sistem tarafından stack adı verilen bellek biriminde tutulur.

Yeni çalıştırılan bir aktivite nedeniyle bir aktivite durdurulduğunda, bu durum değişikliği aktivitelerin yaşam döngüsü metotları tarafından aktiviteye bildirilir. Durumundaki değişikliklere bağlı olarak, aktivitelerin alabileceği bir kaç metod vardır. Sistem aktiviteleri oluşturarak, durdurarak, tekrar başlatarak veya yok ederek gerçekleştirdiği durum değişikliklerinde, duruma uygun olarak özel işlemlerin yapılabilmesine olanak sağlar. Örneğin, durdurulduğunda, bir aktivite ağ ve veritabanı gibi büyük nesneleri serbest bırakmalıdır. Aktivite devam ettiğinde, gerekli kaynaklar tekrar elde edilir ve işlemler devam eder.

Bir aktivite oluşturmak için, Activity sınıfından veya bu sınıftan türetilmiş bir sınıftan bir alt sınıf türetmeniz gerekir. Oluşturduğunuz sınıf içinde, bir aktivite oluşturulduğunda, durdurulduğunda, devam ettiğinde veya yok edildiğinde yaşam döngüsündeki farklı durumlar arasında bir aktivitenin yaptığı geçişler esnasında sistem tarafından çağrılan geri çağrı metotlarının kod kısımlarını oluşturmanız gerekir.

En önemli geri çağrı metotları:

onCreate()

Bir aktivite oluşturulduğunda çağrılan bu metodun kodlarını yazmanız gerekir. Kodlar içinde, aktivitenizin önemli bileşenlerini başlatmanız gerekir. Daha da önemlisi, setContentView() fonksiyonunu çağırarak aktivite kullanıcı arabirimini oluşturmanız gerekir.

onPause()

Kullanıcı bir aktiviteyi kullanmayı bıraktığında, sistem bu metodu çağırır. Burada, yapmanız gereken değişiklikleri uygulamanız gerekir.

Bir aktivitenin kullanıcı arayüzü **View** sınıfından türetilen nesneler tarafından sağlanır. Her bir view kontrolü aktivite penceresi içinde belirli bir dikdörtgen alanı kontrol

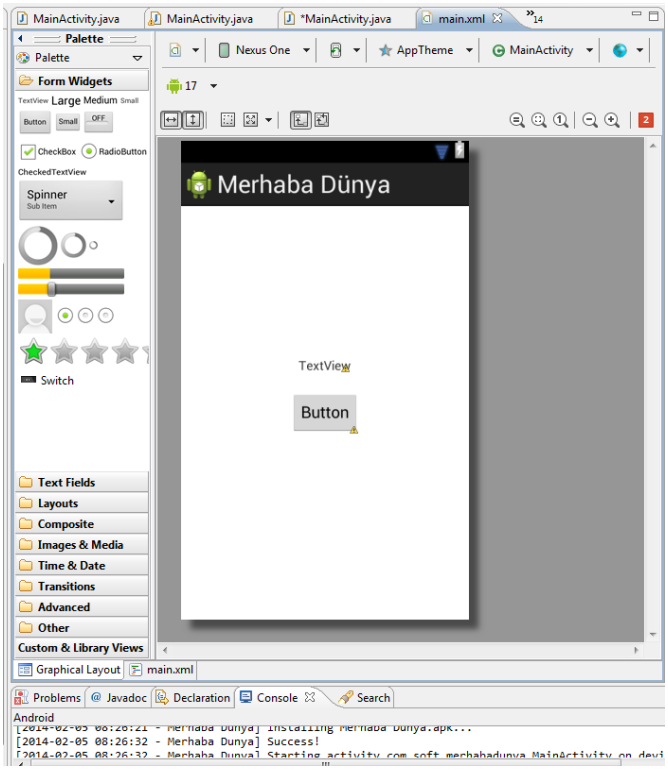
eder ve kullanıcı işlemlerine karşılık cevap verir. Örneğin, kullanıcı tıkladığında belirli bir işlem yapan bir buton bir view'dır.

Android, program arayüzünü oluşturmak için kullanabileceğiniz bir çok hazır view sağlar. Widget'lar ekran için görsel ve interaktif özelliği olan buton, metin alanı, checkbox gibi view'lardır. Ekran görüntüsü (layout) ViewGroup sınıfından türetilen ve çocuk view'ları için linear, grid veya relative görüntü gibi benzersiz bir görüntü sağlayan view'lardır. Aynı zamanda, kendi widget ve görüntülerinizi oluşturmak ve aktivite ekranınıza uygulamak için View ve ViewGroup sınıflarından veya mevcut alt sınıflarından yeni sınıflar türetebilirsiniz. Kullanımının nasıl olduğuna kısaca bakılacak olursa tablo2.2'de görüldüğü gibi bir activity sınıfı oluşturulması için gerekli kodlar genel hatlarıyla görülmektedir.;

Tablo 3.1. Activity kullanımı

```
Paket oluşturulması ve bileşenlerin import edilmesi
Aktivitenin oluşturulması
Kulanılacak ekran arayüzünün belirlenmesi
Button hesap=(Button)findViewById(R.id.button1);
hesap.setOnClickListener(new View.OnClickListener(){
Tıklama olayında yapılacakların belirlenmesi
String t="1.Aşı-(Aşı Türü)-aşı tarihi ";
Ekran çıktısının görüleceği yerin belirlenmesi
}}
```

Sınıf ismi olarak Mainactivity isminin kullanılması zorunlu değildir. Fakat android programlamada dikkat edilmesi gereken noktalardan birisi, java kodlamasının yapıldığı yerler olan activity'lerin xml dosyaları olan view bileşenleriyle ve projenin tamamının bir haritası olarak adlandırılabilecek olan manifest dosyası ile entegre olduğu unutulmamalıdır. Bu entegrasynun gereği olarak mainactivity adlandırmasının değiştirlmesi halinde xml dosyasından ve manifest dosyasından da gerekli değişikliklerin yapılması gerekmektedir. Manifest dosyası activity dosyasını çağırdığı gibi activity dosyası da xml dosyasını çağırmaktadır. Aşağıda bir xml dosyasının tasarım görünümü şekil 2.17'de verilmiştir. Eclipse içerisinden android uygulamalarının ekran görünümlerinin tasarlanıp nesne yerleşimlerinin yapıldığı ekran olarak xml içerikli view dosyaları kullanılmaktadır. Gerekli tasarım ve düzenlemeler eclipse programının içerisinde yapılabilmektedir.



Şekil 3.3. Android uygulama tasarım ekranı

Xml dosyalarının tasarlama yoluyla oluşturulmasının yanında kodlarla tasarlanması da mümkündür.Bu kodlama yapısı kullanıcı tarafından yapılmasa eclipse tasarıma gore kendi xml kodlarını oluşturmaktadır. Tablo 2.3'e bakıldığında şekil 2.17'de verilen ekran tasarımının xml kodları görülmektedir.

Tablo 3.2. Xml dosyası kod görünümü

Xml tanımlamaları
android:layout_width="match_parent"
android:layout_height="match_parent" >
İsteğe bağlı genişlik ve yükseklik belirleme
android:layout_centerHorizontal="true"
android:layout_marginBottom="19dp"
android:text="TextView" />
</RelativeLayout>

View'lar kullanarak görünüm tanımlamanın en yaygın yöntemi uygulama kaynakları arasına kaydedilen XML görüntü dosyası kullanmaktır. Bu yöntemle, programın kullanıcı arayüzünü aktivitedeki işlemleri yönlendiren kaynak kodlarından

bağımsız olarak tanımlamış olursunuz. Aktiviteniz için kullanıcı ara yüzünün görüntüsünü kaynak ID'sini setContentView() metodu ile kullanarak ayarlayabilirsiniz.

Sistem tarafından giriş yapılabilmesi için, aktivitenizin bildirimini manifest dosyasında yapmanız gerekir. Aktivitenin bildirimini yapmak için manifest dosyası açılıp, <application> elemanının alt elemanı olarak bir <Activity> elemanı eklenir ve

Tablo 3.3. Activity tanımlama

<Activity Android:name=".mainactivity">

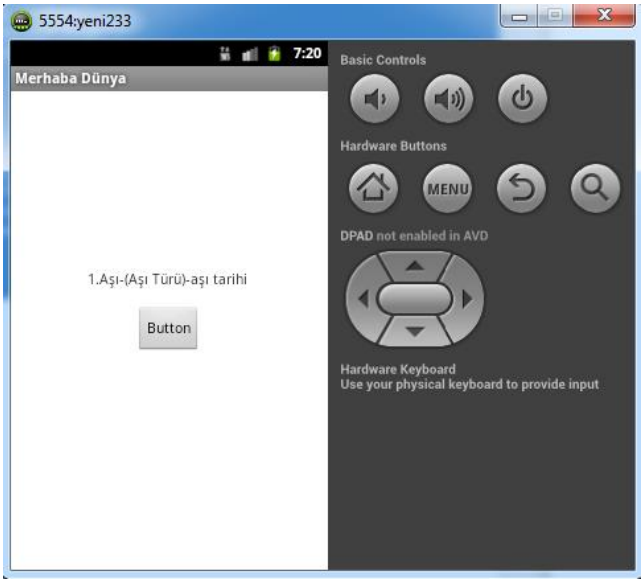
şeklinde bir tanımlama satırı oluşturulur.

Yukarıda verilen örnek programın manifest dosyasının içeriğine bakıdığında tablo2.4'de görüldüğü gibi manifest, application, activity etiketleri önem arzetmektedir.

Tablo 3.4. Android uygulamasına ait manifest dosyası içeriği

Manifest dosyası tanımlamaları ve versiyon belirlenmesi
Kullanılabilecek minimum ve hedef, yazılım araçlarının belirlenmesi
Aktivite tanımlanması
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</manifest>

Aktivite,xml görünümü ve manifest dosyası verilen kodların ekran çıktısına bakıldığında şekil 2.18'de görüldüğü gibi butonun üzerinde bulunan textview nesnesine aktivitenin içerisinde tanımlanmış t değişkeninin içeriğinin aktarıldığı görülmektedir. Daha once açıklaması yapılmış olan emulator oluşturma sayesinde program çalıştırıldığı zaman emulator otomatik olarak çalıştırılıp, belirlenen android versiyonunun açılması gerçekleştikten sonar hazırlanmış olan application dosyası çalıştırılıp ekran görüntüsü elde edilebilir.



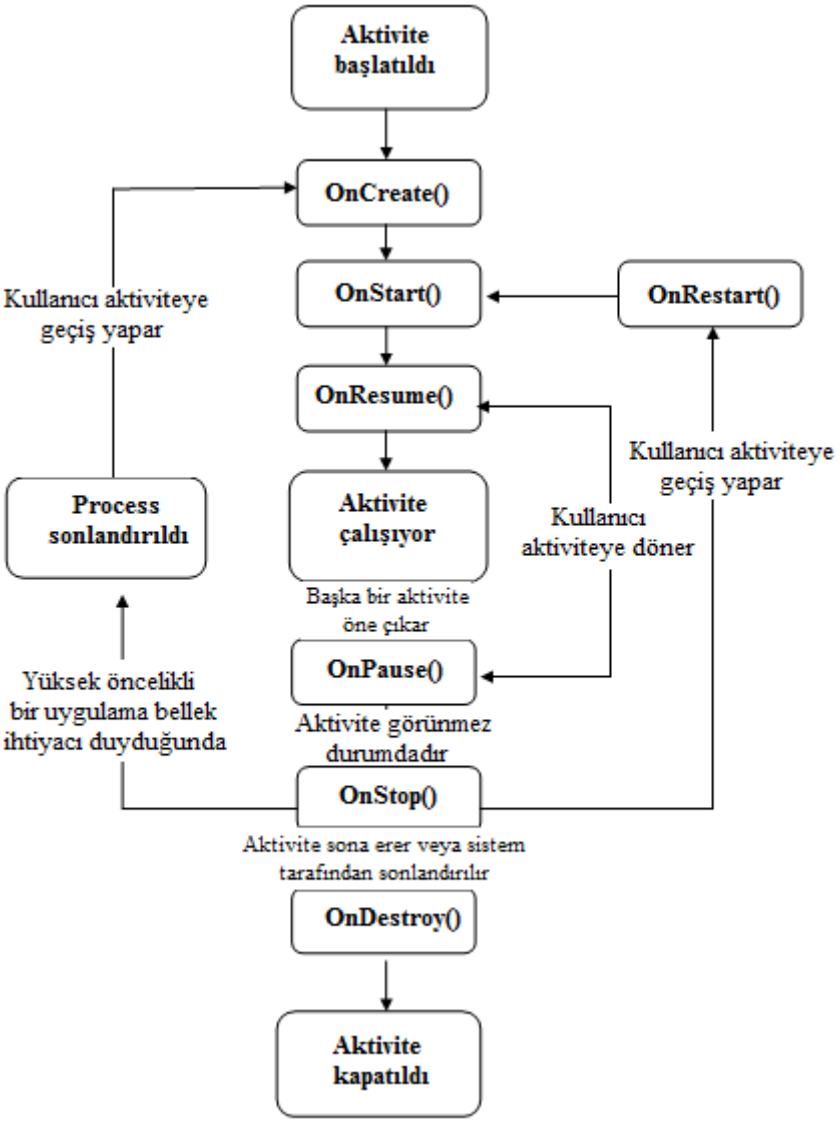
Şekil 3.4. Aktivite ekran çıktısı

Bir aktiviteyi başlatmak için, startActivity() metoduna başlatılmak istenilen aktiviteyi gösteren bir Intent nesnesini parametre olarak geçirerek bir aktivite başlatılabilir. Intent nesnesi başlatmak istediğiniz aktiviteyi veya gerçekleştirmek istediğiniz işlemi tanımlar. Sistem diğer uygulamalar içinde de olsa uygun olan aktiviteyi bulur ve başlatır.

Aktiviteler şekil 2.19'da görüldüğü gibi genel yapı olarak şu dört durumdan birinde bulunmaktadır. Çalışıyor (Active/Running) durumu adından da anlaşılacağı gibi uygulamanın tetikleyici bir olay ardında çalışarak kullanıcı ile etkileşim içinde olduğu durumdur. Bu durumdaki bir uygulama doğrudan kullanıcının gözü önünde olduğu için de Android işletim sistemince en yüksek öncelikle işletilmektedirler, sistem kaynakları öncelikle bu durumdaki uygulama tarafından kullanılmaktadır. Askıda (Paused) durumu cihazın uyku durumuna geçmesi ya da uygulama ara yüzünün ekranda görünür olmasına karşın aslında transparan bir başka aktivitenin çalışıyor durumda olması durumu için geçerlidir. Askıda bulunan uygulama hala durum değişimlerinden haberdar olabilmektedir.

Android işletim sistemi açısından ikinci en öncelikli uygulamalar askıda durumunda olanlardır. Arkaplanda (Stopped/Backgrounded) durumu kullanıcının gözü önünde olmayan, aktif olarak kullanımda olmayan uygulamaların durumudur. Durmuş gözüyle de bakabileceğimiz bu uygulamalar yukarıda sıraladığım durumlarla kıyaslandığında Andorid işletim sistemince en düşük öncelikte olan uygulamalardır. Sistem kaynaklarının olası yetersizliği durumunda öncelikle arkaplanda durumunda olan uygulamalar sonlandırılacaktır. Çalışmıyor (*Destroyed*) durumu, hafızada dahi

bulunmayan durumdaki uygulamalar olduğu için hiç bir sistem kaynağını kullanamazlar.[44]



Şekil 3.5. Android hayat döngüsü

Aktivitelerde kullanılacak yazı, resim, renk,ölçü,animasyon veya ekran tasarımı gibi bileşenlerin sabit olmayıp belli bir kaynaktan alınmasını sağlayan yapılar mevcuttur. Bu aktivitelerin daha esnek bir yapıda çalşabilmesini sağlar. Bu sayede uygulama için yapılması gereken köklü değişimler bile rahat bir şekilde çözülebilir. Aktivite ekranlarında kullanlan yazıların renklerin ölçülerin tamamı xml dosyaları olarak saklanır, proje çalıştırıldığında bu kaynaklardan veriler okunup ekranın nasıl olması gerektiği bilgisi alınır ve kullanıcıya sunulur.

Bu ölçüler android projelerinde res klasörü içerisinde kayıtlıdırlar. metin dosyaları,resim dosyaları,renk verileri,menü tanımları,layoutlar, xml konfigürasyon dosyaları,temalar,stiller,binary dosyaları (pdf, mp3, video dosyaları) gibi dosyalar strings.xml > String kaynakları, colors.xml > Renk kaynakları, dimens.xml > Ölçü kaynakları, arrays.xml > Dizi kaynakları,styles.xml > Stil kaynakları şeklinde saklanır.

Tanımların ayrı bir kaynak dosyasında yapılmasının sağladığı kolaylık, sadece kaynak dosyasında değişiklik yaparak bütün ekranlarınızın arka planını (ya da tanımını yaptığınız başka herhangi bir rengi) tek seferde değiştirebilme olanağı sunmasıdır.

String ifadelerin program içerisinde kullanımı aşağıdaki biçimde kullanılarak belirlenebilir.

Tablo 3.5. String kullanımı

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
<string name=" txt_selam">Merhaba Android,
Merhaba Dünya...</string>
<string name="app_name">Merhaba_Android
</string>
</resources>
```

Color Resource tanımında kullandığımız değer # ile başlar ve Alfa (şeffaflık), Kırmızı, Yeşil, Mavi değerlerini 16'lık düzende aşağıdaki formatlarda alabilir.

Tablo 3.6. Renk tanımlama formatları

#RGB #ARGB
#RRGGBB
#AARRGGBB

Proje hiyerarşisinde res klasörü içinde drawable-hdpi, dravvable-ldpi, dravvable-mdpi gibi 3 tane grup vardır. 3 farklı klasör olmasının sebebi, cihazın ekran çözünürlüğüne bağlı olarak hdpi - high (yüksek) dpi, mdpi - medium (orta) dpi, Idpi - low (düşük) dpi klasörlerinden bir tanesindeki çizimlerin kullanılacak olmasıdır. Ekran boyutuna bağlı olarak farklı boyutlarda simgeler kullanılarak, tutarsız görüntülerin oluşması engellenir.

Layout tasarımları kullanılarak ekranların içerisinde bulunan bileşenlerin nasıl yerleştirilmesi gerektiği belirlenebilmektedir. Yatay, dikey ya da nesnelerin ilişkisel

yerleştirilmesinin istenilmesi durumunda linear, vertical, horizontal, releative gibi layout seçenekleri seçilerek yerleşim tanımlanmış olur ve artık projeye eklenen verilerin ekranda dizilim şablonu kararlaştırılmış olmaktadır. Layoutların eklenmesi sayesinde istediğimiz belirli yerleşimlere uyabilecek ekranlar oluşturulabilmektedir. Yoksa araçların, nesnelerin hizalanmaları mümkün olmayacaktır. Bu düzenleme sayesinde layout dosyaları için uygulamamızdaki etkinliklerin tasarımını oluşturan dosyalardır denilebilir. Xml türünde olan bu dosyayı kullanarak etkinliklerin nasıl görüntüleneceğine karar verilir. Xml dosyasını elle düzenlemenin yanında bu dosyayı görsel bir şekilde düzenleme imkanı da bulunmaktadır. Layoutlar arayüz için kontrolleri yerleştirmenize yarayan ViewGroup sınıfından türetilmişlerdir. Layoutlar iç içe kullanılabildiği gibi değişik kombinasyonları ile karmaşık arayüzler de oluşturulabilmektedir. Android SDK arayüz tasarımı için bazı hazır basit layoutlar içermektedir.

FrameLayout: En basit layouttur ve tüm view'ları sol üst köşeye yerleştirir. Birden fazla view eklendiğinde her zaman son eklenen view bir öncekini görünmez yapar. Nesneler grafik tasarımı yapılırken üstüste biner. Program içerisinde nesnelerin belirli şartlara gore arkaplana geçmeleri, görünüp görünmemeleri durumlarında kullanılabilir.

LinearLayout: İçerisine yerleştirilen view'ları yatay veya dikey olarak yerleştirir. Eklenecek olan nesneler doğrusal olarak eklenir. Eğer vertical linear layout kullanılırsa nesneler alt alta, horizontal linear layout kullanlırsa yan yana eklenir. Layout oluşturulduktan sonar orientation özelliği değiştirilerek de vertical veya horizontal yapılabilir. Tasarım ekranında layout eklenebilmesi için sol tarafta bulunan paletlerden layouts tıklandıktan sonra Linearlayout seçilip ekrana bırakılabilir. Layout eklendikten sonra içerisine diğer nesnelerin yerleşimi orientation özelliğine bağlı olarak belirlenir. Bu yerleşimleri xml dosyası içerisinden kodlarla da değiştirmek mümkündür. Tablo 2.5'de görüldüğü gibi kodlarla layoutun belirlenmesi, nesnelerin hizalanmasının sağlanması ve nesnelerin oluşturulması da mümkündür. Kodlama da dikkat edilmesi gereken noktalardan bir tanesi buton, textview gibi nesnelerin layout etiketlerinin içerisine yerleştirilmesidir.

Tablo 3.7. Linear layout-orientation:horizontal olan bir xml dosyası kodları

```
Xml tanımlanması
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android: layout_height="fill_parent"
android: orientation="horizontal" >
Kullanılacak nesneler
</LinearLayout>
```

RelativeLayout: İlişkisel bir dizilim oluşturulabilmesinin sağlar. Nesneleri birbirlerine gore yerleştirme ihtiyacı hissedilen projelerde kullanlabilen,esnek bir layouttur ve alt view'ları ekran sınırlarına veya diğer viewlara bağlı olarak viewlarınızın pozisyonunu belirlemenize imkan sağlar. Nesnelerin yerleşimlerini yaparken nesneleri istenilen yere bırakmaya izin verir.Belirli bir hizayı takip etmeden araçları isteğe gore yerleştirmek mümkündür.

TableLayout: Adındanda anlaşılabileceği gibi satır ve sütunlar kullanarak viewların yerleştirilmesini sağlar. Bu layout çeşidinde araçlarımız bir tablo şeklinde tutulur. Tablelayout kullanıldığı zaman tablonun içerisine eklenecek her bir satır için TableRow adında satır eklenmesi gerekmektedir.

Gallery: Gallery layoutu tek satırlık bir içeriği yatay olarak görüntülemeye yardımcı olur. Bunlarla beraber layoutların içerisinde başka layoutları kullanmak mümkündür. Bunlara ek olarak RelativeLayout'a benzer olan AbsoluteLayout bulunmaktadır. AbsoluteLayout'ta kullanacak bir view'ın x ve y koordinatlarının kesin olarak belirtilmesi gerekir. Kullanımı zor olduğundan çok fazla tercih edilmezler. AbsoluteLayout'ların yerine RelativeLayout kullanılabilir. Aktivitelerin hangi layout dosyasını kullanacağını ise aktivitenin içerisndeki setContentView satırı belirlemektedir.Bu satırda ifade edilen komutun yanında hangi layoutun adı yazılmışsa aktivite o layout dosyasını ekranı olarak alıp çalıştırır. Farklı cihazlar için farklı layoutlar oluşturulabilir. Örneğin; /res klasörü altında layout-land/ adında bir klasör tanımlanırsa, burada ki layout cihaz yatık konumdayken görüntülenir. layout-large/ adında bir klasör oluşturulursa, bu layout büyük ekranlarda (tablet gibi) görüntülenir. Uygulamadaki farklı ekranlar için farklı isimlerde Layout'lar oluşturulabilir.

Uygulamanın farklı ekranlarda nasıl göründüğüne bakmak için Graphical Layout görünümündeyken "Preview All Screen Sizes" ya da "Preview Representative Sample" seçerek farklı cihazlardaki görünümüne bakılabilir. Yine, cihaz adına tıklayarak, manual olarak varsayılan cihaz görünümü değiştirilebilir. Uygulamaya başka bir dilde destek vermek için, string.xml dosyasından yeni dil için bir kopya oluşturulması yeterlidir. Yeni dosya, /values-tr /values-es gibi bir klasöre kopyalanırsa Türkçe ve İspanyolca sürümleri elde edilmiş olur. /values-pt-rBR /values-pt-rPT gibi adlandırarak Brezilya Portekizcesi, Portekiz Portekizcesi gibi yerel sürümün de seçilmesi sağlanabilir. LinearLayout tag'ının içerisinde yer alan layout_width ve layout_height layoutun ekran boyutlarının belirlenmesine yarar. Rakamsal değer alabileceği gibi hazır değerlerde alabilmektedirler. Ekran boyutlarındaki değişkenlikten dolayı rakamsal değerler kullanmak fazla tercih edilmektedir. Hazır değerlere ise;

fill_parent: Bir üst layout'u tamamen doldurur. Eğer layout en üstteki layout ise ekranı kaplar

match_parent:fill_parent'in API level 8'den sonraki adıdır. Kısacası her ikisi aynı görevi görür.

wrap_content:Layout'unuzun kullandığınız view kadar büyük olmasını belirtir.

EditText nesnesine girilen metnin, Button nesnesine basıldığında TextView nesnesinde görüntülenmesi için yapılan kodlama örneğinde bir aktivitenin nasıl çalıştığı görülebilmektedir.[26] Ekranda bir TextView, bir EditText ve Button nesnesi ile çalışan denemeactivty.java dosyası şekil 2.24'de görülmektedir. Yapılan ekran tasarımının kodları ise şekil 2.24'de görülmektedir.

Tablo 3.8. DenemeActivity.java dosyasının içeriği

```
Bileşenlerin projeye eklenmesi
public class denemeActivity extends Activity
{private EditText eText;
private TextView tView;
private Button bGonder;
public void onCreate(Bundle savedInstanceState)
{super.onCreate(savedInstanceState);
Kullanılacak ekran ve bileşen tanımlamaları
bGonder.setOnClickListener(new OnClickListener()
{
Butona tıklandığında yapılacak işler
```

});}}

Tablo 3.9. Main.xml dosyasının içeriği

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
>
Kullanılacak bileşenlerin eklenmesi
```

Intent

Kelime anlamı olarak niyet, maksat anlamına gelir. Proje içerisinde bileşenler arasında gönderilen mesajlar intent ile ifade edilir. Intent ile herhangi bir aktivity başlatılabileceği gibi bir servis de çalıştırılabilir, arkaplanda işlemler de yürütülebilir.

Intent kullanımında genellikle yapılan işlemlerden biri olan sayfalar arası geçiş işlemini gerçekleştirmek için iki sayfanın tasarlanması ve ilk sayfadan ikinci sayfaya geçişi başlatacağımız bir tetikleyici nesnenin bulunması gereklidir. Bununla birlikte manifest dosyası içerisinde kullanılacak olan java sınıflarının tablo 2.8'deki şekilde tanıtılması gereklidir.

Tablo 3.10. Aktivitelerin intent için manifest dosyasında tanımlanması

```
Uygulama tanımları
Intent ile çalışacak ekran tesbiti
</activity>
<intent-filter>
<action android:name="view.holder.gecisiyonet" />
<category android:name="android.intent.category.DEFAULT" />
</intent-filter>
</activity>
</application>
</manifest>
```

Manifest dosyası oluşturulduktan sonra ilk ekran olan gecisactivty dosyasının şekil 2.27'deki biçimde xml içeriğinin oluşturulması gereklidir.(Akyüz,2014) Ekran için

minimum ihtiyaç olarak bir buton olmalıdır. Buton sayesinde intent çalıştırılıp ikinci sayfaya geçişi gerçekleştirilmelidir.

Tablo 3.11. Gecisyap ekranının tasarım kodları

```
Layout tasarım kodları
Buton tanımı
android:text="gecis yap"
android:id="@+id/btn_ gecisyap "
android:layout_width="match_parent">
</LinearLayout>
```

Ikinci ekran olarak ekrana gelmesi istenilen ekran için de yeni bir xml dosyası oluşturulup isimlendirilmeli ve şekil 2.28.'de görüldüğü gibi şekillendirilmelidir.

Tablo 3.12. İkinci ekranının tasarım kodları

```
Layout tanımlamaları
Kullanılacak nesne tanımlamaları
<TextView android:textAppearance="?android:attr/textAppearanceLarge"
android:text=""
android:layout_height="wrap_content"
android:layout_width="wrap_content"
android:id="@+id/textView_metin"></TextView>
<EditText android:layout_width="match_parent"
android:layout_height="wrap_content"
android:id="@+id/editText_metin">
</EditText>
<Button android:text="Gönder"
android:id="@+id/btn_Gonder"
android:layout_height="wrap_content" android:layout_width="match_parent">
</Button>
</LinearLayout>
```

Ekran tasarımları tamamlandıktan sonra ilk açılacak java dosyasına intent için gerekli kodların yazılması gereklidir. Bu bölümde yönlendirme yapmak için gerekli kodlar bulunmalıdır. Ilk ekrandaki butonun onClick olayı ile çağırılacak olan aktivitenin manifetst dosyasındaki tanımlanmasına uygun olarak şekil 2.29.'da görüldüğü gibi çağrılması gereklidir.

Tablo 3.13. Birinci ekranın java kodları

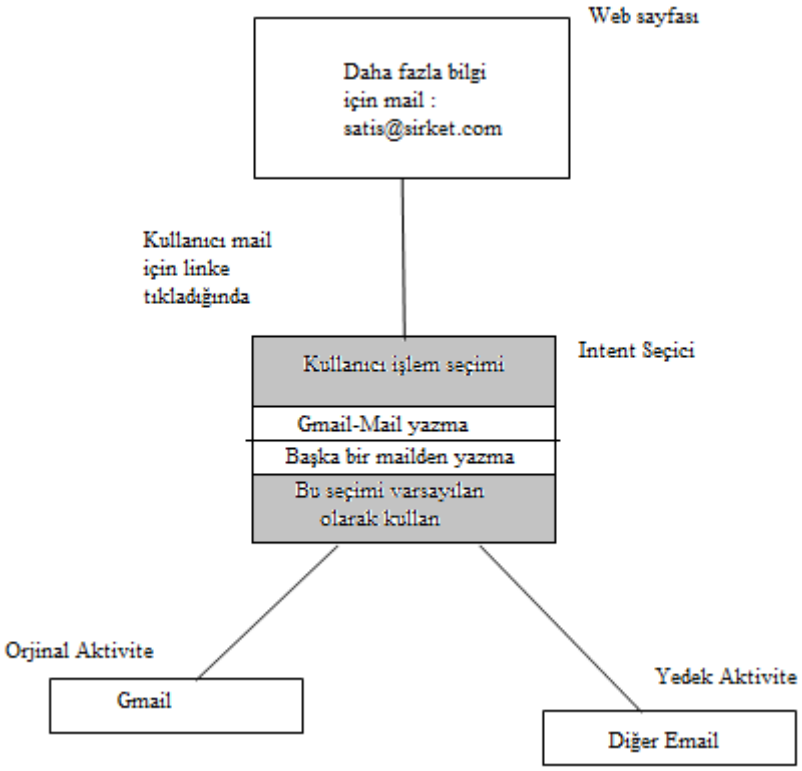
```
public class gecisyapActivity extends Activity {
public void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
Buton tanımlaması
Butonun yapacağı fonksiyonun belirlenmesi}
}
```

Çağırılan java dosyasında ise birinci dosyadan gönderilen intenti alıp işleyecek bir yapı şekil 2.30'da görüldüğü gibi hazırlanmalıdır.

Tablo 3.14. İkinci ekranın çalışma kodları

```
Paket ve bileşen tanımlamaları
public class formuYonet extends Activity
{
private EditText eText;
private TextView tView;
private Button bGonder;
protected void onCreate(Bundle savedInstanceState)
{
Ekranın ve nesnelerin görsel seçimi
// Button nesnesine tıklandığında çağırılan inner class yapısı
bGonder.setOnClickListener(new OnClickListener(){
public void onClick(View v){
tView.setText(eText.getText());}});}}
```

Bu yapı sayesinde bir ekrandan başka bir ekrana geçiş ve ekranlar arası bilgi göndermenin nasıl yapılacağı intent kavramı ile gösterilebilir. Bu aşamalar arasında kullanıcının isteğine bağlı olarak şekil 2.31'de görüldüğü gibi yapılması gereken işlerin hangi uygulama ile gerçekleştirileceğinin kullanıcının tercihine bırakılması da intent ile sağlanabilmektedir.[23]



Şekil 3.6. İntent bileşeninin çalışması

İki çeşit intent kullanılmaktadır: Explicit ve implicit. Implicit Intent; ne yapmak istenildiğinin bilindiği, ama bunu kimin yapması gerektiği konusunda bir karar verilmemiş durumlarda kullanılır. Bu durumda programcı sadece niyetinin olduğunu belirtir. Gerisiyle ilgilenmez. O işi yapabiliecek tek bir uygulama varsa o başlatılır, birden fazlaysa kullanıcının seçimine bırakılır. Kısaca niyet açıkça belli ise, yani hangi uygulamanın veya hangi aktivitenin çalıştırılacağı belli ise explicit intent kullanılır. Eğer çağrılan bileşenin veya verinin hangi uygulama ile çalıştırılacağı belli değil ise veya sistemin seçim yapması istenilen uygulamalar için implicit intent kullanılır.

Service

Servisler uygulamaların arka planında ve herhangi bir arayüze sahip olmadan çalışan bileşenlerdir. Bir servis başlatıldıktan sonra, çalışan uygulamalara bakmaksızın çalışmaya devam edecektir. Bu sebeple kullanıcılar ile ara yüz ihtiyacı duymayan işlemler için servisler kullanılabilir. Servisler activity classlarına benzer yapıları vardır. Activity classlarındaki onResume ve onPause gibi fonksiyonların yerine servislerde onStart ve onDestroy

adında yardımcı fonksiyonlarımız bulunmaktadır. Şekil 2.32'de servis uygulaması için gerekli bir ana java dosyası gösterilmektedir.

Tablo 3.15. Ana dosya kodları

```
Ana ekran ve buton tanımlaması
Kullanılacak ekranın belirlenmesi
Nesnelerin ekrandaki hangi nesnelere karşılık geleceğinin belirtilmesi
public void onClick(View v) {
Intent için kullanılacak sınıfın belirlenmesi
Servisin başlatılıp durdurulması için gerekli kodlar
Servisin ve nesnelerin durdurulması
return true;}}
```

Servislerin çalışabilmesi için kendilerine ait bir servis dosyasının eklenmesi şekil 2.33'de görülmektedir.[50]

Tablo 3.16. Arka planda çalışmayı sağlayan servis dosyası

```
Servis adı ve kullanılacak nesnelerin tanımlanması {
public void onStart(Intent intent, int startId){
Log.i("onStart", "Service onStart");
Bileşenin tanımlanıp çalışmaya başlatılması
Bileşenin yapacağı işin belirlenmesi
nesne.setOnBufferingUpdateListener(new OnBufferingUpdateListener() {
Servis içerisindeki nesnelerin başlatılıp durdurulması işlemleri
```

Ayrıca servis dosyasının çalışabilmesi için gerekli olan izinlerin manifest dosyası içerisine şekil 2.34.'de gösterildiği gibi girilmesi gereklidir.

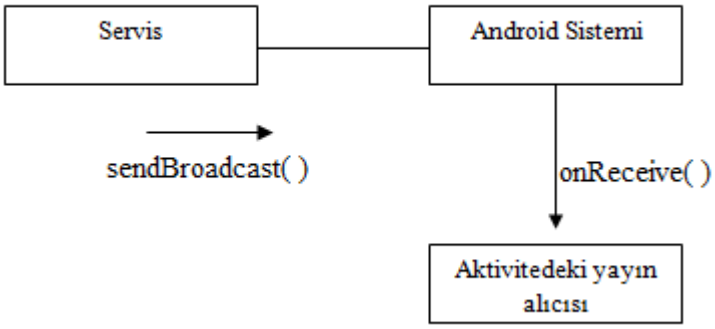
Tablo 3.17. Servis dosyası için manifest izin kodları

```
<service android:enabled="true" android:name="Servisin adı" />
<uses-permission android:name="android.permission.INTERNET"></uses-permission>
```

Broadcast Receivers

Broadcast Receivers; işletim sistemi içinde gerçekleştirilen olayları ve bunlara karşı yapılması gereken işlemleri çalıştırabilmeyi sağlayan yayın alıcılardır. Işletim sistemi

tarafından uygulamalara gönderilen sinyalleri dinlemek ve takip etmekle görevlidirler. Ayrıca geliştirilen uygulamalardan sistemin tamamına veya belli bölümlere haber göndermek amacıyla kullanılabilirler. Bu şekilde uygulamalar telefona gelen çağrılardan,kısa mesajlardan ve diğer olaylardan haberdar olurlar. Yapılan yayınları algılayıp onlara göre çalıştırılması gerekli bölümlere yönlendirme yapan alıcılara ise intent receivers adı verilir. Bu birimler yayınların veri ve mesajlarını alıp ilgili birimlere veya activity'lere şekil 2.35'de gösterildiği gibi ulaştırma vazifesini üstlenirler.[37] Bu sebeple karşılıklı çalışmaktadırlar.



Şekil 3.7. Broadcast receiverin çalışma yapısı

Sistem üzerinde herhangi bir olay esnasında Android işletim sistemi yayın yapar. Örneğin bir sms geldiğinde, telefon çaldığında, batarya uyarıları vs böyle durumlarda sistem haber verir. Broadcast Receiver'lar Broadcast Intentleri beklerler. Haberdar olmak isteyen uygulamalar, bu yayınlara kayıt olarak arzu ettikleri işlemleri gerçekleştirebilirler. İlgili intent geldiği takdirde yayın yapılır.[45] Bir nesneye BroadcastReceiver özellliğinin eklenebilmesi için öncelikle o nesnenin AndroidManifest dosyasında kaydedilmesi gerekmektedir. İşletim sisteminin gönderdiği sinyalleri alma yetkisi bu şekilde kazandırılabilir. Telefona kısa mesaj geldiğinde uygulamanın harekete geçmesi istenildiğinde şekil 2.36'da görüldüğü gibi manifest dosyasında filtrelemelerin yazılması gereklidir.

Tablo 3.18. Manifest dosyasında broadcastreceiver tanımlanması

```
<receiver android:name="Broadcast Receiver adı" >
<intent-filter>
<action android:name="android.provider.yayınınadı"/>
</intent-filter>
</receiver>
```

BroadcastReceiver bileşenleri yardımıyla telefondaki her türlü değişikliğin algılanabilmesi mümkün olabilmektedir. Telefonun titreşiminin aktif edilmesinin sağlanması için öncelikle manifest dosyasında titreşim izninin şekil 2.37'de görüldüğü şekilde alınması gerekmektedir.

Tablo 3.19. Titreşimin aktif edilmesi

```
<uses-permission android:name="android.permission.VIBRATE" />
```

Bununla birlikte uçuş modunu tesbit edecek bir sınıfın şekil 2.38'de hazırlanmış olan dosyaya benzer bir şekilde oluşturulması ve bu sayede telefonun mod değiştirip uçuş modunda olup olmadığını belirlenmesi gereklidir.

Tablo 3.20. Titreşim tercihini tesbit eden java sınıfı

```
package com.code.broadcastreceiver;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.os.Vibrator;
public class titresimmodu extends BroadcastReceiver {
public void onReceive(Context context, Intent intent) {
Vibrator vibrator = (Vibrator) context.getSystemService(Context.VIBRATOR_SERVICE);
vibrator.vibrate(5000);}}
```

Bunlara ilave olarak titreşim modunu tesbit eden sınıfın manifest dosyasında şekil 2.39.'da belirtildiği gibi alıcı olarak tanımlanması gerekmektedir.

Tablo 3.21. Java sınıfının receiver olarak tanımlanması

```
<receiver android:name=".AirplaneMode" >
<intent-filter>
<action android:name="android.intent.action.AIRPLANE_MODE" />
</intent-filter>
</receiver>
```

Yayın alıcı bileşeninin kullanımı için mutlaka alıcı tarafın tasarlanması ve eğer programcıya bırakılmışsa yayın tarafının da tasarlanması ve kodlanması gereklidir. Uygulamalar için bir alıcı oluşturulduğunda, şekil 2.40.'da belirtildiği gibi bir alıcı kodları oluşturulmalıdır.Bu işlem için,manifest dosyasında alıcı belirtilirken, android:name anahtarına karşılık olarak kullanılacak bir anahtar değeri tanımlanmalıdır.

Tablo 3.22. Manifest dosyasında alıcı tanımlama

```
<receiver android:name="Receiver adı " >
<intent-filter>
<action android:name=" Java dosyasının tam adı" />
</intent-filter>
</receiver>
```

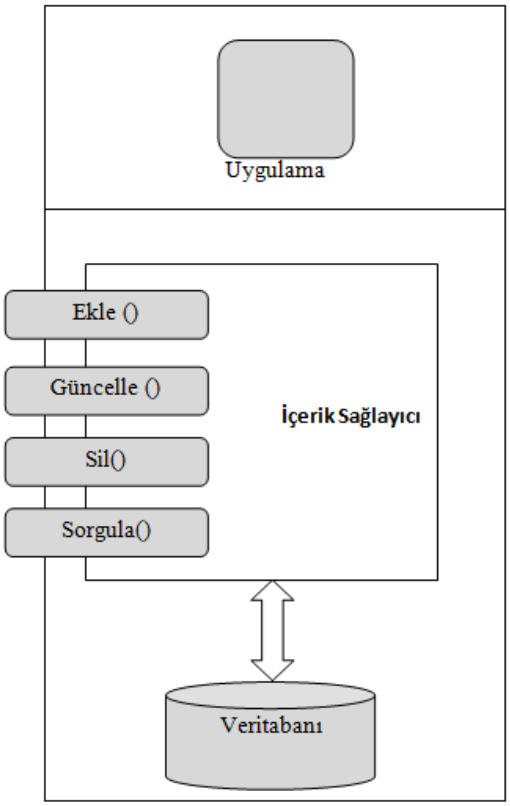
Burada com.codek.Receiverdeneme adında broadcast(yayın) yapıldığında tetiklenip onReceive metodu harekete geçecektir. Eğer yayın gönderme işlemi de tasarlanmak isteniyorsa, kodların içerisinde şekil 2.41'de belirtildiği gibi çağrı yapılması da gerekmektedir.

Tablo 3.23. Yayın gönderimi kodları

```
Intent intent = new Intent( );
intent.setAction("Java dosyasının tam adı ");
sendBroadcast(intent);
```

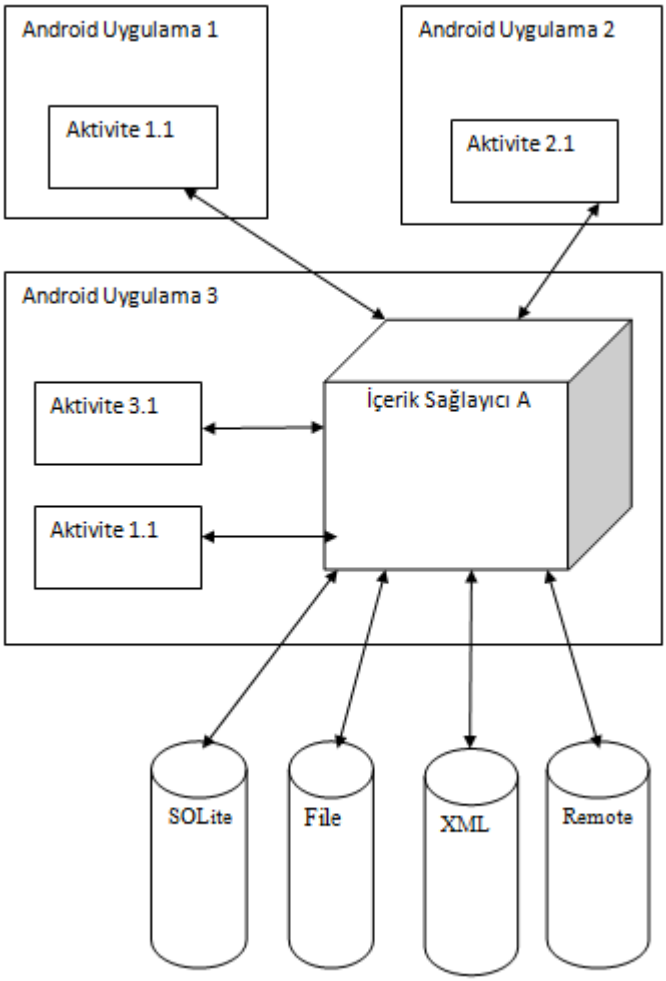
Content Providers

Uygulamalar kendilerine ait verileri kayıt edebilirler, silebilirler, düzenleyebilirler. Bu bilgileri ya dosya sisteminde yada dosya sistemi üzerindeki SQLite veritabanı üzerinde şekil 2.42'deki gibi saklayabilirler.



Şekil 3.8. Content provider'a ait çalışma yapısı

Android işletim düzeninde veri depolanması ve paylaşılması durumlarından Content Provider (Bilgi, Veri, İçerik Sağlayıcı) bileşeni sorumludur.[13] Uygulamalar içerisindeki bilgilerin bazı durumlarda diğer uygulamalarla paylaşılması gerekebilir. Bazen bir uygulama başka bir uygulamanın sakladığı veriye ihtiyaç duyabilir. Uygulamaların bir birlerinin verilerinden istifade etmesi de söz konusu ise uygulamanın kodlandığı zaman programcısı tarafından belirlenen izin sınırlarına göre bir birleri arasından veri paylaşımları da yapabilirler.[22] Böyle bir ihtiyacın oluşması durumunda Content Providers bileşenleri, veriyi standart bir interface (arayüz) aracılığıyla yönetebilmeyi sağlayan yapılar olarak kullanılabilmektedir. İçerik sağlayıcıları kullanarak şekil 2.43'de görüldüğü gibi veriler dış dünyaya açılabilir ve kimlerin nasıl erişebileceğine karar verilebilir.[13]



Şekil 3.9. Content provider bileşeninin uygulamalar arası kullanımı

İzin verildiği takdirde farklı uygulamalar Content Provider üzerinden veritabanına erişebilir, query, update, insert ve delete işlemlerini gerçekleştirebilir. Android sistemi içerisinde kullanıcıların bilgilerini tutabileceği yerlerde sistem herkesin ihtiyaç duyacağı verileri kendi sistem veritabanında tutar ve Content Provider'lar sayesinde diğer uygulamaların kullanımına sunar. Erişime izin verilen bütün uygulamalar bu veritabanlarını kullanarak, kullanıcılara farklı kullanım seçenekleri sunarlar. Bu çalışma yapısından dolayı android işletim sistemi tabanlı akıllı cihaz kullanıcıları yükledikleri her uygulamanın kurulum aşamasında istedikleri izinlere dikkkat ederek kuruluma izin vermek veya vermemek tercihinde bulunmalıdırlar. Content provider olarak telefon rehberinden belirlenen şartları sağlayan bilgileri alan bir kod yapısı şekil 2.44'de görülmektedir. [33]

Tablo 3.24. Kriterlere uyan kişileri textview'e ekleme

```
Paket ve import edilen bileşenlerin belirtilmesi
Aktivitenin isminin belirlenmesi {
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.main);
TextView tv = (TextView) findViewById(R.id.txt1);
getContacts(tv);}
public void getContacts(TextView tv) {
String[] projection = null; // new String[]{
String selection = Kriterin belirlenmesi;
String[] selectionArgs = Seçim kriterinin belirlenmesi;
String sort = Sıralama kriterinin belirlenmesi;
Cursor cursor = getContentResolver().query(
ContactsContract.Contacts.CONTENT_URI,projection,selection,selectionArgs,sort);
while (cursor.moveToNext()) {
Gösterilmek istenen özelliklerin tanımlanması
Textviewde görüntülenmek istenen niteliklerin belirtilmesi
If (Integer.parseInt(hasPhone) > 0) {
Cursor crPhones = getContentResolver().query(
ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
null,
ContactsContract.CommonDataKinds.Phone.CONTACT_ID + " = " + contactId,
null,
null);
while (crPhones.moveToNext()) {
String phoneNumber = crPhones.getString(
crPhones.getColumnIndex(
ContactsContract.CommonDataKinds.Phone.NUMBER));
Görüntülenecek sonuçların formatının belirlenmesi
crPhones.close();}}}}
```

İstenilen şartlara gore telefon rehberinden sorgulama yapan sınıfın ekran çıktısı adı K harfi ile başlayan ve telefon numarası olanları gösterecek şekilde oluşacaktır.

Bu yapı sayesinde telefona indirilen bir bilgi için, uygulamayı da takip etme gibi bir zorunluluk içerisine girmez. Bu bileşenin kullanılabilmesi için data'yı dış dünyaya sunmak gerekir ki bunun ilk adımı Content Provider için bir isim bulmaktır. Bu isim, farklı veritabanlarını ayırt edebilmek için eşsiz olmak zorundadır. Bu isme Provider Authority

adı verilir ve Android, bu isimlerin nasıl verileceğine dair bir standart bir yol önermektedir. Önerilen isimlendirme yöntemi com.<şirket adı>.<uygulama adı>.provider.<provider adı> şeklindedir. Böyle isimlendirmek zorunluluğu bulunmasa da, anlaşılırlık bakımından böyle yapılmalıdır. İsimlendirme şekline bakıldığında ilk kısmın aslında package name isimlendirme şekline benzer olduğunu görülebilir

Uygulamanızdaki bir veritabanı sizin uygulamanızın erişimine açıktır. Başka bir uygulamanın da sizin veritabanınızı okuyabilmesi, içine kayıt ekleyebilmesi için bilgilerin işletim sistemi genelinde paylaşılması gerekmektedir. İçerik sağlayıcılar bu aşamada veritabanınızı farklı uygulamaların erişimine açmanızı ya da bu uygulamalar tarafından sağlanan içeriğe ulaşmanızı sağlar.

Android işletim sisteminin kullanıma sunduğu Contacts Provider, Settings Provider ve Media Store gibi içerik sağlayıcıları vardır. Örneğin herhangi bir medya programının kişisel bilgiler olan resimlere, müzik, video gibi dosyalara erişebilmesi ve yeni kayıt eklemesi, mevcut kayıtları okuması, silebilmesi ve sorgulama yapabilmesi gerekmektedir. Bu sayede farklı temaların aynı sistem üzerinde farklı şablonlarla ve grafiklerle çalışabilmesi sağlanmaktadır.

Content provider yapısı ile bağlantılı olarak Android sistemine özel olan en önemli güvenlik yöntemlerinden birisi olan sandbox yapısı önem arzetmektedir. Çünkü bu yapı sayesinde uygulamaların her birinin ayrı bir dalvik sanal makine üzerinde çalıştırlması sağlanmıştır. Sandboxlar yapı olarak kısıtlı ereişim haklarına sahiip çalışma ortamlarıdır. Ayrı çalıştırılan bu sanal makineler birbirlerinden yalıtılmış olarak çalışmaktadırlar. Bu korunaklı sistem sayesinde sistemin kaynaklarına erişim belirli bir kontrol altına alınmıştır. Herhangi bir uygulama kendisine ait sandbox yapısının dışında bir bileşene erişim sağlayamamaktadır.

Manifest Dosyasi

AndroidManifest.xml, uygulama hakkında önemli bilgilerin sistem tarafından okunulmasını sağlar. Bu dosya uygulamaların kök dizininde bulunmak zorundadır. Manifest dosyası uygulama için ID görevi gören java paketini isimlendirir, uygulama bileşenlerini (Aktiviteler, Servisler, Yayın Alıcıları, İçerik Sağlayıcılar) tanımlar ve hangi durumlarda çalışacağını belirler, bileşenlerin türetildiği sınıfların yetkilerini ve isimlerini belirler, bu bileşenlere hangi işlemlerin sahiplik yapacağını belirler,uygulamanın yetkileri

tanımlanır,uygulamanın çalışabilmesi için asgari Android sürümü tanımlanır,uygulamanın kullandığı kütüphaneleri listeler. Manifest dosyası, proje hiyerarşisinin en tepesinde yer alır ve uygulamanın bütün parçalarının bir araya geliş şeklini şekil 2.45'de görüldüğü gibi tanımlar.

Tablo 3.25. Bir manifest dosyası içeriği örneği

```
<?xml version="1.0" encoding="utf-8">
<manifest xmlns:android=http://schemas.android.com/apk/res/android
package="com.example.merhaba"
android:versionCode="1"
android:versionName="1.0">
<uses-sdk
android:minSdkVersion="8"
android:targetSdkVersion="18">
<application
android:allowbackup="true"
android:icon="@drawable/ic_launcher"
android:label="@string/app_name"
android:theme="@style/apptheme">
<activity
android:name="com.example.merhaba.MainActivity"
android:label="@string/app_name">
<intent-filter>
<action Android:name="Android.intent.action.MAIN"/>
<category android:name="Android.intent.category.LAUNCHER"/>
</intent-filter>
</activity>
</application>
</manifest>
```

Elementler

AndroidManifest.xml dosyasında "manifest" ve "application" elementleri olmak zorundadır. Diğer elementler ise bir çok kez kullanılabilir. Ya da hiç kullanılmaya da bilir. Dosya içindeki aynı seviyedeki elementlerin belli bir sırası yoktur, istenilen düzende yazılabilir.

Nitelikler

Bazı nitelikler dışında hepsi opsiyoneldir. Kök element olan “manifest” dışındaki tüm elementlerin nitelikleri “android:” ön eki ile şekil 2.46’da görüldüğü gibi başlamak zorundadır.

Tablo 3.26. Örnek bir nitelik kullanımı

```
<application
android:allowBackup="true"
android:icon="@drawable/ic_launcher"
android:label="@string/app_name"
android:theme="@style/AppTheme" >
```

Sınıf Tanımlamaları

Manifest dosyası içindeki bir çok element java nesnelerine karşılık gelir. Uygulama özelliklerini gösteren “application” elementi içindeki “activity” aktiviteleri, “service” servisleri, “receiver” yayın alıcıları ve “provider” içerik sağlayıcıları temsil eder. Her element için isim belirtilmek zorundadır ve şekil 2.47’de görüldüğü biçimde isim paket adını da içermelidir. Örneğin;

Tablo 3.27. Sınıf tanımlaması kullanımı

```
<manifest >
<application>
<service android:name="Servis ismi">
...
</service>
</application>
</manifest>
```

Kaynak Değerleri

Bazı nitelikler kullanıcıya da gösterilmek istenebilir. Mesela, aktivite için kullanacağınız bir resim, ya da ses dosyası. Bu dosyaların manifest.xml dosyasında yerleri şekil 2.48’de gösterildiği gibi aktiviteye bildirilmelidir.

Tablo 3.28. Resim kullanımı

<activity android:icon="@drawable/smallPic" ...>

Intent Filtreleri

Uygulama bileşenleri intentler ile aktive edilir. Uygulamanın Android ile iletişimini intentler sağlar. Intentler uygulamalar arası erişim izinlerini belirlemede, Android ile uygulama arasında iletişim için, bileşenler arasında veri akışında kullanılır. Uygulama bileşenlerinin izinleri intent filtrelerine tanıtılmalıdır, bu işlem manifest dosyasında "intent-filter" elementi olarak tanımlanır.

İzinler

İzinler, uygulamanın cihazın herhangi bir bölümüne ya da cihazdaki veriye erişimi sınırlandırmak için kullanılır. Bu sınırlama kullanıcının zarar görmesini engellemek ve veri güvenliğini sağlamak içindir. Yapılacak her önemli değişiklik ya da erişim hakkı için şekil 2.49'da gösterildiği gibi izin tanımlanması gerekir.

Tablo 3.29. Örnek izin kullanımı

android.permission.CALL_EMERGENCY_NUMBERS
android.permission.READ_OWNER_DATA
android.permission.SET_WALLPAPER
android.permission.DEVICE_POWER

Uygulamaya vermek istediğiniz izni manifest.xml dosyasında "uses-permission" elementi ile tanımlanmalıdır.

Kütüphaneler

Uygulamalar varsayılan Android kütüphanelerini bağlıdır ve temel paketleri içerirler. Bazı paketler ise kendi kütüphanelerine bağlı olabilirler. Bu gibi durumlarda manifest.xml içinde "uses-library" elementi ile kütüphane tanımlaması yapılabilir.

Sqlite Veritabanı Sistemi

Uygulama geliştirme süreçleri içerisinde veritabanı yönetim sistemlerine özel bir yer ayrılmaktadır. Genel olarak kullanılan sistemler arasında MSSQL, MySQL, PostgreSQL, Oracle, IBM DB2 gibi veritabanı yönetim sistemleri bulunmaktadır. Android ortamında uygulama geliştirmek için ise SQLite Kütüphanesi kullanılmaktadır. SQLite ile relational(ilişkisel) veritabanı tasarımları yapılabilmektedir.www.sqlite.org adresinde ayrıntılı bilgilere ulaşılabilmektedir. Sqlite C programlama dili ile yazılmış, açık kaynak kodlu, ilişkisel bir veritabanı kütüphanesidir. Diğer veritabanı istemlerinden farkı serverless(sunucusuz) olarak çalışmasıdır. Oluşturulan uygulama içerisinde farklı bir işlem gerektirmeksizin gömülü olarak çalışan bir veritabanı yönetim sistemidir.

Sqlite veritabanı kullanımı için projeye bir kütüphane ekleme şeklinde eklenmektedir. Google bu özel yapısından dolayı Android içerisinde uygulamanın kendine özel bir veritabanı oluşturulması adına kütüphane olarak projeye eklenmesine karar vermiştir. Veritabanı oluşturmak için kullanılan sınıf SQLiteOpenHelper sınıfıdır. Bu sınıftan kullanacağımız bir sınıf türetilmesi yoluyla uygulamanın veritabanı oluşturma ve güncelleme gibi işlemlerini sisteme devretmesi sağlanmaktadır. Kullanımı şekil 2.50'de gösterildiği gibi yapılabilmektedir.

Tablo 3.30. SQLiteOpenHelper sınıfı kullanımı

```
public class DenemeDBHelper extends SQLiteOpenHelper {
static final String dbAdi="dbtest";
static final String kisitable ="kisi";
static final String _id="_id";
static final String ad ="ad";
static final String soyad ="soyad";
static final String yas ="yas";
static final int version =1;
```

Sqlite veritabanı yönetim sistem'inin bir ara yüzü olmadığından dolayı, farklı programlar ile veritabanı içeriği görülmektedir. SQLManager, SQLite Expert Professional gibi programlarla veya Mozilla Firefox üzerine entegre edilen SQLManager adlı bir eklenti ile veritabanı içeriği ve takibi görüntülenebilmektedir. Tablo oluşturma için kullanılan

onCreate metodu içerisşnde bir kişi tablosu oluşturma için gerekli kodlar şekil 2.51'de görülmektedir.

Tablo 3.31. Sqlite kütüphanesi onCreate metodu kullanımı

```
public void onCreate(SQLiteDatabase db) {
db.execSQL ("CREATE TABLE"+ kisitable+"("+_id+" INTEGER PRIMARY KEY AUTOINCREMENT, "
+ ad + " TEXT, "+soyad+" TEXT, "+yas+" INTEGER);");}
```

Veritabanında yapılacak güncelemeler için kullanılan sınıf onUpgrade sınıfıdır.Kullanımı şekil 2.52'de görülmektedir.

Tablo 3.32. onUpgrade metodu kullanımı

```
public void onUpgrade(SQLiteDatabase db, int eskiVersiyon, int yeniVersiyon){
db.execSQL("DROP TABLE IF EXISTS"+kisitable);
onCreate(db);}
```

Veritabanı kapatılması için close() metodu, okunabilir veritabanı açılması için select cümleleriyle kullanılmak üzere getReadableDatabase() metodu kullanılır. Update, delete, insert gibi yazma işlemleri için getWritableDatabase() metodu kullabılabilir.[26] Sqlite ile standart SQL cümleleri ile sorgulama yapılabileiceği gibi kendine ait özelleşmiş sorgulama metotlarını da içerisinde barındırmaktadır. Uygulama içerisinde Sqlite veritabanı kullanılarak yapılan kayıt oluşturma işlemleri şekil 2.53'de görüldüğü gibi oluşturulmuştur.

Tablo 3.33. Kayıt oluşturma işlemleri

```
package com.example.saglik;
Kullanılacak bileşenlerin eklenmesi
import android.app.Activity;
import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
```

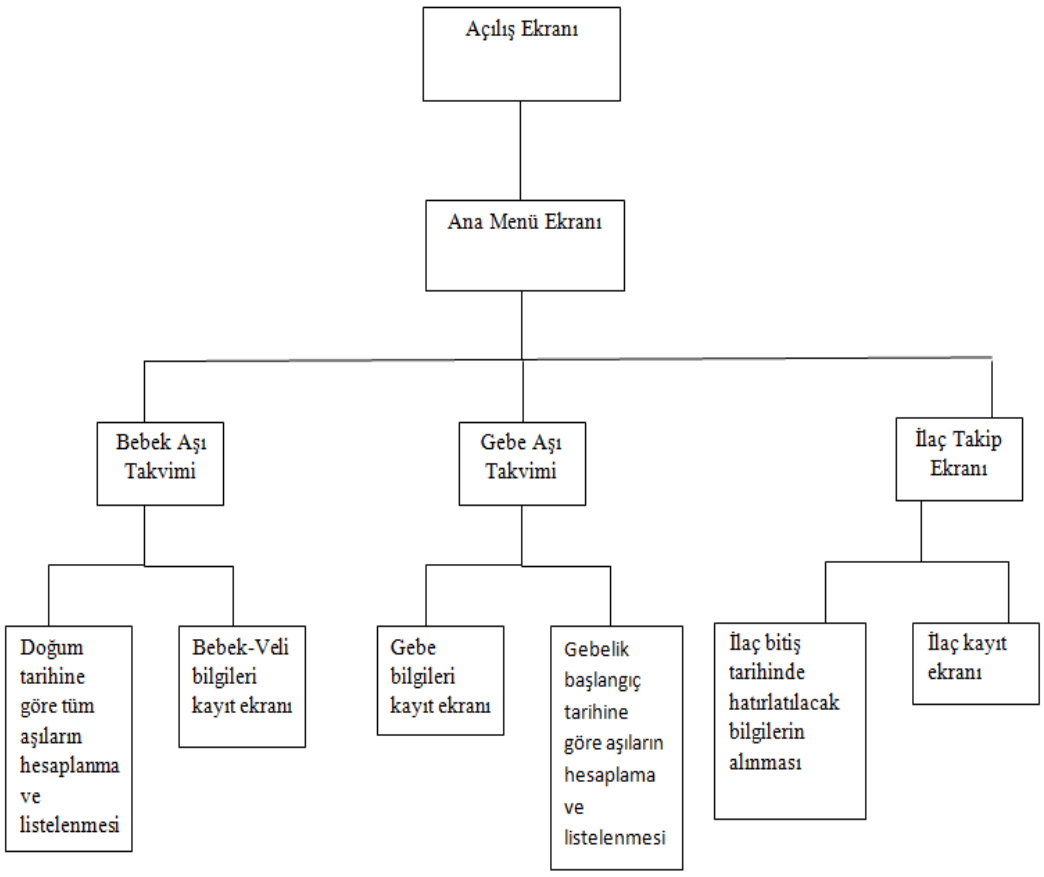
```
public class kayitekran extends Activity {
private Veritabani saglik;
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.kayitdeneme);
saglik = new Veritabani(this);
Kullanılacak nesnelerin oluşturulup isimlendirilmesi
final EditText b_adi=(EditText) findViewById(R.id.editText1);
final EditText b_soyadi=(EditText) findViewById(R.id.editText2);
final EditText b_tc=(EditText) findViewById(R.id.editText3);
verigonder.setOnClickListener(new View.OnClickListener() {
public void onClick(View v) { try{
KayitEkle(b_adi.getText().toString(),
b_soyadi.getText().toString(),
Diğer nesnelerin aynı şekilde Stringe dönüştürülmesi
Cursor cursor = KayitGetir();
KayitGoster(cursor);    }      finally{         saglik.close();
} }   });  } private void KayitEkle(String b_adi,
String b_soyadi, String b_tc, String b_dog_tar, String v_ad, String v_soyadi,
String v_tc,          String v_tel){ SQLiteDatabase db = saglik.getWritableDatabase();
ContentValues veriler = new ContentValues();
veriler.put("b_adi", b_adi);
Verilerin alanlara yerleştirilmesi
db.insertOrThrow("bebekveli ", null, veriler);     }
private String[] SELECT = {"id", "b_adi",
"b_soyadi","b_tc","b_dog_tar","v_ad","v_soyadi","v_tc","v_tel",};
private Cursor KayitGetir(){
SQLiteDatabase db = saglik.getReadableDatabase();
Kayıtların okunması işlemleri
private void KayitGoster(Cursor cursor){
Veri alanlarının sonuna kadar veri çekilmesi için gerekli kodlar
builder.append(id).append(" Bebek Adı: ");
builder.append(b_adi).append(" Bebek Soyadı: ");
Kaydedilen verilerin ekrana aktarılması
TextView text = (TextView)findViewById(R.id.textView11);
text.setText(builder);}}
```

3.2. Sistem Şeması

Gezgin işletim sistemlerinde çalışabilecek bir medikal takip sistemi geliştirilmesi hedeflenmiştir. Geliştirilen medikal izleme sisteminde;

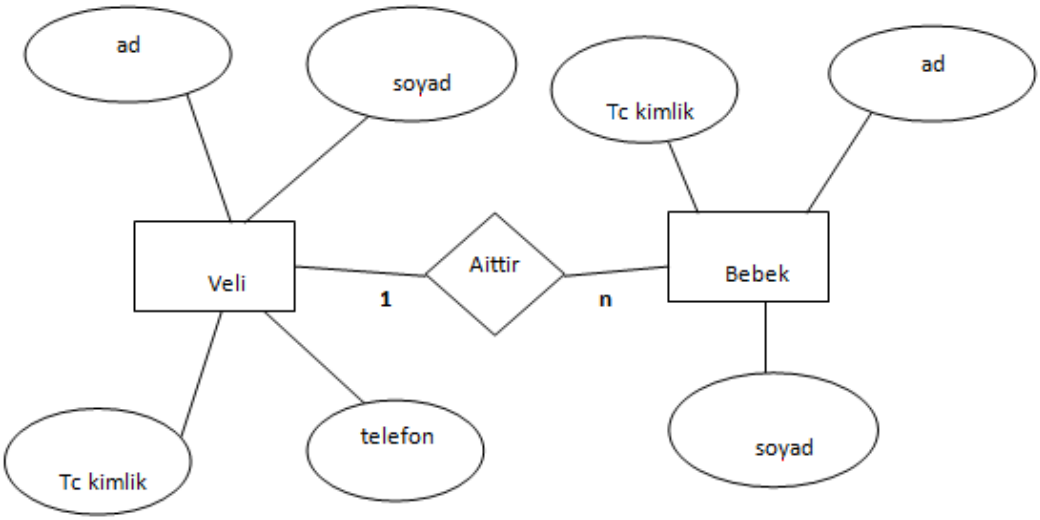
- Bebek aşılarının
- Gebe kontrollerinin
- Kronik hastaların ve bunların kullandığı ilaçların izlenmesi

için gerekli ortam oluşturularak gerekli bilgilendirmelerin ilgili taraflara verimli bir şekilde ulaştırılması ve bilgilerin sorgulanması esas alınmıştır. Bu yapı sayesinde kullanıcı taraflar ihtiyaç duydukları bilgileri şekil 3.2.'de diyagramı gösterilen tek bir program içerisinde bulabilme imkanına kavuşmaktadırlar. Uygulama genel şeması olarak verilen şekil 3.3'deki grafiğe bakıldığında uygulamanın yapısı açık bir şekilde görülebilmektedir.



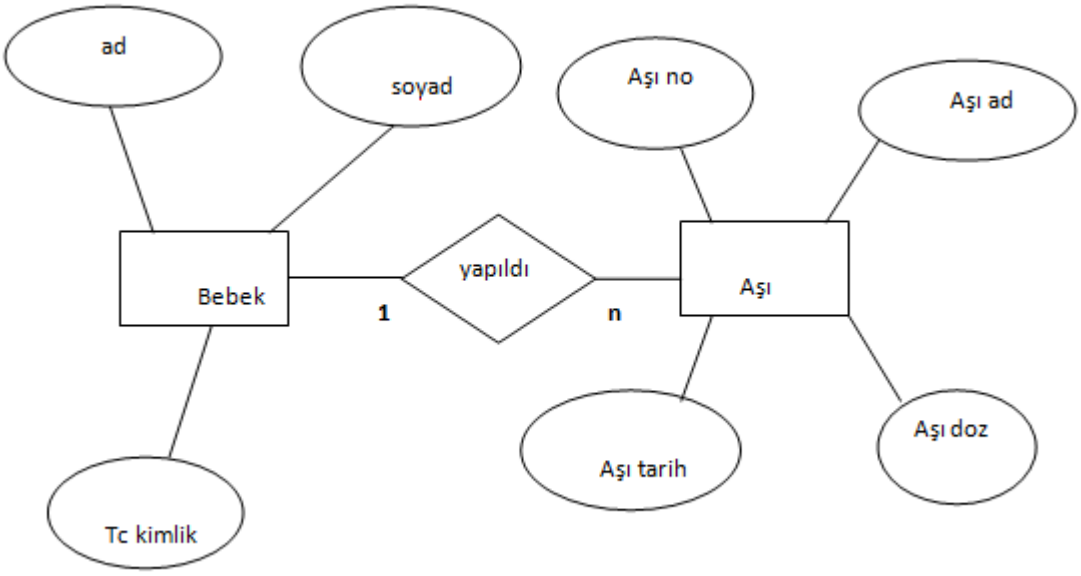
Şekil 3.10. Uygulama genel yapısı

3.3. Veri Modeli Oluşturulması ve Veritabanı Tasarımı



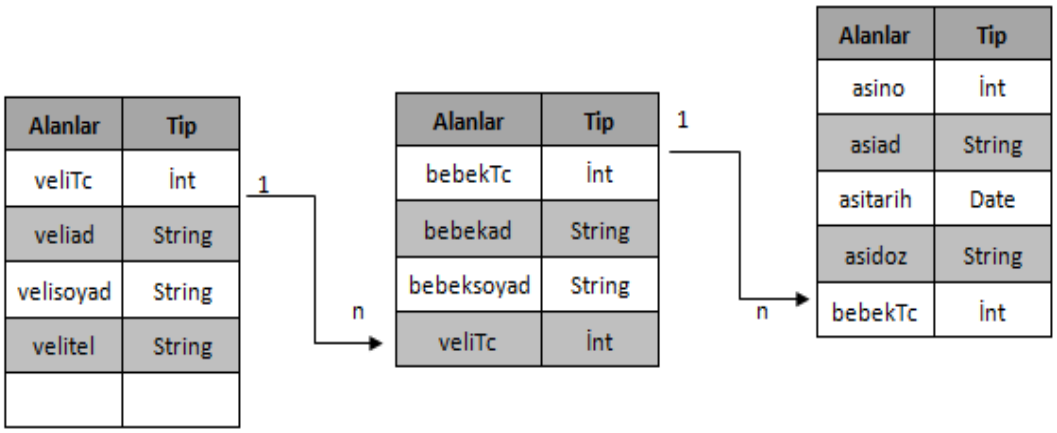
Şekil 3.11. Veli ve bebek varlıkları ilişki modeli

Ayrıca bebek-aşı ilişkisi için yapılan varlık-ilişki modeli şekil 3.16'da gösterildiği biçimde hazırlanmıştır.



Şekil 3.12. Bebek aşı varlıkları ilişki modeli

Tasarlanan ilişki modelleri şekil 3.17'de görülen tablo ve ilişkilere dönüştürülerek tablolaması yapılmıştır.



Şekil 3.13. Veritabanı veri modeli

Kullanıcı uygulamayı çalıştırdığında karşılama ekranı olarak şekil 3.4.'de görülen splash animasyon ekranı ile açılmakta ve daha sonra uygulama için ana ekran olarak tanımlanan, ilgili bölümlerin butonlarının bulunduğu ve bölümlere geçişlerin yapılacağı ekran, şekil 3.5'de görülmektedir.

Şekil 3.14. Splash ekranı

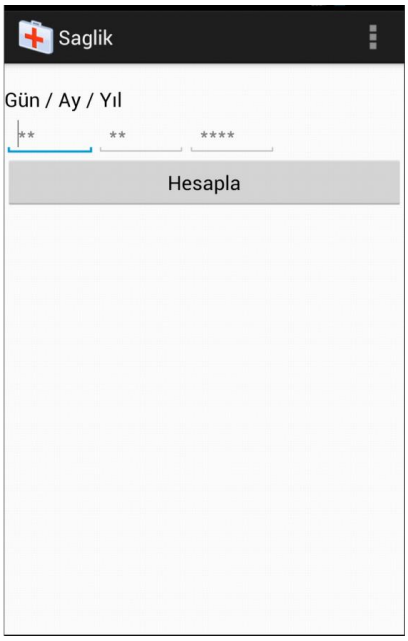
Şekil 3.15. Kullanıcı menü seçim ekranı

İlk karşilama ekranı olarak splash ekran ve içeriğe uygun bir resimle giriş yapılıp ana ekrana ulaşılması ve kullanıcı takvim hesaplatma sayfalarına ulaşmak için gerekli butonların bulunduğu menu ekranı tasarlanmıştır. Geçiş ekranı için kullanılan kodlar şekil 3.6'da verilmiştir.

Tablo 3.34. Splash ekran açılış kodu

```
public void run (){
try {
sleep(500);
startActivity(new Intent("com.example.saglik.EKRAN1"));
} catch (InterruptedException e) {
e.printStackTrace();}
finally{
finish(); } };
```

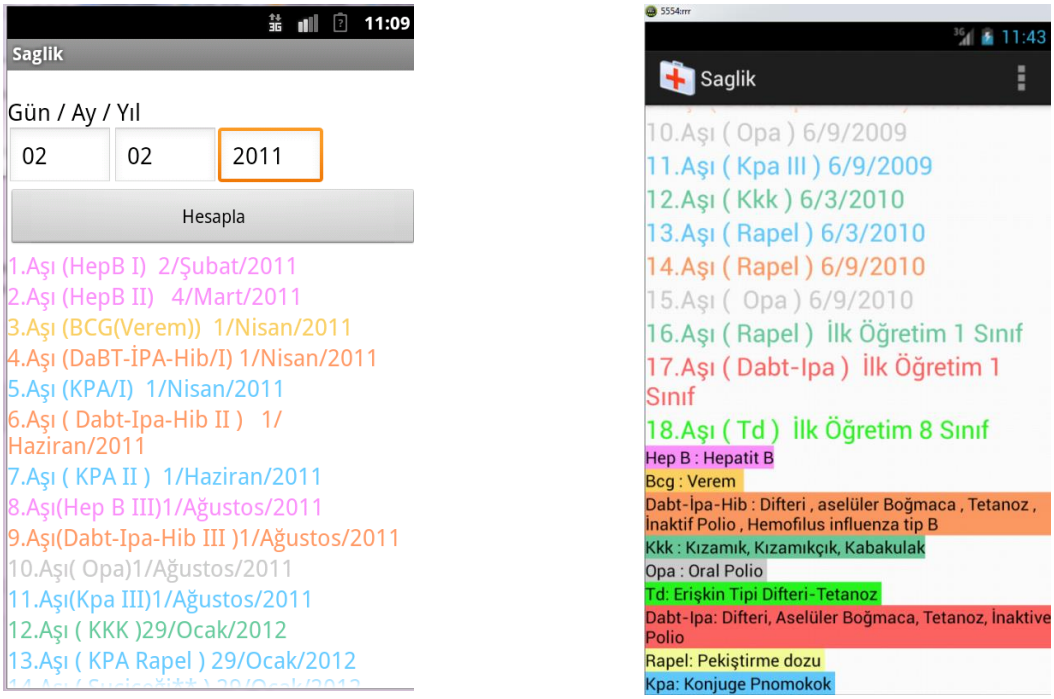
Ana ekranda bulunan menülerden bebek aşı takvimi hesaplamalarının bulunduğu bebek aşı takvimi butonuna basıldığında kullanıcının karşısına şekil 3.7'de görülen aşı takvimi hesaplamasının yapılabilmesi için gün/ay/yıl bilgilerinin girilmesini bekleyen 3 adet edittext ve butonla beraber şu an için içleri boş olan textviewler gelecektir.



Şekil 3.16. Kullanıcıdan bebek doğum tarihi istenen bilgi giriş ekranı

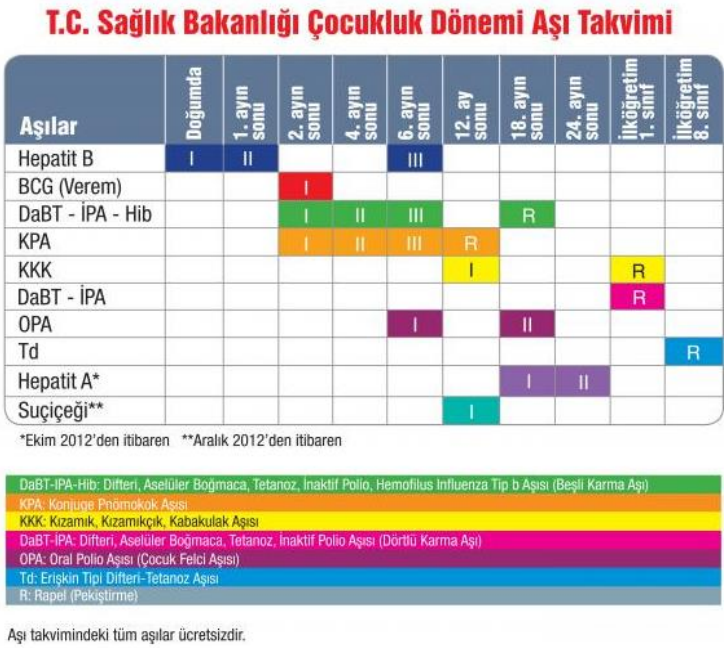
Kullanıcının bebek doğum tarihini gün/ay/yıl olarak girmesi ve hesapla butonuna basması ile birlikte karşısına doğumtarihi girilen bebeğin ilk ayından, ilköretim 8.sınıfına kadar yaptırılması gereken aşılarının adlarını, aşı tarihlerini ve aşı pekiştirme dozlarını içinde barındıran kişiye özel aşı listesi hazırlanacaktır. Bu listede kaç aşı yaptırılacağı, her bir aşının içeriği hakkında bilgileri net olarak görmek mümkündür. Açıklamalar, listenin

en altına yerleştirilmiştir. Ayrıca her bir aşının, kendi içinde renkleri korunarak, takibi şekil 3.8'de görüldüğü gibi görsel olarak da kolaylaştırılmıştır.



Şekil 3.17. Aşı takvimi hesaplama modülü ekran görünümleri

Aşı takvimi hesaplamada esas alınan sağlık bakanlığına ait 2012 yılı için hazırlanan aşı takvimi şekil 3.9'da görülmektedir.[51]

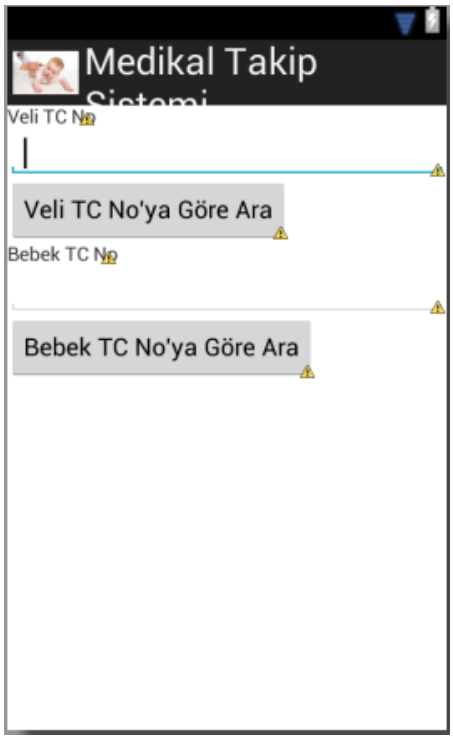


Şekil 3.18. T.C. Sağlık bakanlığı bebek aşı takvimi

Ayrıca kullanıcı isteğine bağlı olarak bilgilerin veritabanına kayıt edilmesi de mümkündür. Android sisteminde veritabanı olarak sqlite veritabanı kullanılmaktadır. Veritabanı için istenen bilgilerin bulunduğu kayitdeneme.xml adlı dosyanın tasarımı şekil 3.10'da gösterilmiştir.

Şekil 3.19. Veritabanı bilgi giriş ekranı

Veritabanına bilgi eklemek için oluşturulan kayitdeneme.xml sayfası ile sqlite veritabanında saglık adında bir veritabanı ve içerisinde bebekveli adlı bir tablo oluşturulup bilgiler buraya kaydedilmektedir. Buradan menu tuşuna basılarak arama yapılmak istenirse bebek tc no veya veli tc no ile şekil 3.11'de görülen ekranda arama yapılabilmektedir.

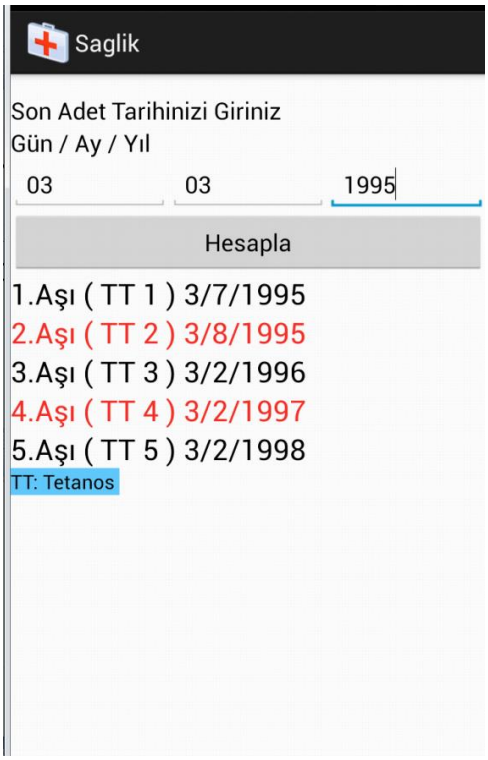


Şekil 3.20. Arama ekranı

Bebek aşı takvimine göre aşıların düzenli ve zamanında yapılması için gereken takibin sistem üzerinde otomatik olarak yapılması hedeflenmiştir.

Yukarıdaki takvim esas alınarak sisteme kaydı yapılan her kullanıcı için aşı tarihlerinin hesaplanması ve aşı zamanı geldiğinde uyarı verilmesi esasına dayanan bir çalışma ortamını içermektedir.

Doğurganlık Çağı denilen 15-49 yaş arasındaki kadınlarda şekil 3.13'de görülen sağlık bakanlığın hazırladığı tetanoz ve difteri aşı takvimi esas alınarak şekil 3.12'de görülen ekranda aşı takvimininin çıkarılması, hasta takibi ve aşı zamanı geldiğinde hatırlatılması temeline dayanmaktadır.[52]



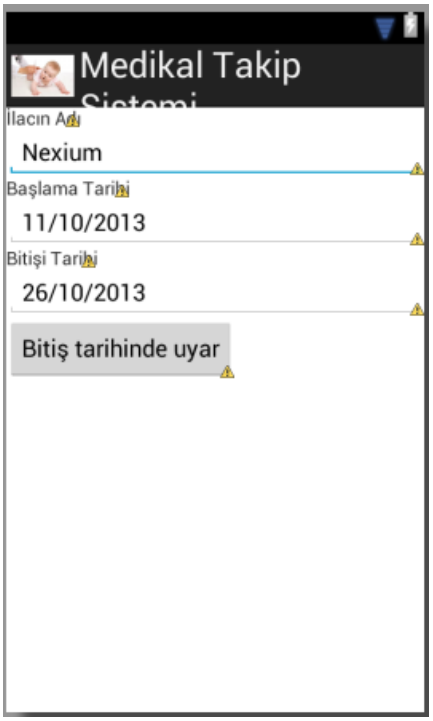
Şekil 3.21. Gebe tetanos aşılarının uygulama zamanlarının listelenmesi

Doğurganlık Çağı (15-49 Yaş)/Gebe Kadınlardaki Tetanoz-difteri Aşı Takvimi

Doz sayısı	Uygulama zamanı	Koruma süresi
Td 1	Gebeliğin 4. ayında - İlk karşılaşmada	Yok
Td 2	Td 1'den en az 4 hafta sonra	1-3 yıl
Td 3	Td 2'den en az 6 ay sonra	5 yıl
Td 4	Td 3'den en az 1 yıl sonra ya da bir sonraki gebelikte	10 yıl
Td 5	Td 4'den en az 1 yıl sonra ya da bir sonraki gebelikte	Doğurganlık çağı boyunca

Şekil 3.22. T.C. Sağlık bakanlığı gebe aşı takvimi

Bu bölümde hastalığı devamlı olan veya belli ilaçlara bağlı yaşayan kişilerin kontrol ve ilaç takibinin, doktor tarafından belirlenen tarihlere göre yapılması düşünülmüştür. Kontrol aralığı ve ilaç kullanma peryotları belirlenen hastanın yönlendirilmesi amacıyla oluşturulmuş bir bölümdür. Bu bölümde ilaç adı,ilaca başlanma tarihi ve ilacın bitiş tarihi girilip ilacın kullanım zamanı bittiğinde yeni ilaç için ilacın adı ve başlangıç tarihi ile beraber uyarı mesajı verilmesi için hazırlanmış şekil 3.14'de görülen ilactakip.xml ekranı hazırlanmıştır.



Şekil 3.23. İlaç takip ekranı

Uygulama geliştirirken veritabanı tasarımı,normalizayonu ve yönetimi için yapılan adımların nasıl gerçekleştirildiğini kısaca ifade etmek için, veritabanı tasarımı aşamasında veli-bebek varlık-ilişki modeli olarak şekil 3.15'de görülen ilişki modeli tasarlanmıştır.

4. SONUÇLAR VE ÖNERİLER

Mobil yazılım alanında çalışmanın hazırlanma aşaması zorlu bir süreci içinde barındırmaktadır. Kaynakların ve yapılan çalışmaların azlığı bu alanın tam şekillenmemiş bir alan olduğunu göstermektedir. Nitekim bilim insanının ve üniversitelerin bilime yön vermek ve katkıda bulunmak gibi önemli görevleri bulunmaktadır. Bu hazırlanma aşamaları içerisinde sağlık, mobil sistemler, yazılım geliştirme ve süreç yönetimi gibi farklı disiplinleri bir araya getirme, bu çalışmanın altyapısını oluşturdu.

Geçmişte kullanılan teknolojilerden, mobil sistemlere geçiş yapmaya iten sebeplerden hız, performans, bakım kalitesi gibi kriterlerin olumlu yönde artış göstermesi evrak maliyeti, evrak karmaşası, bilgiye ulaşılma zorluğu gibi göstergelerinde azalması şimdi bulunduğumuz noktaya gelmemizde önemli adımların atılmasını sağladı[19]. Ayrıca artık eski sistemlerin küçüldüğü gerçeğine dikkat edilecek olunursa, büyük, çok yer kaplayan sistemlerin yerine artık daha küçük, taşınması ve entegre edilmesi daha kolay, farklı platformların günümüzde kullanılmaya başlaması gidilen yolun daha net bir şekilde öngörülebilmesini sağlamaktadır.[31] Bu küçük cep cihazlarının artık daha akıllı ve üretilen uygulamaların daha basit olması ihtiyacı yazılım geliştiricileri, akademisyenleri ve üreticileri hem mobil sistemlere yönlendirmekte, hem de mobil uygulamaların hızlı, basit, güvenilir[30] olmaları için yeni bir yol açma vazifesini de beraberinde getirmektedir. İnsanlar için geliştirilen teknolojilere bakıldığında insanın konforu için geliştirilen teknolojiler daima daha çok tercih edilir ve daha çok pazarlanabilir olmuştur. Günümüz dünyasındaki teknolojilerin ilerlemesi hakkında yapılan gözlemlerde bu nokta rahatlıkla görülebilmektedir.[20]

Mobil sistemler ve akıllı cihazlar üzerine yapılan araştırmalarda akıllı cihazların %74,4'ünün [53] Android işletim sistemini kullandığı tesbit edildi. Bunula beraber sadece cihaz olarak değil artık giyilebilir şekilde farklı teknolojilerin kullanıldığı ürünlere bakıldığında yine android işletim sisteminin sıklıkla kullanıldığı görülmektedir.

Ülkemizde mobil yazılım alanında bir adım atılabilmesi gayesiyle hazırlanan bu çalışma içerisinde sağlık alanında aile hekimliği yapan doktorların, aşı takiplerini yapan hemşirelerin ve bebek sahibi ebeveynlerin ihtiyaçları dikkate alınmış, istekleri gözönüne alınarak uygulama taslağı oluşturulmaya başlanmıştır. Ebeveynlerin aşı zamanlarını bilememeleri, sağlık ocağına gidip gelme noktasında sıkıntı yaşayan, köylerde veya sağlık

ocaklarından uzakta oturan ailelerin ulaşım zorluğu, aşı takiplerini yapan memur ve memurelerin klasik sistemlerle takip yapmaları, doktorların hastalarının aşılarını takip etmedeki zorlukları dikkate alındığında böyle bir yazılıma ihtiyaç duyulduğu görülmektedir.

Toplumuzun bilinçlenmesi adına ebeveynlerin sahip oldukları çocukların hangi dönemde hangi aşıları yaptırmaları gerektiğini öğrenmeleri, gebelerin aşı tarihlerini kendi kendilerine takip edebilmeleri bilinç düzeyini artırıcı olarak görülmüş ve uygulama ile bu bilinçlenme düzeyine katkıda bulunulması da hedeflerden birisi olarak belirlenmiştir.

Bu çalışmada medikal alanda hem aileler hem de aile hekimleri için ihtiyaç olan bebek aşı takiplerinin yapılabilmesi ve izleme sistemi oluşturmak için gereken sistem gereklilikleri, mobil sistem mimarisi, mobil sistemlerde yazılım geliştirme için gerekli bileşenler ve sistem üretim aşamaları anlatılmış ve gerçekleştirilmiştir. Bu alanda oluşturulan sistemle hem aşı takipleri hem de ileride entegre edilebilecek yeni modüllerle sadece bebekler için değil büyük hastalar için de aşı, ilaç, muayene kontrolleri daha detaylı biçimlerde otomatik olarak takip edilebilir olacaktır. Yapılmak istenen sistemin yaygınlaşması bebekler için aşı takiplerinin düzenli olarak aile tarafından yapılabilmesi, gebeliği esnasında kadınların kendi kendilerine aşılarını takip edebilmeleri ve buna ek olarak düzenli ilaç kullanması gereken kronik hastalarda ve ilaç bitiş tarihini takip edemeyen kişilerde büyük bir kolaylık sağlaması sonuçlarını doğuracaktır.

Teknoloji alanındaki gelişmelerle beraber akıllı cihazlarda her ne kadar işlemci hızı, hafıza alanı gibi birimlerde artışlar yaşansa da uygulama alanında güvenlik programları açısından bakıldığında cihaz güvenliği için oluşturulan programlar genel itibariyle kullanıcıya tam olarak sunulamadığı veya cihaz sahipleri tercih etmediği için güvenlik açısından akıllı cihazlar risk altındadır[35]. Ayrıca mobil cihazlar sabit olmamalarından kaynaklanan güvenli bir alan içerisinde devamlı korunamama gibi bir gerçekle de karşı karşıya kalmaktadırlar. Bu noktada kullanıcılara görev düşmektedir. Her ne kadar android işletim sistemi yapı gereği yüklenen bütün programlar için sistemden istenen izinleri, yetki alanı içerisine almak istediği bileşenleri açık bir şekilde kullanıcıdan onay isteyerek gerçekleştirse de çoğu kullanıcı bu izinleri ve programların yapabileceklerini önemsemeden uygulamaları indirip kullanmaktadır. Bu sebeple kullanıcıların akıllı cihazları güvenliğine dikkat ederek kullanmaları büyük önem arz etmektedir.

Sistem geliştirme, yazılım üretme sürecinin teslim aşamasıyla bitmediği asıl önemli olan kısmın teslimden sonra sistemin bakımının ve geliştirilmesi olduğu bilinen bir gerçektir. Bu çalışma özellikle aşı takipleri, gebe takipleri ve periyodik hasta takipleri için kullanıcı ve hekimlerin mobil cihazları sayesinde kolay bir şekilde hatırlama ve izleme olanağı sunmaktadır. Aile hekimleri, veliler, ve hastaların yazılım sayesinde takip etmeleri gereken aşılar, ilaçlar konusunda hem bilgi hem tarih hatırlatma yönleriyle yardımcı olması sayesinde kullanmayı tercih ettikleri görülmüştür. Kullanıcıların tarihleri takip etmesini kolaylaştıran böyle bir sisteme ihtiyaç duymaları göz önüne alınarak hazırlanmıştır. Bu zamana kadar yürütülen hasta defteri veya sabit bilgisayar sistemlerine nazaran ulaşımı daha kolay olan akıllı telefonlar aracılığıyla daha geniş kitlelere daha kolay ulaşım sağlanması ve uyarma sistemi sağlanması bu uygulama ile gerçekleştirilebilmektedir.

KAYNAKLAR

[1] Körpeoğlu, I., 2006, Gezgin Bilgi İşlem, Bilişim Ansiklopedisi, Papatya Yayıncılık,İstanbul.

[2] http://gs.statcounter.com/#mobile_os-ww-monthly-201210-201310-bar, Türkiye Mobil İşletim Sistemleri Kullanım Oranları, 03 Ocak 2014.

[3] http://gs.statcounter.com/#mobile_os-TR-monthly-201210-201310-bar, Dünya Mobil İşletim Sistemleri Kullanım Oranları, 03 Ocak 2014.

[4] Aksoy,F. ,Türkoğlu,İ. , 2013. Mobil Cihazlar Için Sağlik Bilgi Sistemi Geliştirilmesi, International Advanced Technologies Symposium, YTÜ, İstanbul, 30 Ekim-1 Kasım,s.562-565.

[5] Özata,M., Aslan,Dr.Ş., 2004, Klinik Karar Destek Sistemleri ve Örnek Uygulamalar , Kocatepe Tıp Dergisi. Afyon Kocatepe Üniversitesi, 2,11-17.

[6] Acar,M., Altan Akın, N. T., Gökdağ,S.E., Kaya,Z.G.,2014, Benim Dünyam Çocuk Oyunu: Bir Mobil Uygulama, Akademik Bilişim Konferansı ,Mersin.

[7] Sonuç,E., Ortakcı,Y., Elen,A., 2013, Karabük Üniversitesi Bilgi Sistemi Android Uygulaması, Akademik Bilişim Konferansı, Akdeniz Üniversitesi, Antalya.

[8] Aksu, M., Subaşi ,A., Dayak E., Karabulut M., 2005, Üçüncü nesil (3G) gezgin telefonlar için uygulama geliştirme, KSÜ. Fen ve Mühendislik Dergisi, 8(2), 53-61.

[9] http://www.serefakyuz.com/2012/08/android-uygulama-gelitirme-ders-3-intent-sayfa-degistirme.html, İntent kullanımı, 05 Haziran 2013.

[10] Altan Akın N.T., Başaran E., Ceylan N., Güler A., 2014, Malzemeni Söyle Tarifini Al: Bir Mobil Uygulama, Akademik Bilişim Konferansı , Mersin.

[11] Şimşek,M.A., Erdemli,T., Taşdelen,K., 2013, Android Cihazlarda Konum Tespiti ve Aktarılması, Akademik Bilişim Konferansı, Akdeniz Üniversitesi, Antalya.

[12] http://androiduygulama.blogspot.com.tr, Android Uygulama, 25 Mart 2013.

[13] http://www.btsoru.com/questions/556/android-content-provider-bilgi-veri-icerik-saglayc-bileseni-nedir-ne-ise-yarar, Content Provider Bileşeni, 22 Nisan 2013.

[14] https://www.avealabs.com/blog/android-icin-uygulama-gelistirme ortamikurulumu, Androidde Uygulama Geliştirme, 2013.

[15] Bal, V., Bilgi Sistemlerinin Sağlık İsletmeleri Performansına Etkilerinin Veri Zarflama Analizi İle Ölçümü: Türkiye'deki Devlet Hastanelerinde Bir Arastırma, Süleyman Demirel Üniversitesi, Doktora Tezi, 2010.

[16] Bilgen, S., 1998 , "Sağlık Bilgi Sistemleri", Ulusal Enformayon Altyapısı Anaplanına Katkı, Çalısma Belgesi, TÜBİTAK.

[17] http://www.bilgigunlugum.net/android/2android_aktivite.html, Bilgigunlugum, ,2013

[18] Dündar S., Altıntaş V., 2014, Cep Telefonu Değeri Belirlemek için Mobil Uygulama, Akademik Bilişim Konferansı , Mersin,

[19] Faryad, V., Alavi Milani M.M.R., 2011, Mobil İletişim Nesillerin Evrim İncelemesi: 4G'ye kadar, XIII.Akademik Bilişim Konferansı,İnönü Üniversitesi, Malatya,

[20] Güles, H.K., Özata M., 2005, Sağlık Bilisim Sistemleri, Nobel Yayın Dağıtım, Ankara,.

[21] Güzelyazıcı ,Ö., Dönmez ,B., Kurtuluş ,G., Hacıosmanoğlu Ö., 2014, Yeni Yüzyıl Üniversitesinde Mobil Öğrenme,Mersin

[22] http://www.edureka.in/ , Android Programlama, 03 Ocak 2013

[23] http://ilanguage.ca/about/2012/02/, Intent Seçici,23 Şubat 2014

[24] Kamel ,M., Fawzy ,S., El-Bialy ,A., Kandil, A., 2011,Secure Remote Patient Monitoring System, Biomedical Engineering (MECBME), 2011 1st Middle East Conference on,Sharjah, 339-342

[25] http://blog.melihmucuk.com/android-programlama-android-service-kullanimi/, Android de servis kullanımı, 06 Nisan 2014.

[26] Narman, A. E., 2013, Android Programlama, KODLAB Yayın, İstanbul.

[27] Okur, M.R., Salar, H. C., Süral İ., Uça Güneş E.P., Mobil Teknolojilerin Uzaktan Eğitimde Kullanımı ,http://www.mobilogrenme.net /?page_id=66,07 Ağustos 2013.

[28] Öğütmen, N., 2011, Android, KODLAB Yayın , İstanbul

[29] Sağbaş E.A., 2014, Küresel Konumlama Servisi Kullanarak Araç Takibi ve Mobil Cihazlar Arası Haberleşme , Akademik Bilişim Konferansı, Mersin.

[30] Sağıroğlu Ş., Bulut H., 2009, Mobil Ortamlarda Bilgi Ve Haberleşme Güvenliği Üzerine Bir İnceleme, Gazi Üniv. Müh. Mim. Fak. Der., 3, 499-507,

[31] Soy H., Özdemir Ö., Bayrak M., 2012, Gelecek Nesil Mobil Haberleşme Sistemleri: 3G, 4G ve Ötesi, Akademik Bilişim Konferansı, Uşak.

[32] Taç,M., 2011, Android Programlama,Dikeyeksen Yayıncılık,İstanbul.

[33] Topkaya,C., 2013,Content provider uygulama,http://sanirimbuyok.blogspot.com.tr /2011/05/content-provider-ile-telefon-rehberini.html ,06 Temmuz 2013.

[34] Türker, G.F., Kutlu, A., 2014, Araçlarda On Board Diagnostic Sistem ve Mobil Cihaz Uygulamaları, Akademik Bilişim Konferansı ,Mersin.

[35] Türkiye Bilişim Derneği, Mobil Uygulamalarda Güvenlik ,2.Belge Grubu ,Ankara,2009.

[36] Uzman,O., Akıllı Yazılım kısayolları,http://www.akilliyazilim.org/androiddersleri /eclipse-kisayollari.html, 08 Ocak 2014.

[37] Vogel,L., Android Servis kullanımı,http://www.vogella.com/tutorials/Android Services/article.html, 08 Ocak 2014.

[38] Warren, I., Weerasinghe, T., Maddison ,R., Wang ,Y., 2011, OdinTelehealth: A Mobile Service Platform for Telehealth, The 8th International Conference on Mobile Web Information Systems (MobiWIS), Niagara Falls, Ontario, Canada, September 19-21,s. 681-688.

[39] Yüce, Y.K., Bilge U., Saka O., 2005, MATDS: Mobil Astım Takip ve Değerlendirme Sistemi, 2. Ulusal Tıp Bilişimi Kongresi/Medical Informatics Turkey, Belek,Antalya,17-20 Kasım, s.26-32.

[40] https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-201/android-mimarisi-ve-sistem-ozellikleri, Android İşletim Sistemi Mimarisi, 06 Ocak 2014.

[41] http://jayandroidblogs.blogspot.com.tr/2013/07/dalvik-virtual-machine.html, DVM ile JVM'nin Çalışma Biçimleri, 06 Mart 2014.

[42] http://denizkilinc.com/2013/07/08/yazilim-yasam-dongusu-temel-asamalari-software-development-life-cycle-core-processes/, Yazılım Geliştirme Yaşam Döngüsü,

[43] Ocak,Ş.,Yıldıztekin,M.,2011, Güvenli Yazılım Geliştirme Süreç Modellerinin Karşılaştırılması Uygulamaları,Elektrik-Elektronik ve Bilgisayar Sempozyumu, Fırat Üniversitesi, Elazığ, 5-7 Ekim, 338-342.

[44] http://kod5.org/activity-sinifi, Android Aktvite Yaşam Döngüsü,04 Mart 2013.

[45] Türedi,G., 2013, Broadcast Receiver Çalışma Yapısı,http://gturedi.blogspot.com.tr/2014/03/androidservicedenactivitye-deger.html,06 Mart 2013.

[46] Çevik, K.K., Koçer, H.E., 2012, "Mobil Cihaz Tabanlı Yabancı Dilde Kelime Öğrenme Uygulaması", Selçuk Üniversitesi Teknik-Online Dergisi, Cilt 11., Sayı 2., Sayfa 60-70.

[47] Albayrak, Y., Koçer, A., Uslu, S., 2013, Web Servis Aracılığıyla Android Cihazlardan Sıcaklık Kontrolü, Akademik Bilişim Konferansı, Akdeniz Üniversitesi, Antalya.

[48] Tuncer ,S.A., Alkan, A., 2013, Android İşletim Sisteminde Rgb Histogram (Kanal) Değerlerinin Gerçek Zamanlı Olarak Elde Edilmesi, Akademik Bilişim Konferansı, Akdeniz Üniversitesi, Antalya.

[49] Güvensan , M.A., Kansız ,A.O., 2013, Mobil Telefon ile Kaza Tespiti, IEEE 21. Sinyal İşleme ve İletişim Uygulamaları Kurultayı, Girne KKTC.

[50] Mucuk, M.,2013, Android Programlamada Service Kullanımı, http://melihmucuk.com/android-programlama-android-service-kullanimi/,14 Mart 2013.

[51] http://www.oncecocugum.com/asi-takvimi-3/ , Sağlık Bakanlığı Çocukluk Dönemi Aşı Takvimi, 04 Şubat 2014.

[52] http://www.canakkale.hsm.saglik.gov.tr/asi/guncel_asi_takvimi.php, Doğurganlık Çağı Aşı Takvimi, 23 Nisan 2014.

[53] http://www.gartner.com/newsroom/id/2482816, Gartner 2013 Yılı Mayıs Raporu, 24 Nisan 2014

ÖZGEÇMİŞ

Faruk AKSOY, 1981 yılında Adana'da doğdu. 2006 yılında Selçuk Üniversitesi Teknik Eğitim Fakültesi Elektronik-Bilgisayar Eğitimi Bilgisayar Öğretmenliği Bölümünden mezun oldu. 2006-2010 yılları arasında Bilgisayar Öğretmeni olarak görev yaptı. 2010 yılında göreve başladığı Dicle Üniversitesi Ergani Meslek Yüksek Okulu Bilgisayar Programcılığı bölümünde Öğretim Görevlisi olarak çalışmaktadır.