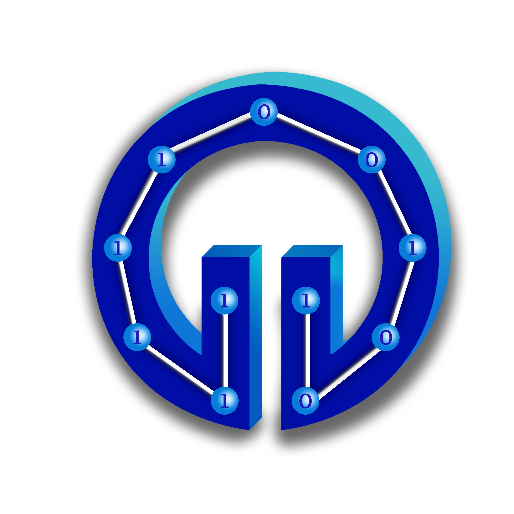
**KARADENİZ TEKNİK ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

****

**AKILLI MARKET OTOMASYONU**

**BİTİRME PROJESİ**

**Emre ÖKSÜZ**

**Barış Bahadır ŞAVUR**

**2018-2019 GÜZ/BAHAR DÖNEMİ**

**KARADENİZ TEKNİK ÜNİVERSİTESİ**

**MÜHENDİSLİK FAKÜLTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**AKILLI MARKET OTOMASYONU**

**BİTİRME PROJESİ**

**Emre ÖKSÜZ**

**Barış Bahadır ŞAVUR**

**Bu projenin teslim edilmesi ve sunulması tarafımca uygundur.**

**Danışman : Doç. Dr. Bekir DİZDAROĞLU ......................**

**2018-2019 GÜZ/BAHAR DÖNEMİ**

|  |  |  |
| --- | --- | --- |
| ieee.jpg | **IEEE Etik Kuralları**  **IEEE Code of Ethics** | **ieee.jpg** |

Mesleğime karşı şahsi sorumluluğumu kabul ederek, hizmet ettiğim toplumlara ve üyelerine en yüksek etik ve mesleki davranışta bulunmaya söz verdiğimi ve aşağıdaki etik kurallarını kabul ettiğimi ifade ederim:

1. Kamu güvenliği, sağlığı ve refahı ile uyumlu kararlar vermenin sorumluluğunu kabul etmek ve kamu veya çevreyi tehdit edebilecek faktörleri derhal açıklamak;
2. Mümkün olabilecek çıkar çatışması, ister gerçekten var olması isterse sadece algı olması, durumlarından kaçınmak. Çıkar çatışması olması durumunda, etkilenen taraflara durumu bildirmek;
3. Mevcut verilere dayalı tahminlerde ve fikir beyan etmelerde gerçekçi ve dürüst olmak;
4. Her türlü rüşveti reddetmek;
5. Mütenasip uygulamalarını ve muhtemel sonuçlarını gözeterek teknoloji anlayışını geliştirmek;
6. Teknik yeterliliklerimizi sürdürmek ve geliştirmek, yeterli eğitim veya tecrübe olması veya işin zorluk sınırları ifade edilmesi durumunda ancak başkaları için teknolojik sorumlulukları üstlenmek;
7. Teknik bir çalışma hakkında yansız bir eleştiri için uğraşmak, eleştiriyi kabul etmek ve eleştiriyi yapmak; hatları kabul etmek ve düzeltmek; diğer katkı sunanların emeklerini ifade etmek;
8. Bütün kişilere adilane davranmak; ırk, din, cinsiyet, yaş, milliyet, cinsi tercih, cinsiyet kimliği, veya cinsiyet ifadesi üzerinden ayırımcılık yapma durumuna girişmemek;
9. Yanlış veya kötü amaçlı eylemler sonucu kimsenin yaralanması, mülklerinin zarar görmesi, itibarlarının veya istihdamlarının zedelenmesi durumlarının oluşmasından kaçınmak;
10. Meslektaşlara ve yardımcı personele mesleki gelişimlerinde yardımcı olmak ve onları desteklemek.

IEEE Yönetim Kurulu tarafından Ağustos 1990’da onaylanmıştır.

**ÖNSÖZ**

Bu projede Karadeniz Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü “Bitirme Projesi” konularına uygun olarak tasarlanan akıllı market otomasyonu yapılmıştır

Birçok alanda benzer işler için kullanılabilecek olan bu projede bir takım sorunların çözümü ve insan hayatının kolaylaştırılması hedeflenmiştir.

Yaşamımız boyunca hep yanımızda olan ailelerimize, gerekli donanımlar için desteklerini esirgemeyen mühendis arkadaşlarımıza ve proje boyunca danışmanlığımızı yapan sayın hocamız Doç. Dr. Bekir DİZDAROĞLU ‘na teşekkürlerimizi sunarız.

Emre ÖKSÜZ

Barış Bahadır ŞAVUR

Trabzon 2018

**İÇİNDEKİLER**

|  |  |
| --- | --- |
|  | Sayfa No |
| IEEE ETİK KURALLARI | II |
| ÖNSÖZ | III |
| İÇİNDEKİLER | IV |
| ÖZET | V |
| 1. GENEL BİLGİLER | 1 |
| * 1. Giriş | 1 |
| * 1. Proje Tanımı | 1 |
| 2. PROJE ARAÇLARI | 2 |
| 2.1.Android Studio | 2 |
| 2.2.Firebase | 3 |
| 3.PROJE TASARIMI | 4 |
| 3.1.Akış Diyagramı | 4 |
| 3.2.Firebase’nin projeye bağlanması | 5 |
| 3.3.Kayıt Ekranı | 5 |
| 3.3.1.Activity\_kayit.xml , fragment\_yönetici.xml , fragment\_müsteri.xml | 5 |
| 3.3.2.Yöneticiler.java ve Müsteriler.java | 7 |
| 3.3.3.KayitActivity.java | 8 |
| 3.4.Login Ekranı | 9 |
| 3.4.1.Activity\_login.xml | 9 |
| 3.4.2.LoginActivty.java | 10 |
| 3.5.Ana Ekran | 11 |
| 3.5.1.Activity\_main.xml | 11 |
| 3.5.2.MainActivity.java | 12 |
| 3.6.Alışveriş Ekranı | 13 |
| 3.6.1.Activity\_sepet.xml | 13 |
| 3.6.2.SepetActivity.java | 14 |
| 3.6.3.Activity\_qrscanner.xml | 15 |
| 3.7.Yönetici Ekranı | 17 |
| 3.7.1.Activity\_yonetici.xml | 17 |
| 3.7.2.YoneticiActivity.java | 18 |
| 3.7.3.Activity\_urun\_ekle.xml | 19 |
| 3.7.4.UrunEkleActivity.java | 20 |
| 4.ÖNERİLER | 21 |
| 5. KAYNAKLAR | 22 |
| 6.STANDARTLAR ve KISITLAR FORMU | 23 |

**ÖZET**

Günümüzde marketlerin sahip olduğu çalışanlar , kısıtlı açık olma zamanları , alış-veriş sırasında ki aşırı zaman kaybı marketin kazancını arttırmakta engel olan etkenlerdir.Bu durumda iş veren kişinin sürekli ilişki içinde bulunması gerektiği çalışanlar ile ilgilenmemesi çalışanlar üzerinde negatif bir etki yaratmaktadır.Bu projenin yapılmasında globalleşen dünyamızda kazancı arttırmak da engel olan etkenleri azaltıp daha kolay bir alış-veriş imkanı sürmeyi amaçlamıştır.Geleneksel market anlayışını pozitif yönde değiştirip alış-verişi daha kısa zamanda daha güvenli ve daha kolay şekilde yapmayı hedeflemiştir.

Projemizde Android Studio’da Java programlama dili ile geliştirilmiştir.Projemiz de oluşturulan arayüzleri her adımda projeyi oluşturan kişiler tarafından farklı mobil cihazlar da test edilmiştir.Bununla birlikte marketimiz de bulunan alış-veriş malzemelerini kontrol altına almak için Google firmasının geliştirdiği Firebase veritabanı kullanılmıştır.

**1.GENEL BİLGİLER**

**1.1.Giriş**

Ürün, fiyat, yer, promosyon alışılmış market anlayışınının çoğu kısmını tanımlamış oluyor.Durum böyle olunca, işletmenin elindeki hiçbir işe yaramayan ürün, ısrarcı reklam politikalarıyla tüketiciye sunuluyor. Tüketiciler de haliyle işe yaramaz, kalitesiz, özensiz ürüne gerekli tavrı koyarak ürünün pazarda tutunamamasının sebebi oluyor. Pazarlamacıların en temel amacı şirket için yeni fırsatlar araştırmak ve Pazar bölümlendirerek, hedeflemeyi ve konumlandırmayı işletmeyi doğru alana sevketmek için dikkatlice kullanmalarıdır.Aynı zamanda işletmenin bünyesinde barındırdığı çalışanlardan sorumlu olması , onların çalışma potansiyellerini sürekli kontrol etmesi şirketin gelişmeşinde bir engel teşkil ediyor.Bu proje alışılmış market ve pazarlama anlayışından farklı bir davranış sergiliyor.

**1.2.Proje Tanımı**

Projemizi betimlemek gerekirse , bir kullanıcı kendi mobil cihazına indirdiği uygulamamızı kullanmaya başlamadan önce bir kayıt yaptırması gerekiyor.Kayıt sırasında TC kimliği,ismi,soyismi,e-maili bilgileri alınarak unique olacak şekilde bir QR kod atanıyor.

Bu kod kullanıcının markete giriş sırasında kullancağı koddur.Kullanıcı kayıt olduktan sonra giriş ekranında kendi QR kodu bulunacaktır ardından alış-veriş yapacağı markete geldikten sonra markete giriş esnasında cihazı ile alışverişe başlalayabilir

Şekil 1.2.1 Alışveriş sırasında QR kodu taramak

O sırada müşterinin mobil cihazında bir alış-veriş arayüzü açılacaktır.Kişi alış-verişi sırasında istediği ürünü raftan alma sırasında yapması gereken tek şey ürünün kendine ait QR kodu kişinin telefonunun kamerası ile okutmak ardından ürünü kendi market sepetine koyabilir.Bu sırada marketin veri tabanından alınan ürünün stoğundan bir eksilecektir.Ürünü alan kişinin telefonunda ise o ürünün ismi eklenecektir.Kişi alacağı her ürün için yapması gereken tek şey ürünün QR kodunu okutmak.Alış-verişi tamamladıktan sonra kişi telefonuna ekleyebileceği kredi kartı bilgileri ile alış-veriş ücretini ödeyebilecektir.

Projenin ,normal marketlere göre avantajlı durumlar :

1.Herhangi bir kasiyere gerek duyulmaması

2.Marketin esnek çalışma saatleri

3.Marketin kendi bünyesinde ki yöneticilerin istediği yerden stokları kontrol edebilmesi

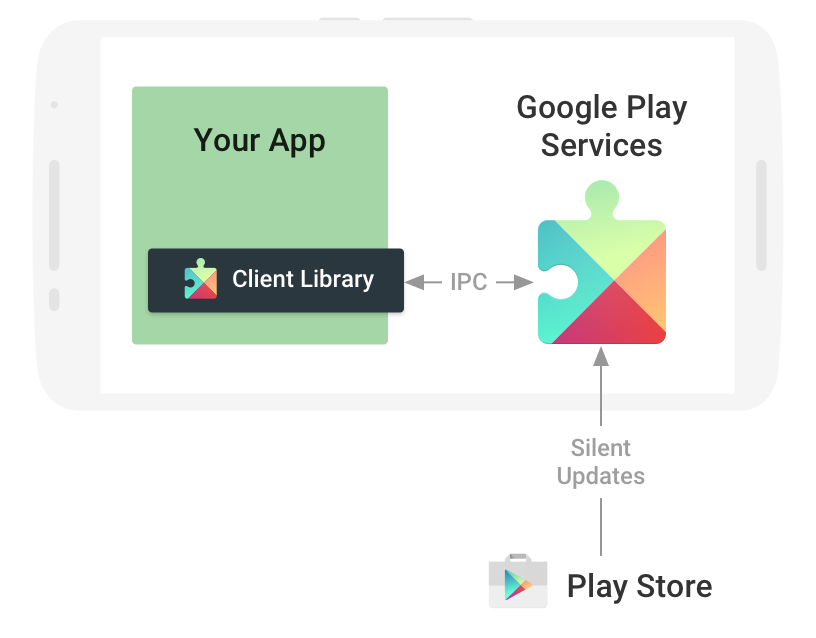
4.Müşterinin yaptığı toplam alış-veriş ücretini anlık sepetinde görebilmesi.

5.Hızlı ve kolay alış-veriş imkanı.

6.Mobil uygulamanın basit bir arayüze sahip olması

**2.PROJE ARAÇLARI**

Projemiz bir android uygulaması olduğu için android studio kullandık.Verilerimizi saklamak için ise Google firmasına ait olan Firebase veritabanını kullandık.

 **2.1.Android Studio**

Şekil 2.1.1 Android Studio algorithm

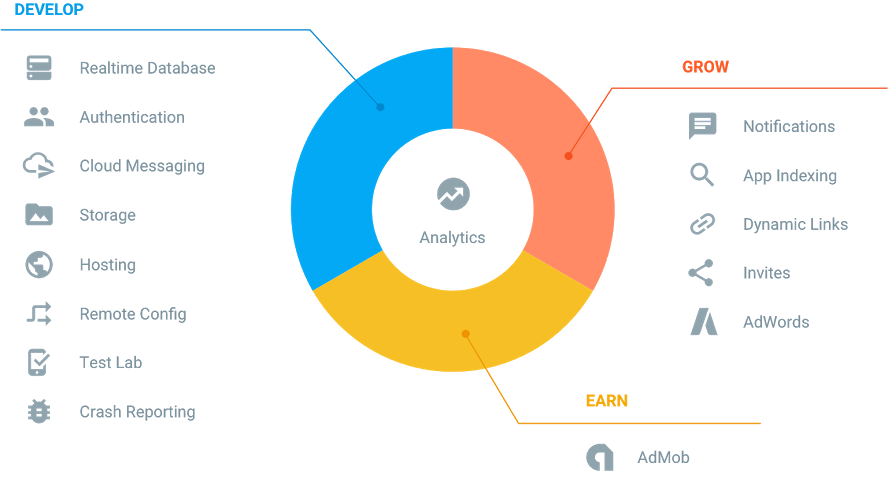
Android Studio, Android için resmi tümleşik geliştirme ortamı. 16 Mayıs 2013 tarihinde Google I/O etkinliğinde tanıtılmıştır. Android Studio, IntelliJ IDEA’ya dayalı olup Android geliştirme için özel olarak tasarlanmıştır.

Android Studio , diğer android programları geliştirme(Eclipse) ortamlarına göre artıları:

* JetBrains Intellij Idea üzerine kurulmuş (Java geliştirmede en rahat IDE’lerden biri. Hatta en iyisi diyebilirim. Araştırmalar da bu görüşümü destekliyor sanırım)
* Google tamamen bir Android geliştirme ortamı sağlamış. IDE içerisinden Google kendi kütüphanelerini kolayca kullanmamızı sağlıyor.
* SDK yönetimi içerisinde gömülü olarak geldiği için dışardan herhangi bir SDK Manager ile uğraşmanıza gerek kalmıyor.
* IDE sayesinde Refactor, Reflection vb. işlemleri çok rahat yapabiliyorsunuz.
* Kod yazımı çok rahat. IntelliSense isterseniz hızlandırabiliyor ve her şeyi tamamlar hale getirebiliyorsunuz.
* Debugger ve Publish çok rahat, Gradle içerisinde gerekli konfigürasyonları yaparak her build esnasında build sistemini yeniden çalıştırmayıp (Gradle’ı daemon olarak çalıştırmak) işlemleri hızlandırabilirsiniz.
* Logcat sayesinde olup biten her şeyi (hataları,uyarıları) çok rahat izleyebiliyorsunuz.

**2.2.Firebase**

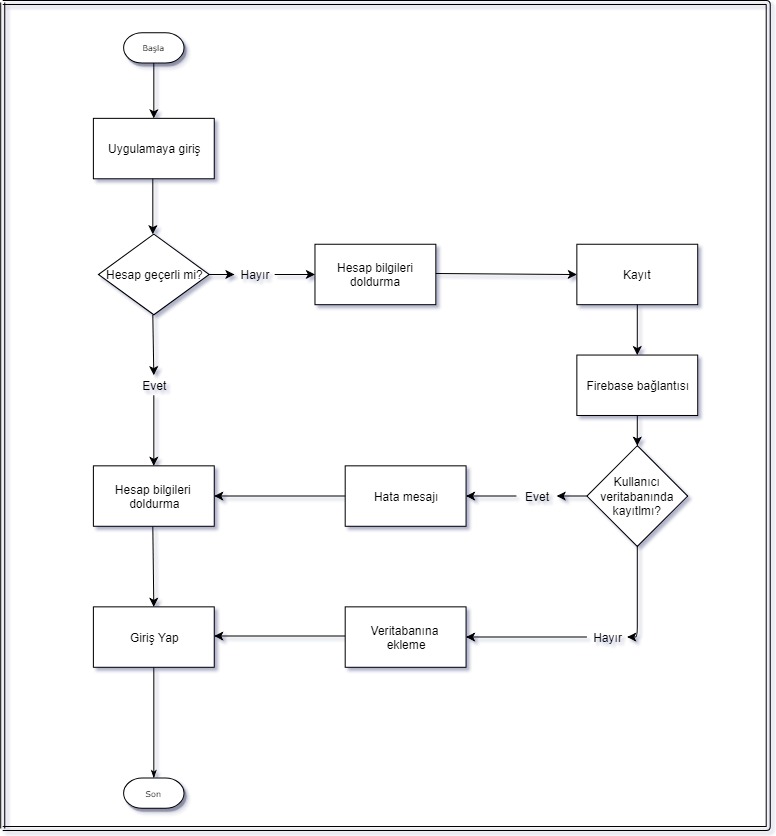
Uygulama yönetim, kullanım takip, veri depolama, bildirim gönderme gibi işlemleri,extra bir sunucuya ve sunucu taraflı kod yazmaya gerek kalmadan halleden bir platformdur.

Günümüzde uygulamalar platform farketmeksizin aynı veriye her cihazdan erişmek istiyor. Uygulaması, birçok kullanıcı tarafından yüklenen geliştiricilerin de kayıt – oturum bilgilerini tutma, uygulamaların kullanım verilerini analiz etme, yeni duyurular yapmak için aynı zamanda kullanıcıya bildirim gönderme, uygulamayı test etme gibi işlemleri rahatlıkla yönetebileceği bir yönetim paneli gerekiyor. İşte Google tarafından yeni özelliklerin eklenmesiyle sürekli kendini geliştiren Firebase, bütün bu ihtiyaçları karşılayabilmek için uygulama geliştiricilerine ücretsiz kullanım da sunan bir platformdur.

Şekil 2.2.1 Firebase Analitiği

**3.PROJE TASARIMI**

**3.1 Akış diyagramı**



**3.2 Firebase’nin projeye bağlanılması**

Projemizin Android>Gradle Scripts>build.gradle dosyası içerisine öncelikle gerekli kütüphanleri ekledik.Ardından gerekle gradle versiyonlarını güncelledik.

İmplementation ‘com.google.firebase:firebase-database:16.0.2’

implementation ‘com.google.firebase:firebase-core:16.0.3’

Projemizde tools>Firebase sekmesinden realtime database’i seçtikten sonra connect your app to firebase ile projemize firebaseyi bağlamış oluyoruz.Gerçek zamanlı çalıştığını görmek için de aynı sekmede bize örnekler veriyor.

**3.3 Kayıt Ekranı**

**3.3.1 Activity\_kayit.xml , fragment\_yönetici.xml , fragment\_müsteri.xml**

Projede öncelikle kayıt ekranı oluşturarak başlamaya karar verdik.Activity\_kayit.xml ile sekmeleri tanımladık.Activtymizde ilk önce bir tabloyut tanımladık ardından tablayout içerisine iki adet tabItem tanımladık.TabItem’lardan biri müşteri kaydını diğeri ise yönetici kaydını temsil ediyor.

<**android.support.design.widget.TabLayout  
 android:id=”@+id/tabLayout”**

**android:layout\_width=”0dp”  
 android:layout\_height=”rap\_content”  
 app:layout\_constraintEnd\_toEndOf=”parent”  
 app:layout\_constraintStart\_toStartOf=”parent”  
 app:layout\_constraintTop\_toTopOf=”parent”  
 app:tabMode=”fixed”**>  
  
 <**android.support.design.widget.TabItem  
 android:id=”@+id/MüsteriTab”  
 android:layout\_width=”rap\_content”  
 android:layout\_height=”rap\_content”  
 android:text=”Müşteri Kaydı”** />  
  
 <**android.support.design.widget.TabItem  
 android:id=”@+id/YöneticiTab”  
 android:layout\_width=”rap\_content”  
 android:layout\_height=”rap\_content”  
 android:text=”Yönetici Kaydı”** />  
  
</**android.support.design.widget.TabLayout**>

Daha sonra tasarlayacağımız iki adet fragmentimizi sekmelerde gösterebilmemiz için bir tane viewpager tanımlıyoruz.

<**android.support.v4.view.ViewPager  
 android:id=”@+id/viewpager”  
 android:layout\_width=”0dp”  
 android:layout\_height=”0dp”  
 app:layout\_constraintBottom\_toBottomOf=”parent”  
 app:layout\_constraintStart\_toStartOf=”parent”  
 app:layout\_constraintTop\_toBottomOf=”@+id/tabLayout”** />



Fragment\_yönetici.xml dosyamızda ise yönetici kayıt edecek edittextler ve kayıt butonu bulunuyor.Yönetici kayıt olurken kişiniz Tc kimlik numarası , adı , soyadı , şifre , e-mail ve şirket isimlerinin yazılabileceği edittextler oluşturduk sonrasında gerekli bilgileri doldurduktan sonra firebase veritabanımıza kayıt olabilmesi için kaydet butonu oluşturduk.Her edittext’in sadece kendine ait ID’si bulunmakta , her edittextin yazı boyutları ,yazı renkleri vs. aynı yaptık.

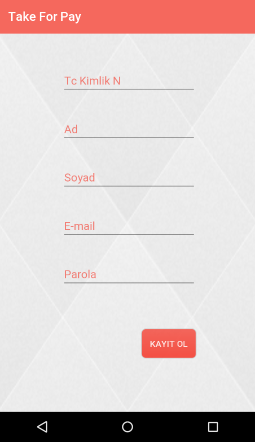
**Android:textColor=”#aaF44336”  
android:textColorLink=”#CCF44336”**

**android:layout\_width=”rap\_content”  
android:layout\_height=”rap\_content”  
android:layout\_marginTop=”32dp”  
android:layout\_marginEnd=”96dp”**

Fragment\_müsteri.xml dosyasında ise aynı şekilde kişinin Tc Kimlik numarasını,adı soyadı,şifre ve e-mail bilgilerini almak için edittextler ve firebase veritabanına eklememiz için bir adet kaydet butonu bulunuyor.Edittextlerin yazı rengi boyutu vs. için yönetici fragmentinden olduğunu gibi hepsinde aynı tanımladık.

**Android:textColor=”#aaF44336”  
android:textColorLink=”#CCF44336”**

**android:layout\_width=”rap\_content”  
android:layout\_height=”rap\_content”  
android:layout\_marginTop=”32dp”  
android:layout\_marginEnd=”96dp”**



**3.3.2 Yöneticiler.java ve Müsteriler.java**

Yöneticiler.java ve Müsteriler.java da yönetici ve müşterilerin verilerini tanımlamamız gerekiyor.Bu sınıflarda öncelikle private olarak kimlik , ad , soyad vs. string olarak tanımladık.Ardından constructor oluşturduk.

**Public** Yöneticiler(String ykimlik, String yad, String ysoyad, String yemail, String yparola, String ysirketAdi) {  
 **Ykimlik** = ykimlik;  
 **Adı** = yad;  
 **Soyadı** = ysoyad;  
 **Yemail** = yemail;  
 **Yparola** = yparola;  
 **YsirketAdi** = ysirketAdi;  
}

**public** Müsteriler(String mkimlik, String mad, String msoyad, String memail, String mparola) {  
 **Mkimlik** = mkimlik;  
 **Adı** = mad;  
 **Soyadı** = msoyad;  
 **Memail** = memail;  
 **Mparola** = mparola;  
}

Constructor tanımladıktan sonra bu verileri kullanabilmemiz için getter ve setter fonksiyonlarını da tanımlamız gerekiyor.Herbir değişken için ayrı getter ve setter fonksiyon tanımladık.Getter ve setter fonksiyonları public fonksiyonlardır ve getter fonksiyonları geri dönüş yapabilen fonksiyonlardır setter fonksiyonlarda void fonksiyonlardır.

**Public** String getMkimlik() {**return Mkimlik**;}

**public void** setMkimlik(String mkimlik) {**Mkimlik** = mkimlik;}

**public** String getAdı() {**return Adı**;}  
  
**public void** setAdı(String adı) {**Adı** = adı;}  
  
**public** String getSoyadı() {**return Soyadı**;}  
  
**public void** setSoyadı(String soyadı) {**Soyadı** = soyadı;}  
  
**public** String getMemail() {**return Memail**;}  
  
**public void** setMemail(String memail) {**Memail** = memail;}  
  
**public** String getMparola() {**return Mparola**;}  
  
**public void** setMparola(String mparola) {**Mparola** = mparola;}

**3.3.3.KayitActivity.java**

KayitActivity sınıfında activity\_kayit.xml dosyasında ki oluşturduğumuz edittext ve butonları aktif kullanmamız için gerekli kodları yazmamız gerekiyor.Öncelikle edittext ,tabLayout ve ViewPager değişkenleri oluşturduk.Önceden oluşturduğumuz fragmentleri kullanabilmemiz için adapter tanımladık , viewpager sayesinde fragmentleri sekmelerde görünmesini sağladık.

MyAdapter adapter=**new** MyAdapter(getSupportFragmentManager());  
  
adapter.fragmentEkle(**new** rapic\_fragment(),**”Müşteri Kaydı”**);  
adapter.fragmentEkle(**new** rapici\_fragment(),**”Yönetici Kaydı”**);  
**viewpager**.setAdapter(adapter);  
**tab**.setupWithViewPager(**viewpager**);

MüsteriKaydet fonksiyonu oluşturup xml dosyasında oluşturduğumuz butonları ve edittextleri kullanabilmemiz için burada oluşturdupumuz değişkenlere bağdaşlaştırıyoruz.

**MTC**=(EditText) findViewById(R.id.***msteriTcText***);  
**MAD**=(EditText)findViewById(R.id.***msteriAdText***);  
**MSOYAD**=(EditText)findViewById(R.id.***msteriSoyadText***);  
**MEMAİL**=(EditText)findViewById(R.id.***msteriEmailText***);  
**MPAROLA**=(EditText)findViewById(R.id.***msteriSifreText***);  
**Mkaydi**=(Button)findViewById(R.id.***KaydetBTN***);

Kayıt ekranında bulunan butona setOnClickListener ile tıklandığında çalışacağı kodları ekliyoruz.Öncelikle edittext de ki değeleri stringe çeviriyoruz.

String mtc = **MTC**.getText().toString();  
String mad=**MAD**.getText().toString();  
String msoyad=**MSOYAD**.getText().toString();  
String memail=**MEMAİL**.getText().toString();  
String mparola=**MPAROLA**.getText().toString();

Kayıt ekranında ki butona tıkladığımızda veritabanımıza değerleri ekleyebilmemiz için öncelikle Firebase veritabanına bağlanmamız gerekiyor.Bunun için öncelikle gerekli kodları ekledik.Burada müşteriler sekmesi için ekleme yapılıyor , yönetici için eklemede de aynı şekilde oluşturuluyor.

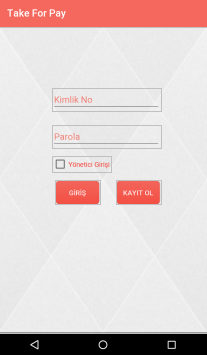
FirebaseDatabase database=FirebaseDatabase.*getInstance*();  
DatabaseReference MüsterilerRef=database.getReference(**“Müsteriler/”**+mtc+**”/”**+mparola+**”/”**+mad+**” “**+msoyad);  
Müsteriler rapic=**new** Müsteriler(mtc,mad,msoyad,memail,mparola);

Gerekli kodları ekledikten sonra MüsterilerRef değişkenini kullanarak veritabanımıza değerlerimizi ekliyoruz.

MüsterilerRef.child(**“Tc Kimlik”**).setValue(mtc);  
MüsterilerRef.child(**“Adı”**).setValue(mad);  
MüsterilerRef.child(**“Soyadı”**).setValue(msoyad);  
MüsterilerRef.child(**“E-mail”**).setValue(memail);  
MüsterilerRef.child(**“Şifre”**).setValue(mparola);

**3.4.Login Ekranı**

**3.4.1.Activity\_login.xml**

Login ekranımızda kullanıcı kimlik numarası ve şifresini girerek uygulamaya giriş yapacaktır.Burada kimlik numarasını ve şifresini veritabanın da olup olmamasını butona tıkladıktan sonra veritabanında arıyoruz.Activity\_login.xml de Tc kimlik ve şifre için edittext, yönetici girişi checkbox , giriş yapmak için buton ve sekmelerin bulunduğu kayıt ekranına gitmek için de kayıt ol butonu bulunmaktadır.

Kodumuzda tanımladığımız edittextlerin yazı renkleri , boyutları vs. aynı.Butonlarda ise background olarak ikiside aynı özellikleri taşımaktadır.Kimlik numarası Türkiye de 11 haneli olduğu için maxLength 11 olarak sınırlandırdık.Bütün argümanların kendine ait Id’leri de tanımladık.

<**EditText  
 android:layout\_width=”rap\_content”  
 android:layout\_height=”rap\_content”  
 android:layout\_marginTop=”28dp”  
 android:layout\_marginEnd=”92dp”  
 android:ems=”10”  
 android:inputType=”text”  
 android:textColor=”#aaF44336”  
 android:textColorLink=”#CCF44336”**/>

Butonlarda ise width ve height değerlerini rap\_content olarak tanımladım.Yazı renklerini de beyaz renk olarak ayarladık.

<**Button  
 android:layout\_width=”rap\_content”  
 android:layout\_height=”rap\_content”  
 android:layout\_marginTop=”16dp”  
 android:layout\_marginEnd=”32dp”  
 android:background=”@drawable/button\_radius”  
 android:textColor=”#DDFFFFFF”**/>

**3.4.2.LoginActivity.java**

Activty\_login.xml dosyasında oluşturduğumuz argümanları kullanabilmemiz için ve onlar üzerinde işlem yapabilmemiz için gerekli kodları yazmamız gerekiyor.Öncelikle button,edittext ,checkbox vs. değişkenleri oluşturduk.Ardından bunları xml dosyasında ki argümanlarla bağdaştırmak için kodları yazdık.

**Kullanıcı**=findViewById(R.id.***kullaniciText***);  
**parola**=findViewById(R.id.***parolaText***);  
**btnGiris**=findViewById(R.id.***girisbtn***);  
**check**=findViewById(R.id.***YöneticicheckBox***);

Butona tıkladığımızda olacak olaylar için setOnClickListener metodunu kullanıyoruz.Onun içerisinde onClick fonksiyonu oluşturuyoruz.Edittext de yazılan değerleri kullananabilmemiz için öncelikle stringe çeviriyoruz.

**Final** String kullanıcıText=**kullanıcı**.getText().toString();  
**final** String parolaText=**parola**.getText().toString();

Firebase veritabanında ki değerlerimize ulaşıp işlem yapacağımız için gerekli kodları ekliyoruz.

**Final** FirebaseDatabase database=FirebaseDatabase.*getInstance*();  
**final** DatabaseReference YöneticilerRef=database.getReference(**“Yöneticiler/”**+kullanıcıText);  
**final** DatabaseReference MüsterilerRef=database.getReference(**“Müsteriler/”**+kullanıcıText);

Veritabanına bağlandıktan sonra değerlere ulaşmamız için MüsterilerRef veya YöneticilerRef değişkenini kullanarak addValueEventListener metodu oluşturuyoruz.OnDataChange fonksiyonu veritabanında ki herbir değeri kontrol ediyor.

**Public void** onDataChange(@NonNull DataSnapshot dataSnapshot)

DataSnapshot değişkeni kullanarak değerlere ulaşıp edittext de yazılan değer ile aynı olup olmamasını kontrol ediyoruz.Burada aynı şey şifre içinde geçerlidir.Checkbox da yönetici olup olmadığını kontrol etmemize yarıyor.

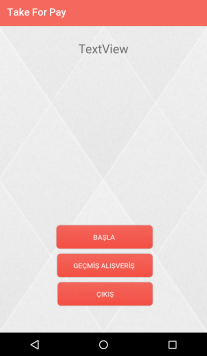
**İf**((kimlik2.equals(kullanıcıText) && şifre2.equals(parolaText) && **check**.isChecked()!=**true**))

Eğer if doğru çalıştıysa kullanıcı yeni bir ekrana yani MainActivity.java ekranına ulaşıyor.Eğer if doğru çalışmazsa ekrana “Hatalı kullanıcı veya şifre girdiniz” mesajı görünüyor.

Intent yeniintent = **new** Intent(LoginActivity.**this**, MainActivity.**class**);  
  
startActivity(yeniintent);

**3.5.Ana Ekran**

**3.5.1.Activity\_main.xml**

Kullanıcı giriş yaptıktan sonra gelen ekranın üst kısmında kişinin adı soyadı için textView sonrasında gelecek olan kişiye özgü QR kod için linearLayout içerisinde imageView, alışverişe başlaması için başla butonu ve hesaptan çıkması için ise çıkış butonu bulunmaktadır.

LinearLayoutumuzun width ve height değerlerini eşit yapıp kare olmasını sağladık.Ardından içerisine QR kod görünmesi için ImageView tanımladık.ImageView’in width ve height değerlerini match\_parent yaptık.

<**LinearLayout  
 android:id=”@+id/LinearLayout”  
 android:layout\_width=”300dp”  
 android:layout\_height=”300dp”  
 android:layout\_marginStart=”8dp”  
 android:layout\_marginTop=”40dp”  
 android:layout\_marginEnd=”8dp”  
 android:orientation=”horizontal”  
 app:layout\_constraintEnd\_toEndOf=”parent”  
 app:layout\_constraintStart\_toStartOf=”parent”  
 app:layout\_constraintTop\_toBottomOf=”@+id/AdSoyadTextView”**>  
  
 <**ImageView  
 android:id=”@+id/imageVİEW”  
 android:layout\_width=”match\_parent”  
 android:layout\_height=”match\_parent”** />  
</**LinearLayout**>

Ana ekranımız da bulunan butonlarda diğer sayfalarda ki butonlarla aynı olacak şekilde background tanımladık.Boyutları , yazı rengi ve yazı boyutları da diğer butonlar ile aynı şekildedir.

**3.5.2.MainActivity.java**

Xml dosyamız da tanımladığımız argümanları kullanabilmemiz için öncelikle java dosyamız da değişkenler tanımladık.Argümanların ID’lerini kullanarak tanımladığımız değişkenlere bağdaşlaştırdık.

**Name**=findViewById(R.id.***AdSoyadTextView***);  
**btnCikis** =findViewById(R.id.***CikisBTN***);  
**denemebuton**=findViewById(R.id.***AyarlarBTN***);  
**imageee**=findViewById(R.id.***imageVİEW***);

QR oluşturmada firebase veritabanımızda kullanıcı yapan kişinin değerlerine erişmemiz için gerekli kodları ekledik.Login sayfasından giriş de kullandığımız kimlik numarası ve şifreyi LoginActivty’den getString ile getirdik.Bu değerleri de veritabanımızda kişinin diğer değerlerine ulaşmak için kullandık.

**Final** String gelenisim=getIntent().getStringExtra(**“isim”**);  
**final** String gelenp=getIntent().getStringExtra(**“parola”**);

**final** FirebaseDatabase database=FirebaseDatabase.*getInstance*();  
**final** DatabaseReference YöneticilerRef=database.getReference(**“Yöneticiler/”**+gelenisim+**”/”**+gelenp);  
**final** DatabaseReference MüsterilerRef=database.getReference(**“Müsteriler/”**+gelenisim+**”/”**+gelenp);

AddValueEventListener metodu içerisinde onDataChange fonksiyonu oluşturduk.GetValue ile kişinin bütün değerlerine ulaşıyoruz.Yöneticiler.java veya müşteriler.java da tanımladığımız get metodları ile verileri alabiliyoruz.

**Y** = d.getValue(Yöneticiler.**class**);  
String kod = **y**.getAdı();  
String kod2 = **y**.getSoyadı();  
String email = **y**.getYemail();  
**Name**.setText(kod+**” “**+kod2);

QR oluşturmak için kişinin e-maili kullandık.QR kod için öncelikle MultiFormatWriter nesnesi oluşturduk.String değerimizi encode ile bitmatrix nesnesine dönüştürdük.Ardından barcodeEncoder nesnesini kullanarak bitmap nesnesi oluşturduk.Kodumuzun boyutlarını 300,300 şeklinde ayarladık.Oluşturduğumuz qr kodumuzu da tanımladığımız imageViewe setledik.Müşteriler için de yöneticiler içinde aynı kodları yazdık.İkisinin de qr oluşturulmasında e-mail değerlerini kullandık.

MultiFormatWriter multiFormatWriter=**new** MultiFormatWriter();  
**try** {  
 BitMatrix bitMatrix=multiFormatWriter.encode(email, BarcodeFormat.***QR\_CODE***,300,300);  
 BarcodeEncoder barcodeEncoder=**new** BarcodeEncoder();  
 Bitmap bitmap=barcodeEncoder.createBitmap(bitMatrix);  
 **imageee**.setImageBitmap(bitmap);  
} **catch** (WriterException e) {  
 e.printStackTrace();  
}

**3.6.Alışveriş Ekranı**

**3.6.1.Activity\_sepet.xml**

Kullanıcı ana ekrana bağlandıktan sonra başla butonuna basınca sepet ekranı açılıyor.Sepet ekranında baş ksıımda TableLayout var alt tarafta iki adet buton var.Butonların bir tanesi Qr kod tarama ekranını yani kamerayı açıyor diğeri ise alışveriş tamamlandıktan sonra ödeme ekranına yönlendiriyor.

Butonları tanımlarken diğer sayfalarda ki gibi renklerini , boyutlarını ,width ve height değerlerini gibi tanımladık.TableLayout ise önce tagını oluşturup içerisine row(satır) tanımladık.Satırın yüksekliğini 28dp ayarladık.Genişliğini ise içerisinde kolon görevi görecek herbir başlığın uzunluğu kadar tanımladık.Oluşturduğumuz TabloRow tagını, satırda yazacağımız kolon başlıklarını tanımlamamız için TextView taglarını kullandık.Her textview’in içerisine android:text ile temsil edeceği kolon ismini yazdık.TextView’lerin ID’leri ise temsil ettiği isime benzer isimler verdik.Kolon başlıklarının belirgin olması için de background renklerine aynı tanımladık.

<**TableLayout  
 android:id=”@+id/TableL”  
 android:layout\_width=”1300px”  
 android:layout\_height=”350dp”**>  
 <**TableRow  
 android:id=”@+id/TableRow”  
 android:layout\_width=”match\_parent”  
 android:layout\_height=”match\_parent”**>  
  
 <**TextView  
 android:id=”@+id/ÜrünSeriKodu”  
 android:layout\_width=”115dp”  
 android:layout\_height=”28dp”  
 android:background=”#CCF44336”  
 android:text=”Ürün Serikodu”** />  
  
 <**TextView  
 android:id=”@+id/textView10”  
 android:layout\_width=”550px”  
 android:layout\_height=”28dp”  
 android:background=”#CCF44336”  
 android:text=”Ürün Adı”** />  
  
 <**TextView  
 android:id=”@+id/textView11”  
 android:layout\_width=”200px”  
 android:layout\_height=”28dp”  
 android:background=”#CCF44336”  
 android:text=”Fiyat”** />  
  
 <**TextView  
 android:id=”@+id/textView12”  
 android:layout\_width=”200px”  
 android:layout\_height=”28dp”  
 android:background=”#CCF44336”  
 android:text=”Adet”** />  
  
 </**TableRow**></**TableLayout**>

**3.6.2.SepetActivity.java**

Activity\_sepet.xml dosyasında tanımladığımız argümanları kullanabilmek için öncelikle public değişkenler tanımladık.Ardından tanımladığımız değişkenleri argümanlar ile bağdaşlaştırmak için findViewById kullandık.

**Scanner**=(Button)findViewById(R.id.***ALBTN***);  
**ödeme**=(Button)findViewById(R.id.***ÖdeBTN***);  
*text*=(TextView)findViewById(R.id.***QRTXT***);  
*para*=(TextView)findViewById(R.id.***ToplamTutarTXT***);  
*tableLayout* = (TableLayout) findViewById(R.id.***TableL***);

Butonlara tıklandığında başka sayfalar açılması için scanner değişkenini setOnClickListener ile kullanarak tanımladık.Aynı kullanım ödeme değişkeni için de geçerlidir.Tarama yapmak için QRScannerActivity.java dosyasını , ödeme yapmak için de OdemeActivity.java dosyası tanımladık.

**Scanner**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(**final** View v) {  
 startActivity(**new** Intent(getApplicationContext(),QRScannerActivity.**class**));  
 }  
});

**3.6.3.Activity\_qrscanner.xml**

Alışveriş sırasında kullanıcı kamerayı aç butonuna bastığında activity\_qrscanner.xml ekranı açılacaktır.Bu ekranla birlikte QR kod okuyabilen kamera açılacaktır.Bunun için öncelikle build.grandle içerisinde barcode okumamız için gerekli olan kütüphaneleri ekledik.Sonrasında xml dosyası içerisinde kamera açılacağından dolayı hiçbir argüman oluşturmadık.Kameranın çalışmasını java kodları ile kontrol ettik.

İmplementation **‘me.dm7.barcodescanner:zxing:1.9’**implementation **‘com.google.zxing:core:3.2.1’**implementation **‘com.journeyapps:zxing-android-embedded:3.2.0@aar’**implementation **‘info.androidhive:barcode-reader:1.1.5’**

**3.6.4.QRScannerActivity.java**

Kullanıcı kamerayı aç butonuna bastıktan sonra ürünün QR kodunu okuttuktan sonra yapılacak olan işlemleri QRScannerActivity.java’ya yazılacak kodlar yapacaktır.Öncelikle QR kod için gerekli olan ZxingScannerView nesne tanımladık.Sonrasında okunulan ürün üzerinden işlem yapabilmek için public ürün nesnesi ve sepetin toplam fiyatını göstermemiz için double türünde toplamfiyat değişkeni tanımladı.QR okunduğunda QR kodu temsil eden string değeri SepetActivity.java de bulunan ve ID’si text olan argümana setText ile setledik.

SepetActivity.*text*.setText(result.getText());

Bulunduğumuz QRScanner.java da SepetActivity.java’da ki değişkenlere erişip üzerinde işlem yapabilmemiz için string değişkenler tanımladık.GetText() ile o değeri bulunduğumuz activity’e alıp toString() ile string değere çeviriyoruz.

**Final** String gelentext=SepetActivity.*text*.getText().toString();  
**final** String gelenpara=SepetActivity.*para*.getText().toString();

QR kod ile tarama yapıldıktan sonra ürüne veri tabanından ulaşabilmemiz için öncelikle firebase de ki veritabanına ulaşmamız için gerekli olan kodları yazdık.Yukarda tanımladığımız gelenttext yani ürünün serikodu ile Ürünler veritabanın da ki o serikoda ait ürünün bilgilerini çağırıyoruz.

**Final** FirebaseDatabase database=FirebaseDatabase.*getInstance*();  
**final** DatabaseReference ÜrünlerRef=database.getReference(**“Ürünler/”**+gelentext);

ÜrünlerRef gelecek olan ürünün üzerinde işlem yapmamıza yarıyor. ÜrünlerRef.addValueEventListener() içersinde tanımladığımız onDataChange() fonksiyonu ile ürüne erişebiliyoruz.Burada parametre olarak datasnapshot kullanılıyor.Daha önceden tanımladığımız ürün nesnesine getValue() ile Ürünler.class kullanarak nesnenin stok ,fiyat,serikodu , markası vs. değerlerine ulaşabiliyoruz.Serikodu , ismi ,markası ve fiyatını tabloya gösterdiğimizden dolayı string değişkenler tanımladık.Bu değişkenleri ürün özelliklerini get metodu ile aldık.Toplam fiyat da her ürün okumada değişiklik olacağından dolayı her ürün okumada o an ki ürünün fiyatını gelendoublepara değişkeni ile toplayadık.Sonrasında gelendoublepara değerini string değere dönüştürerek SepetActivity de ki ID’si para olan textView’e setledik.

String kod = **ürün**.getSericode();  
String ad = **ürün**.getAdı();  
String m = **ürün**.getMarka();  
**double** fiyat = **ürün**.getFiyat();  
**double** gelendoublepara=Double.*parseDouble*(gelenpara);  
  
gelendoublepara=gelendoublepara+fiyat;  
  
String stringFiyat=Double.*toString*(fiyat);  
String stringtoplamFiyat=Double.*toString*(gelendoublepara);  
SepetActivity.*para*.setText(stringtoplamFiyat);

Ürün okunduktan sonra sepetimizde ürünün serikodunu , adını ,markasını ve fiyatını görmemiz için öncelikle TableRow nesnesi oluşturduk.

TableRow tableRow = **new** TableRow(getApplicationContext());  
tableRow.setLayoutParams(**new** TableLayout.LayoutParams(TableLayout.LayoutParams.***WRAP\_CONTENT***, TableLayout.LayoutParams.***WRAP\_CONTENT***));

Tabloda her bir satırda görünecek olan değerler için textView argümanı oluşturduk.Bu argümanlara get() metodu ile aldığımız ürünün özelliklerini setleyeceğiz.Her okumada yeniden oluşturacaktır.

TextView serikodTXT = **new** TextView(getApplicationContext());  
TextView adTXT = **new** TextView(getApplicationContext());  
TextView fiyatTXT = **new** TextView(getApplicationContext());  
EditText adetTXT = **new** EditText(getApplicationContext());

Ürünün örneğin seri kodu için get() metodu ile aldığımız serikodunu serikodTXT değişkenine setledik.Ardından kolon başlığı ile aynı genişliğe ve yüksekliğe sahip olması için setWidth() metodunu kullandık.İstediğimiz textSize’a sahip olması için setTextSize() metodunu kullandık.Her satırda ki textlerin background renklerini de aynı renk olması için her textView için setBackgrounColour() metodunu kullandık.Altta ki kod serikod için yapılmıştır.Diğer değerler için de aynı işlemler yapılmıştır.

SerikodTXT.setText(kod);  
serikodTXT.setWidth(400);  
serikodTXT.setHeight(140);  
serikodTXT.setTextSize(14);  
serikodTXT.setBackgroundColor(Color.*parseColor*(**“#00990000”**));  
serikodTXT.setLayoutParams(**new** TableRow.LayoutParams(TableRow.LayoutParams.***WRAP\_CONTENT***, TableRow.LayoutParams.***WRAP\_CONTENT***));

Her textView’e değeri setledikten ve özellik ekletikten sonra tableRow nesnesine addView ile text değişkenlerimizi satıra eklemiş oluyoruz.SepetActivity de ki tableLayout’a ise yine addView() metodu kullanarak tabloya eklemiş oluyoruz.

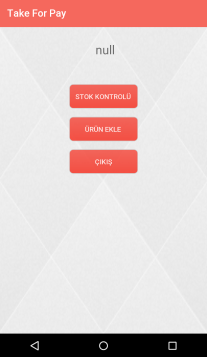
TableRow.addView(serikodTXT);  
tableRow.addView(adTXT);  
tableRow.addView(fiyatTXT);  
tableRow.addView(adetTXT);  
  
SepetActivity.*tableLayout*.addView(tableRow);

**3.7.Yönetici Ekranı**

**3.7.1.Activity\_yonetici.xml**

Uygulamamızı müşteri yanısıra yönetici de kullandığı için yönetici için ayrı bir ekran yaptık.Burada , login ekranında yönetici girişi checkbox’ı seçtikten sonra giriş butonuna bastığımzda activtiy\_yonetici.xml ekranı açılıyor.Bu ekranda yönetici marketinde ki ürünlerin stoklarını kontrol edebilecektir.Aynı zamanda yeni bir ürün eklemek istediğinde ise ürün ekle butonu ile ürün ekleme sayfasına gidecektir.Çıkış butonu ise login sayfasına geri dönmemizi sağlamaktadır.Ekranda tanımladığımız textView yöneticinin giriş yaptığında ismini yazacaktır.Butonların background renklerini , yazı rengi , width , height , yazı boyu değerlerini aynı yaptık.

<**Button  
 android:id=”@+id/StokBTN”  
 android:layout\_width=”135dp”  
 android:layout\_height=”rap\_content”  
 android:background=”@drawable/button\_radius”  
 android:text=”Stok Kontrolü”  
 android:textColor=”#DDFFFFFF”** />



**3.7.2.YoneticiActivity.java**

Activity\_yonetici.xml dosyasında oluşturduğumuz buton ve textView’i YoneticiActivity de kullanmamız için öncelikle public şekide buton ve textView değişkenleri oluşturuyoruz.

**Public** TextView **YöneticiName**;  
**public** Button **ürünekle**;  
**public** Button **Cikis**;  
**public** Button **StokKontrol**;  
**public** Yöneticiler **y**;

Oluşturduğumuz değişkenleri activity\_yonetici.xml de oluşturduğumuz buton ve textView’lerin ID’lerini kullanarak birleştiriyoruz.

**YöneticiName**=findViewById(R.id.***AdSoyadTextView***);  
**ürünekle**=findViewById(R.id.***ÜRÜNBTN***);  
**Cikis**=findViewById(R.id.***CİKİSBTNN***);  
**StokKontrol**=findViewById(R.id.***StokBTN***);

Login ekranında giriş yaparken LoginActivity.java’dan bu activitye giriş yapan yöneticinin ismi ve parolasını da gönderdiğimiz için burada kullanırken öncelikle isim ve parolayı tanımlıyoruz.

**Final** String gelenisim=getIntent().getStringExtra(**“isim”**);  
**final** String gelenp=getIntent().getStringExtra(**“parola”**);

Firebase veritabanında giriş yapan yöneticiye erişmek için gerekli olan kodu ve yöneticinin bilgilerine erişmek için de gelenisim ve gelenp değişkenlerini kullanıyoruz.

**Final** FirebaseDatabase database=FirebaseDatabase.*getInstance*();  
**final** DatabaseReference YöneticilerRef=database.getReference(**“Yöneticiler/”**+gelenisim+**”/”**+gelenp);

Yöneticiye ait bilgilere erişmek için YöneticilerRef değişkenini kullanıyoruz. OnDataChange fonksiyonunun dataSnapshot nesnesini kullanarak yöneticiye ait isim ,kimlik no vs. gibi alt özelliklerine erişebiliyoruz.Üstte tanımladığımız Yöneticiler nesnesine veritabanından gelen yöneticiye ait ad ve soyad değerlerini get() kullanarak atadık.Sonrasında ekranımızda bulunan ID’si YöneticiName olan textView’e setledik.

**for** (DataSnapshot d:dataSnapshot.getChildren()) {  
  
 **y** = d.getValue(Yöneticiler.**class**);  
  
 String kod = **y**.getAdı();  
 String kod2 = **y**.getSoyadı();  
 **YöneticiName**.setText(kod+**” “**+kod2);  
}

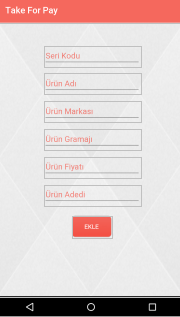
Ürün ekle , stok kontrol ve çıkış butonlarını kullanmamız için SetOnClickListener tanımladık.Ardından onClick fonksiyonu kullandık.

**Ürünekle**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent yeniintent = **new** Intent(YoneticiActivity.**this**, UrunEkleActivity.**class**);  
 startActivity(yeniintent);  
 } });

**3.7.3.Activity\_urun\_ekle.xml**

Yönetici giriş yaptıktan sonra ürün eklemek istediğinde ürün ekle butonuna basacaktır.Ürün ekleme ekranı Edittext’lerden oluşmaktadır.Burada yönetici eklemek istediği ürünün serikodunu , ürün adını , ürün markasını , ürünün gramajını , ürünün fiyatını ve ürünün stok değerlerini girebilecektir.Ekle butonuna ile girilen verileri firebase veritabanına eklemeyecektir.Edittext’lerin yazı boyu ,rengi ,margin değerleri vs. aynıdır.

<**EditText  
 android:id=”@+id/ÜrünGramajTXT”  
 android:layout\_width=”rap\_content”  
 android:layout\_height=”rap\_content”  
 android:layout\_marginTop=”16dp”  
 android:layout\_marginEnd=”96dp”  
 android:ems=”10”  
 android:inputType=”textPersonName”  
 android:text=”Ürün Gramajı”  
 android:textColor=”#aaF44336”  
 android:textColorLink=”#CCF44336”** />



**3.7.4.UrunEkleActivity.java**

Kodlamanın java tarafında öncelikle edittext’leri tanımlamamız gerekiyor.Ardından activity\_urun\_ekle.xml dosyasında tanımladığımız edittextler ile bağdaşlaştırıyoruz.

**Serikod**=findViewById(R.id.***SerikodTXT***);  
**ürünad**=findViewById(R.id.***ürünadTXT***);  
**ürünmiktarı**=findViewById(R.id.***ÜrünGramajTXT***);  
**ürünstok**=findViewById(R.id.***ÜrünadetTXT***);  
**ürünfiyat**=findViewById(R.id.***ÜrünFiyatTXT***);  
**ürünmarka**=findViewById(R.id.***ürünMarkaTXT***);  
**Ekle**=findViewById(R.id.***ÜrünekleBTN***);

Edittext’lere girilen değerleri veritabanına eklememiz için Ekle butonuna seOnClickListener tanımlıyoruz.OnClick() fonksiyonu çalıştığında edittextler de girilen değerleri öncelikle stringe çeviriyoruz.Bu çeviriyi kullanırken herbir değer için ayrı bir string değişkeni tanımlıyoruz.Ardından firebase veritabanına ekleme yapacağımızdan dolayı firebase için gerekli olan kodları ekliyoruz.Ürün eklerken başlık olarak serikodunu kullanıyoruz.Bunun için getKey()’i kullanmamız gerekiyor.Her özellik için bir child ekliyoruz.SetValue() ile edittext’lerde yazılan değerleri veritabanına aktarmış oluyoruz.

String SerikodValue=SerikodRef.getKey();

ÜrünlerRef.child(**"Sericode"**).setValue(sk);  
ÜrünlerRef.child(**"Adı"**).setValue(üa);  
ÜrünlerRef.child(**"Stok"**).setValue(valÜS);  
ÜrünlerRef.child(**"Miktar"**).setValue(valÜmi);  
ÜrünlerRef.child(**"Fiyat"**).setValue(valÜF );  
ÜrünlerRef.child(**"Marka"**).setValue(üm);

**4.ÖNERİLER**

Bu projede android studio kullanıldığı için kullanımını daha önceden öğrenilmesi gerekir.Aynı zamanda xml ve java konusunda da fikir sahibi olmak önemli ölçüde zaman kazandırır.

Projede veritabanı olarak Google firmasına ait firebase kullanıldığı için gerekli kütüphaneleri eklemeye dikkat edilmelidir.Aynı zamanda firebase hakkında fikir sahibi olunması gereklidir.

Proje de test aşamasında mobil cihaz için apk oluşturup telefonda denenmesi yerine Genymotion isimli emülatör ile bilgisayar üzerinde test edilebilir.Bu da test aşamasında biraz daha zaman kazandırır.

Projede markete giriş esnasında kullanılan cihaz , şuan ki projede mobil cihazdır.Mobil cihaz yerine ardunio veya raspberry Pİ gibi cihazlar kullanılabilir.

**5.KAYNAKÇA**

1. <https://medium.com/@sinanyilmaz/android-firebase-authentication-email-parola-ve-google-cc154d1bf5fe> Android Firebase Doğrulama
2. <https://kodcu.com/2011/10/bolum-4-r-java-resources-kaynaklar-ve-assets-varliklar/>

FindViewById

1. <https://www.mobilhanem.com/android-imageview-kullanimi/> İmageview Kullanımı
2. <https://www.idautomation.com/font-encoders/java-class-library/> Barcode Class
3. <https://medium.com/mobiletech/android-firebase-projeye-ekleme-c9c97560e225> Firebase Projeye Ekleme
4. <https://bit.ly/2s09pMo> AppCompatActivity
5. <https://bit.ly/1RIHCCP> Firebase nedir?
6. <https://bit.ly/2LE8Q3O> Activity ve layout bilgi

**6.STANDARTLAR ve KISITLAR FORMU**

Projenin hazırlanmasında uyulan standart ve kısıtlarla ilgili olarak, aşağıdaki soruları cevaplayınız.

1. Projenizin tasarım boyutu nedir? (Yeni bir proje midir? Var olan bir projenin tekrarı mıdır? Bir projenin parçası mıdır? Sizin tasarımınız proje toplamının yüzde olarak ne kadarını oluşturmaktadır?)

|  |
| --- |
| Projeye sıfırdan başladık.Dışardan projeye bazı temel eklemeler yapılmıştır. (Kütüphaneler vs.) |

1. Projenizde bir mühendislik problemini kendiniz formüle edip, çözdünüz mü? Açıklayınız.

|  |
| --- |
| Projenin akışını biz oluşturduğumuz için karşılaştığımız sorunları kendimiz çözdük. |

1. Önceki derslerde edindiğiniz hangi bilgi ve becerileri kullandınız?

|  |
| --- |
| Android programlama dersini(Windows programlama) aldığımız için bilgi birikimimiz ile projeye başladık. |

1. Kullandığınız veya dikkate aldığınız mühendislik standartları nelerdir? (Proje konunuzla ilgili olarak kullandığınız ve kullanılması gereken standartları burada kod ve isimleri ile sıralayınız).

|  |
| --- |
| IEEE / EIA 12207 – Yazılım Geliştirme Yaşam Döngüleri |

1. Kullandığınız veya dikkate aldığınız gerçekçi kısıtlar nelerdir? Lütfen boşlukları uygun yanıtlarla doldurunuz.

a) Ekonomi

|  |
| --- |
| Maliyeti göz önünde bulundurup daha basit işlemlere başvurduk. |

b) Çevre sorunları:

|  |
| --- |
| Projemizde çevre için sorun oluşturacak bir durum olmamaktadır. |

c) Sürdürülebilirlik:

|  |
| --- |
| Projemiz sürekli aktif olacağından dolayı herhangi bir zaman kısıtlaması bulunmamaktadır. |

d) Üretilebilirlik:

|  |
| --- |
| Projemizde üretilebilirlik android uygulamasına güncelleştirme yaparak mümkün olabilir. |

e) Etik:

|  |
| --- |
| Ürettiğimiz proje herhangi bir etik kurala aykırı değildir. |

f) Sağlık:

|  |
| --- |
| Projemizde herhangi bir sağlık sorunu oluşturacak bir durum söz konusu değildir. |

g) Güvenlik:

|  |
| --- |
| Projemizde kişisel verilerin güvenliği için benzersiz id’lere yer verilmiştir. |

h) Sosyal ve politik sorunlar:

|  |
| --- |
| Projemizde sosyal ve politik sorunlara yol açmamaktadır. |