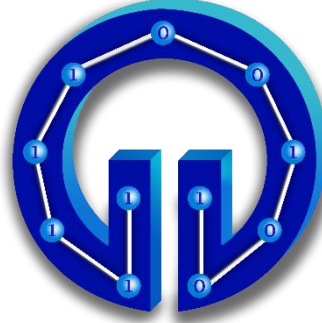


**KARADENİZ TEKNİK ÜNİVERSİTESİ  
MÜHENDİSLİK FAKÜLTESİ  
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**



**ANDROİD MOBİL MARKET UYGULAMASI**

**MÜHENDİSLİK TASARIMI**

**Kürşat ERCAN  
Burak ALTUNTAŞ**

**2020-2021 GÜZ DÖNEMİ**

**KARADENİZ TEKNİK ÜNİVERSİTESİ  
MÜHENDİSLİK FAKÜLTESİ  
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ**

**ANDROİD MOBİL MARKET UYGULAMASI**

**MÜHENDİSLİK TASARIMI**

**Kürşat ERCAN  
Burak ALTUNTAŞ**

**2020-2021 GÜZ DÖNEMİ**



## **IEEE Etik Kuralları IEEE Code of Ethics**



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 ayrımcılık 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

“Android Mobil Market Uygulaması” isimli bu çalışma, Karadeniz Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü’nde güz dönemi tasarım projesi olarak hazırlanmıştır.

En başta danışman hocamız Doç. Dr. Bekir DİZDAROĞLU’na ve desteğini hiçbir zaman esirgemeyen kıymetli ailelerimize teşekkürlerimizi sunarız.

Kürşat ERCAN  
Burak ALTUNTAŞ  
Trabzon 2021

## İÇİNDEKİLER

	Sayfa No
IEEE ETİK KURALLARI.....	II
ÖNSÖZ.....	III
İÇİNDEKİLER.....	IV
ÖZET.....	V
1. GENEL BİLGİLER.....	1
1.1. GİRİŞ.....	1
1.2. KULLANILACAK TEKNOLOJİLER.....	2
1.2.1. JAVA.....	2
1.2.2. ANDROID STUDIO.....	2
1.2.3. FİREBASE.....	3
2. PROJE TASARIMI.....	4
2.1. GEREKSİNİM ANALİZİ.....	4
2.1.1. SİSTEM GİRDİLERİ.....	4
2.1.2. SİSTEM KAYNAKLARI.....	4
2.1.3. SİSTEM KARARLILIĞI.....	4
2.1.4. SİSTEM HEDEFLERİ.....	4
2.2. MİMARİ TASARIM.....	5
2.3. UML – USE CASE DİYAGRAMI.....	10
2.4. YAPILAN ÇALIŞMALAR .....	11
2.4.1. FİREBASE ENTEGRASYONU .....	11
2.4.2 GİRİŞ VE KAYIT İŞLEVLERİNİN GERÇEKLENMESİ.....	11
2.4.3 İLAN OLUŞTURMA.....	13
2.4.4 İLANLARIN AKIŞ SAYFASINDA GÖRÜNTÜLENMESİ.....	14
2.4.5 MESAJLAŞMA İŞLEVİNİN GERÇEKLEMESİ.....	15
3. KAYNAKLAR.....	17
STANDARTLAR ve KISITLAR FORMU.....	18

## ÖZET

Projenin amacı insanların yetenek ve uzmanlıklarını sergiledikleri, ilanlar açtıkları aynı zamanda girişimcilerin de ihtiyaçları doğrultusunda destek arama amacıyla ilanlar açabildiği ve bu ilanlar vasıtasıyla kullanıcıların iletişim kurabilecekleri bir platform oluşturmaktır.

Platformun geliştirilme motivasyonu insanların yeteneklerini sergileyebildikleri, hobilerine maddi karşılıklar bulabildikleri, istedikleri ve sevdikleri işi, ne zaman, nerede ve kiminle yapacaklarına karar verme özgürlüğüne sahip oldukları bir pazar yeri düşüncesinden beslenmektedir.

## **1. GENEL BİLGİLER**

### **1.1. Giriş**

Firmaların çalışma saatlerindeki esnek olmayan katı kuralları insanların kişisel gelişmelerinde engel, sosyalleşmelerinde olumsuz etken ve üretkenliklerinde performans kaybına neden olabilmektedir. Ayrıca insanların nereden ve nasıl iş bulacakları konusunda yaşadıkları endişeler yüzünden mutsuz oldukları, çalıştıkları yerlerde hak ettikleri değeri göremedikleri, işverenlerin bütçelerini zorlayan giderleri ve doğru çalışanı bulamadıkları bir zaman diliminde yaşamaktayız.

İnsanlar hem sevdikleri işi yapmak hem de mutlu olmayı isterler. Geliştirmekte olduğumuz bu proje ile yetenekli insanlara yeteneklerini değerlendirebilecekleri, öğrencilere ve iş arayanlara gelir elde edebilecekleri ve firmaların ihtiyaçlarına uygun fiyat ile hızlı olarak çözümler elde edebilecekleri bir ortam sunmayı hedeflemekteyiz. Geliştirilen bu platform ile kullanıcılara hobileri ile gelir elde etme imkânı, istedikleri ve sevdikleri işi, ne zaman, nerede ve kiminle yapacaklarına karar verme özgürlükleri sağlanacaktır.

## 1.2 KULLANILACAK TEKNOLOJİLER

Proje Java programlama dili ve Android Studio geliştirme ortamı kullanılarak geliştirilmiştir. Projede veritabanı ve depolama işlemleri için sunucu tabanlı Firebase servisi kullanılmaktadır.

### 1.2.1 JAVA

Java, 1995 yılında James Gosling tarafından geliştirilerek piyasaya sürülmüş bir uygulamadır. Java mobil ve gömülü uygulamalar, web tabanlı içerikler ve oyunlar başta olmak üzere neredeyse her platformda kullanılmaktadır. Dünya üzerinde 9 milyon Java geliştiricisinin bulunması, kullanım yelpazesinin ne denli geniş olduğunu bir göstergesidir.

Java nesneye yönelimli, çok fonksiyonlu yapıda ve yüksek seviyeli bir dildir. Herhangi bir bilgisayar mimarisine bağlı olmadan ve platformlardan bağımsız olarak çalışabilmektedir.

Java yazılımlarını geliştirmek için JDK(Java Development Kit) programının sisteme kurulması gerekmektedir.

Java'nın başlıca nitelikleri şunlardır:

- Yazımı, derlenmesi ve düzeltilmesi bakımından diğer dillere göre basittir.
- Otomatik bellek tahsisi yapar ve işi biten nesneleri yok eder.
- Nesne yönelimli programla değerlerinin bütün avantajlarını taşımaktadır.
- Çoklu iş yeteneklerine sahiptir.
- Yüksek performansa sahiptir.
- Güçlü hata ayıklama (debug) yeteneğine sahiptir.

### 1.2.2 ANDROID STUDIO

Android Studio, Google tarafından 16 Mayıs 2013 tarihinde Google I/O etkinliğinde tanıtılmıştır. IntelliJ IDEA'ya dayalı olup Android uygulamaları geliştirmek için özel olarak tasarlanmış bir tümleşik geliştirme ortamıdır (IDE).

Android Studio, akıllı bir kod düzenleyici ve hata ayıklayıcı, performans analiz araçları, emülatörler ve daha pek çok araç dahil olmak üzere uygulama geliştirmek için kullanıcıların ihtiyacını karşılayabilecek birçok özellik barındırır.

**Android Studio'nun başlıca nitelikleri şunlardır:**

- Gradle tabanlı bir oluşturma sistemi içerir.
- Kendi içerisinde hazır bir AVD (emülatör) barındırır.
- Anında çalıştırma özelliğiyle emülatör üzerinde çalışan uygulamanızın hızlı bir şekilde yansıtılmasını sağlayarak düzenleme, derleme ve çalıştırma süreçlerini hızlandırır.
- Github entegrasyonu kolaydır.
- Akıllı kod düzenleyicisi ve kod şablonları içerir. Bu sayede hızlı ve güvenli yazılım geliştirme imkanı sunar.
- Performans, kullanılabilirlik, sürüm uyumluluğu ve diğer sorunları yakalamak için Lint araçları mevcuttur.
- Geliştirilen uygulamaları APK formatında çıktı olarak almaya imkân tanır.



### 1.2.3. FIREBASE

2014 yılının Ekim ayında Google tarafından satın alınan Firebase platformu, mobil ve web geliştiriciler için back-end hizmeti sunan bir cloud (bulut) teknolojisidir. Firebase, database, authentication, storage, hosting, bildirim gönderme ve test ortamları gibi hizmetleri sunucu tarafında geliştiricinin bir kodlama yapmadan kullanmasına imkan tanır.

Firebase sunduğu hizmetler ile uygulama geliştirme sürecinde geliştiriciye zaman kazandır.

#### Firestore'in başlıca nitelikleri şunlardır:

- İçerisinde uygulamaya kaydolun kullanıcıların bilgilerini doğrulamak için authentication hizmeti mevcuttur. Bu hizmet sayesinde eposta doğrulama, eposta değiştirme, şifre sıfırlama ve farklı sosyal medya hesapları(Google, Facebook, Twittter ve vb.) ile kullanıcıların sisteme kayıt olma ve giriş yapma işlemleri kolaylıkla gerçekleştirilir.
- İstemci ve sunucu tarafı geliştirme amacıyla verileri depolamak ve senkronize etmek için Cloud Firestore adında verileri koleksiyonlar halinde düzenlenmiş belgelerde saklayan NoSQL bulut veritabanı barındırır.
- Cloud Firestore, mobil, web ve sunucu geliştirme için esnek, ölçeklenebilir bir veritabanıdır.
- Firestore verileri gerçek zamanlı dinleyiciler aracılığı ile istemci uygulamalarında senkronize halde tutar ve uygulamalarda kullanıcılara senkron bir şekilde veri aktarımına izin verir.
- Firestore içerisinde bulut depolama hizmeti sunmaktadır. Tüm metin, resim gibi dosyalar burada saklanabileceği gibi uygulamaların Log dosyaları da burada tutularak uygulamada oluşan hataları anlık olarak takip etmeye imkan tanır.
- Uygulamada kullanıcılar ile anlık olarak iletişime geçilmek isteniyorsa Firestore içerisinde bulunan notification servisi ile kullanıcılara anlık olarak bildirim gönderilebilir.
- Firestore Analytics ile uygulama içi aktiviteler, kullanıcılar ve daha bir çok bilgiyi görüp analizler yapılabilir.
- Dokümantasyonu istenilen bilgiye en hızlı ve anlaşılır olarak ulaşılabilecek şekilde hazırlanmıştır.

## **2. PROJE TASARIMI**

### **2.1 GEREKSİNİM ANALİZİ**

#### **2.1.1 SİSTEM GİRDİLERİ**

Sistem girdileri şu şekildedir:

- Kullanıcılar üyelik oluştururlar ve bu üyelikleri ile giriş yaparlar.
- Kullanıcılar ilan oluşturabilirler.
- Kullanıcılar ihtiyaç duydukları alanlarda ilanları ve ilan sahibinin portfolyosunu inceleyebilir.
- Kullanıcı çalışmaya karar verdiği ilan sahibi ile platform üzerinden iletişim kurabilir.
- Kullanıcılar arasında bir anlaşmaya varılır ise kullanıcı ilan sahibine sipariş verebilir.
- İlan sahibi siparişi kabul eder ve çalışmayı teslim ederse müşteri ilan sahibine değerlendirme notu verebilir ve profiline yorum yazabilir.

#### **2.1.2 SİSTEM KAYNAKLARI**

Sistem kaynakları; platform arayüzleri ve kullanılan ikonlar, kullanıcıların profilleri, yayınlanan ilanlar, değerlendirmeler, geliştirme ortamları (Java / Android Studio), tüm verilerin tutulduğu sunucu tabanlı bir veri tabanı (Firebase - Firestore) ve bulut depolama ortamı olarak sıralanmaktadır.

#### **2.1.3 SİSTEM KARARLILIĞI**

Sistem kararlılığını belirleyen iki unsur şu şekildedir:

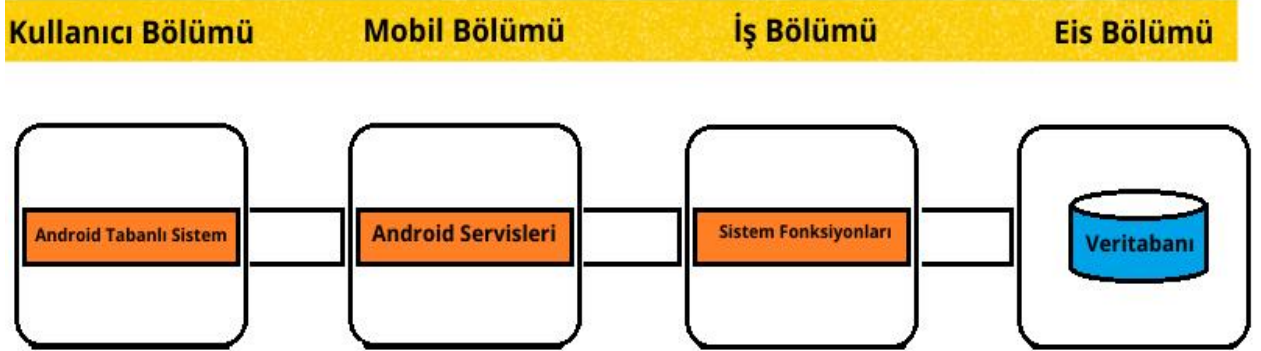
- Tüm cihazlar arasındaki senkronizasyonun doğru bir şekilde sağlanması.
- Sistemin girdileri ve çıktılarının kullanıcıya doğru bir şekilde sunulması.

#### **2.1.4 SİSTEM HEDEFLERİ**

Sistemin minimum hedefi sistem girdi ve çıktılarının doğru olarak kullanıcıya sunulmasıdır. Belirlenen asıl hedef ise hızlı, stabil ve güvenli bir platform oluşturmak

## 2.2 MİMARİ TASARIM

### Mantıksal Bakış:



- Kullanıcı Bölümü uygulamanın sayfaları gibi kısımlarla görevlidir.
- İş Bölümü sistemin temel fonksiyonlarını yerine getirmekle görevlidir.
- EIS (Enterprise Information System) bölümü veri saklamakla görevlidir.

### Test:

- Kodlama aşamasında birim testleri yapılmalıdır.

### Performans:

- Veritabanı sorguları 2 saniye içerisinde sonuçlanmalıdır.
- Uygulama tüm sistemlerde kararlı çalışmalıdır.

### Lokalizasyon:

Uygulamamız İngilizce ve Türkçe dillerine destek vermektedir. Bu desteği Android'in bize sunmuş olduğu "strings.xml" dosyası ile yapılacaktır.

#### Strings\_turkish.xml

```
<resources>
  <string name="action_settings">Ayarlar</string>
</resources>
```

#### - Strings\_english.xml

```
<resources>
  <string name="action_settings">Settings</string>
</resources>
```

### Riskler:

- Ödeme sistemi için uygulama herhangi bir güvence sağlamamaktadır. Dolayısıyla ödeme tamamen kullanıcılar arasındaki iletişim ve güvene dayalı olacağından istenmeyen durumların ortaya çıkma olasılığı olabilir.

### Veritabanı İlişkileri:

Veritabanı ilişkileri, iki kullanıcı ve ilan olduğu varsayılarak aşağıdaki şekilde hazırlanmıştır.

#### Kullanıcılar:{

```
User_id_1:{
  Kullanıcı Adı: "isim",
  Email: "test@test.com",
  Profil Fotoğrafı:
  Bio:
}
User_id_2:{
  Kullanıcı Adı: "isim",
  Email: "test1@test.com",
  Profil Fotoğrafı:
  Bio:
}
```

}

#### İlanlar:{

```
Ilan_id_1:{
  User_id:user_id_2,
  Ilan_baslik: "Ben sizin için kurumsal web sitenizi tasarlayabilirim.",
  Msg_tarih: "16 Ocak 2021 03:52:17 UTC+3",
  kategoriler[]:{kategori_1,kategori_2},
  ilan_turu: "0" // 0 iş yaptırmak için 1 çalışmak için,
  ilan_detayi: "ilan detayı",
  ilan_basligi: "ilan başlığı",
  fiyat_bilgisi: "500TL"
}
```

}

#### Kategoriler:{

```
Katagori_1:{
  Kategori_adi: "web",
}
Katagori_2:{
  Kategori_adi: "mobil",
}
```

}

#### Mesajlar:{

```
mesaj_id_1:{
  Mesaj_alan:user_id_1,
  Mesaj_gonderen:user_id_2,
  Mesaj_icerikleri:{
    mesaj_id_1:{
      msg_gonderen: user_id_2
      msg_icerik: "Merhaba."
      Msg_tarih: "16 Ocak 2021 05:52:17 UTC+3"
```

```

    }
    mesaj_id_2:{
      msg_gonderen:user_id_1
      msg_icerik: "Merhaba."
      msg_tarih: "16 Ocak 2021 05:52:17 UTC+3"
    }
  }
}

```

```

Teklifler:{
  teklif_id_1:{
    Teklif_veren_id:user_id1,
    Teklif_yapilan_ilan_id:ilan_id_1,
    kabul_edildi:false
  }
}

```

```

Siparislerim:{
  Siparis_1{
    Ilan_id:ilan_id_1,
    User_id:user_id_1,
    onay:false
  }
}

```

```

Incelemeler:{
  Ilan_id:Ilan_id_1,
  inceleme_yorumlari:{
    inceleme_gonderen:user_id_1
    inceleme_icerik:"verilen hizmetten memnun kaldım"
    inceleme_tarih: "16 Ocak 2021 05:52:17 UTC+3"
  }
}

```

```

Sikayetler:{
  Sikayet_id_1:{
    Sikayet_eden_id:user_id_1,
    Sikayet_edilen_id:user_id_2,
    Sikayet_sebep: "sebep"
  }
}

```


```

Banlananlar:{
  Banlananlar_id_1:{
    Ban_kullanici:user_id_2,
    Ban_sebep:"sebep"
  }
}

```

## Kullanıcı Arayüzü Tasarımları:

PM-ProjectMarket



User Name

Email


Password

REGISTER

Already have an account? Sign In.

(Kayıt Arayüzü)

PM-ProjectMarket



Email

Password

SIGN IN


Don't you have an account yet? Now, Sign Up.

(Giriş Arayüzü)

PM-ProjectMarket

×

SHARE




I can ... for you. (in a sentence)

Price..

(İlan Paylaşım Arayüzü)

PM-ProjectMarket



Profesyonel logo tasarımı yapabiliyim.


test 1000

+1400

PROFESYONEL  
LOGO  
TASARIMI

Sizin için logo tasarlayabilirim.

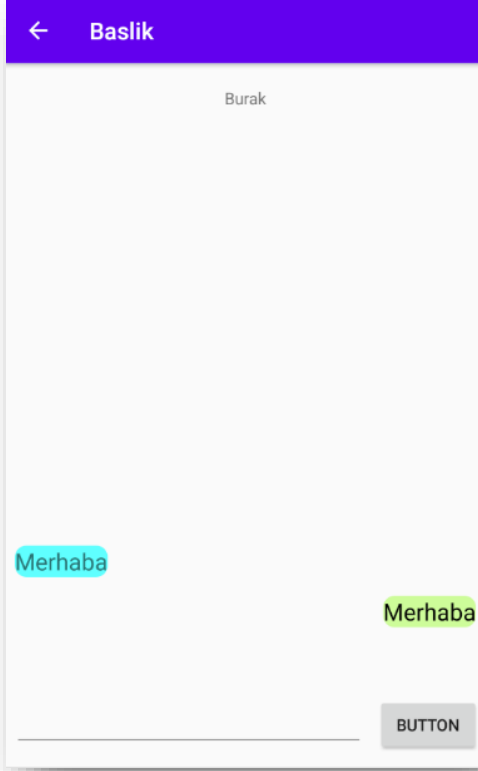
test1 500



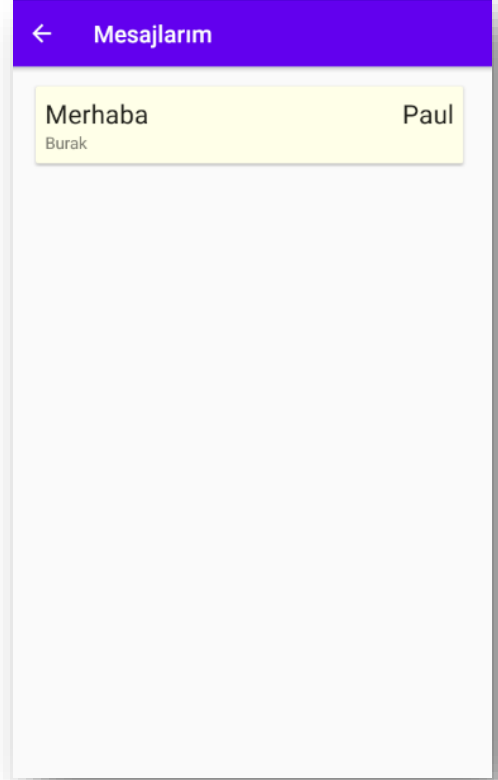
Sizin için arduino projelerinizi gerçekleştire bilirim.

Home Add Messages Profile

(İlan Akış Sayfası)

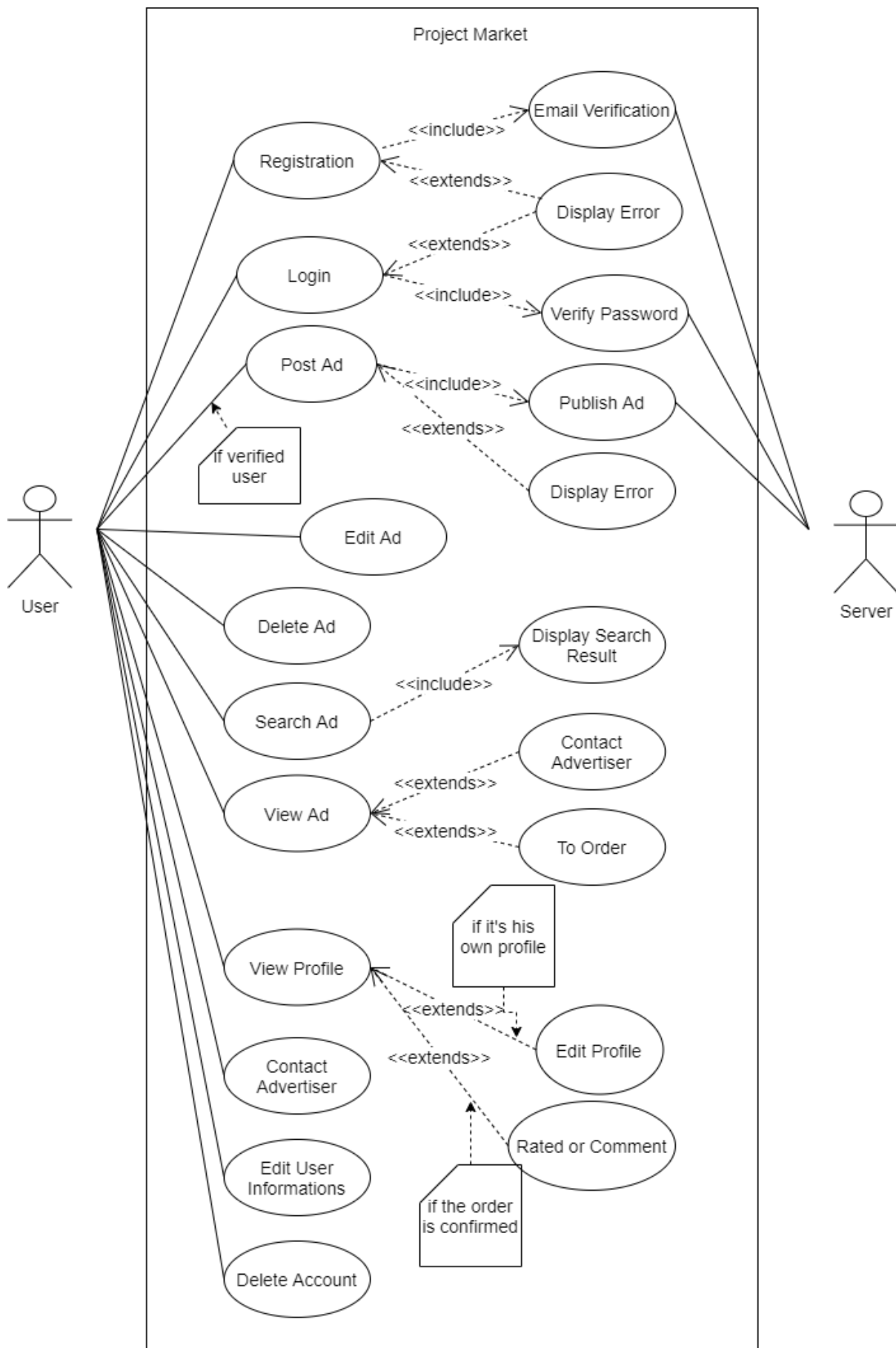


(Mesajlaşma arayüzü)



(Gelen kutusu arayüzü)

## 2.3 UML – USE CASE DIAGRAM





## 2.4. YAPILAN ÇALIŞMALAR

Geliştirilmekte olan projeden beklenen temel yeteneklerin sağlanması üzerine çalışılmıştır. Yapılan çalışmalar 6 başlık altında incelenebilir:

- Firebase kurulumu ve entegrasyonu
- Giriş ve kayıt işlevlerinin gerçekleştirilmesi
- İlan oluşturma
- İlanların akış sayfasında görüntülenmesi
- Mesajlaşma işlevinin gerçekleştirilmesi.

### 2.4.1. FIREBASE ENTEGRASYONU

Firestore konsolu üzerinden yeni bir proje oluşturulup geliştirilmekte olan uygulama projeye eklenmiştir. Ekleme yapıldıktan sonra konsol üzerinden indirilen Firebase Android yapılandırma dosyası (google-services.json) uygulama içerisine eklenip App level Gradle içerisinde eklenti uygulanmıştır.

```
apply plugin: 'com.google.gms.google-services'
```

Uygulamada kullanılacak olan Firebase'in Authentication, Cloud Firestore ve Storage hizmetlerine ait SDK'ları uygulama düzeyinde Gradle içerisine eklenmesi gerekmektedir. Bu konuda Firebase'in Malzeme Listesi olarak adlandırdığı BoM hizmeti ile SDK'ların sürümlerini otomatik olarak güncellemek mümkün.

```
implementation platform('com.google.firebase:firebase-bom:26.2.0')
implementation 'com.google.firebase:firebase-auth:20.0.1'
implementation 'com.google.firebase:firebase-firestore:22.0.1'
implementation 'com.google.firebase:firebase-storage:19.2.1'
```

### 2.4.2. GİRİŞ VE KAYIT İŞLEVİNİN GERÇEKLENMESİ

Uygulamaya ilk defa kaydolacak kişiler kaydolma ekranından kaydolmaları gerekmektedir. Burada kullanıcıdan kullanıcı adı, e-posta ve parola bilgilerini girmesi istenmiştir. Projenin ilerleyen dönemlerinde aktivasyon kodu ile mail onaylaması özelliği eklenecektir. Bu aşamada fonksiyonel olarak çalışması amaçlanmıştır.

Kullanıcı bu bilgileri girdikten sonra öncelikle istenilen alanları doldurup doldurmadığı kontrol edilmiştir. Eğer doldurmuş ise kayıt işlemi için *registerUser* fonksiyonu çağırılmıştır.

```
String userName = userNameText.getText().toString();
String email = emailText.getText().toString();
String password = passwordText.getText().toString();
if (TextUtils.isEmpty(userName) || TextUtils.isEmpty(email) || TextUtils.isEmpty(password)){
    Toast.makeText(RegisterActivity.this, "Please, fill in all fields!",
        Toast.LENGTH_LONG).show();
} else if (password.length() < 6){
    Toast.makeText(RegisterActivity.this, "Password must be longer than 5 characters!",
        Toast.LENGTH_LONG).show();
} else { registerUser(userName, email, password); }
```

registerUser fonksiyonunda ise Firebase Authentication üzerinden kayıt işlemi gerçekleştirilmiştir. Authentication servisi geçersiz bir mail adresi veya daha önce kullanılmış bir mail adresi girilmesi durumunda hata sebebinin geri döndürmektedir. Kullanıcı oluşturma işlemi şu şekilde yapılmıştır:

```
private void registerUser(String userName, String email, String password){
    progressDialog.setMessage("Please Wait!");
    progressDialog.show();
    firebaseAuth.createUserWithEmailAndPassword(email, password)
    .addOnSuccessListener(authResult -> {
        Toast.makeText(RegisterActivity.this, "Welcome..", Toast.LENGTH_LONG).show();
        createUser(userName, email);
    }).addOnFailureListener(e -> {
        Toast.makeText(RegisterActivity.this, e.getLocalizedMessage(), Toast.LENGTH_LONG).show();
        progressDialog.dismiss();
    });
}
```

Verilen kod parçasında da görüldüğü üzere kayıt işlevi gerçekleştirilse createUser adlı fonksiyona çağrılmaktadır. Bu fonksiyonda ise sisteme kaydı yapılan kullanıcı veri tabanına da kaydedilmiştir.

Giriş ekranında ise öncelikle kullanıcıdan doldurulması istenilen alanların dolu olup olmadığını kontrolü yapılmıştır. İstenilen alanları boş bırakmışsa ekranda boş bıraktığı yerleri doldurması için mesaj gösterilmiştir. Eğer boş bırakmamışsa Authentication servisi ile uygulamaya girişi sağlanmıştır.

```
private void loginUser(String email, String password){
    firebaseAuth.signInWithEmailAndPassword(email, password)
    .addOnSuccessListener(authResult -> {
        Intent intent = new Intent(LoginActivity.this, MainActivity.class);
        startActivity(intent);
        finish();
    }).addOnFailureListener(e->Toast.makeText(LoginActivity.this,"Oops! Something went wrong.",
    Toast.LENGTH_LONG).show());    }
```

Giriş ekranı uygulamanın ilk başlatılacak olan aktivitesi olduğundan eğer kullanıcı daha önceden giriş yapmışsa uygulamayı tekrardan başlattığında giriş yapmadan ana menüye geçmesi şu kodlarla sağlanmıştır.

```
FirebaseUser firebaseUser = firebaseAuth.getCurrentUser();

if(firebaseUser != null){
    Intent intent = new Intent(this, MainActivity.class);
    startActivity(intent);
    finish();
}
```

### 2.4.3. İLAN OLUŞTURMA

İlan oluşturma aşamasında kullanıcıdan ilan başlığı, ilan görseli ve fiyat bilgisi istenmiştir. Burada görsel cihaz içerisinden alınacağı için depolama alanına erişim için izinlerin verilmesi gerekmektedir.

AndroidManifest.xml dosyası içerisinde depolama biriminden okuma erişimi şu şekilde ayarlanmıştır:

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

Galeriyi açmak için şu kodlar kullanılmıştır:

```
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {
    if(requestCode == 1){
        if(grantResults.length>0 && grantResults[0] == PackageManager.PERMISSION_GRANTED){
            Intent intentToGallery = new Intent(Intent.ACTION_PICK,
            MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
            startActivityForResult(intentToGallery,2);
        }
    }
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
}
```

Galeriye erişim sağlandıktan sonra seçilen görseli çekilmiş ve decode edilmiştir. Görsel verisinin decode işlemi SDK-28 sürümünden itibaren önceki sürümlerde uygulanan yöntem desteklenmemeye başlanmıştır. Bu değişiklik cihazın versiyon kontrolü ile sağlanmıştır.

```
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    if(requestCode == 2 && resultCode == RESULT_OK && data != null){
        imageData = data.getData();
        try { if (Build.VERSION.SDK_INT>=28){
            ImageDecoder.Source source = ImageDecoder
            .createSource(this.getContentResolver(),imageData);
            selectedImage = ImageDecoder.decodeBitmap(source);
        }else{
            selectedImage = MediaStore.Images.Media
            .getBitmap(this.getContentResolver(),imageData);
        }
        postImage.setImageBitmap(selectedImage);
    } catch (IOException e) { e.printStackTrace(); }
}
super.onActivityResult(requestCode, resultCode, data);
}
```

Veriler toplandıktan sonra postData adlı Hash Map yapısında atılıp veri tabanına postData ile yüklenir.

```
db.collection("Posts").add(postData).addOnSuccessListener(documentReference -> {
    Intent intent = new Intent(PostActivity.this, MainActivity.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
    startActivity(intent);
    finish();
}).addOnFailureListener(e ->
    Toast.makeText(PostActivity.this,e.getLocalizedMessage(),Toast.LENGTH_LONG).show());
```

#### 2.4.4. İLANLARIN AKIŞ SAYFASINDA GÖRÜNTÜLENMESİ

Akış sayfasında ilanlar, en son atılan ilan en üstte görülecek şekilde planlanmıştır. Bu nedenle ilanlar veritabanından çekilirken en yeni tarihli en önce alınacak şekilde orderBy sorgusu ile çekilmiştir. İlanlar Posts koleksiyonundan çekilirken aynı zamanda ilan sahibinin bilgileri de Users koleksiyonundan id bilgisi ile çekilmiştir. İlgili kod parçaları aşağıda verilmiştir.

```
public void getDataFromDB(){
    db.collection("Posts").orderBy("date",Query.Direction.DESENDING)
    .addSnapshotListener((EventListener<QuerySnapshot>) (value, error) -> {
        if(value != null){
            for(DocumentSnapshot doc : value.getDocuments()){
                Post post = doc.toObject(Post.class);

                db.collection("Users").document(post.getUserId()).get()
                .addOnCompleteListener((OnCompleteListener<DocumentSnapshot>) task -> {
                    if (task.isSuccessful()){
                        User user = task.getResult().toObject(User.class);

                        userNameList_db.add((String) user.getUserName());
                        profileImageList_db.add(user.getProfileImageUrl());
                        titleList_db.add(post.getTitle());
                        postImageList_db.add(post.getPostImageUrl());
                        priceList_db.add(post.getPrice());

                        postRecyclerAdapter.notifyDataSetChanged();
                    }
                });
            }
        }
    });
}
```

### 2.4.5. MESAJLAŞMA İŞLEVİNİN GERÇEKLEMESİ

Kişiyle mesajlaşma ilk olarak profil ekranından başlamaktadır. Profil ekranından mesaj at butonuna bastığımızda giriş yapan (o anki kullanıcı) ve profilini görüntülediğim kullanıcının id'leri alınır. "Messages" isimli koleksiyonun içinde iki kişi arasında bir doküman oluşturulur. İki kullanıcının mesaj kutusunda bu konuşma ve yazılan son mesaj görülür. Kullanıcılar tekrar mesajlaşmak için ya mesaj kutusunu ya da profil ekranındaki mesaj gönder butonu üzerinden haberleşebilirler. Bu haberleşme, tasarladığımız mesajlaşma arayüzünden gerçekleştirilmektedir. Mesajlaşma arayüzünde "messages" koleksiyonunun içi sorgulanır mesajı gönderen veya alan o anki giriş yapmış kullanıcı ise daha önce oluşturulmuş iki kişi arasındaki doküman id'si (token) üzerinden bu dokümanın içerisinde olan "mesajlar" dokümanına erişilerek iki kişi arasındaki mesajlar çekilir.

Mesajlaşma arayüzü kodları (Resim: Mesajlaşma Arayüzü):

```
@Override
protected void onStart() {
    super.onStart();
    messagesRef1 = db.collection("messages/" + token + "/inner_messages");
    messagesRef1
        .orderBy("msg_tarih", Query.Direction.ASCENDING)
        .addSnapshotListener(this, new EventListener<QuerySnapshot>() {
            @Override
            public void onEvent(@Nullable QuerySnapshot queryDocumentSnapshots,
                                @Nullable FirebaseFirestoreException e) {
                chatMdl.clear();
                for (DocumentSnapshot document : queryDocumentSnapshots.getDocuments()) {
                    chatMdl.add(new MessagesChatModel(document.getString("msg_gonderen_k"),
"Burak", document.getString("msg_icerik")));
                    mesaj_kutusuAdapter msgAdapter = new mesaj_kutusuAdapter(chatMdl);
                    Rcw.setAdapter(msgAdapter);
                }
            }
        });
}
```

## Gelen mesajlar (Resim: Gelen Kutusu):

```

@Override
protected void onStart() {
    super.onStart();
    messagesRef.addSnapshotListener(this, new EventListener<QuerySnapshot>() {
        @Override
        public void onEvent(@Nullable QuerySnapshot queryDocumentSnapshots,
            @Nullable FirebaseFirestoreException e) {
            if(e!=null){
                Toast.makeText(mesaj_goruntule.this, "Hata !", Toast.LENGTH_SHORT).show();
                return;
            }
            for (DocumentSnapshot document : queryDocumentSnapshots.getDocuments()) {

                if(document.getString("msg_alan_id").equals(msg_alan_id) ||
                    document.getString("msg_gonderen_id").equals(msg_alan_id) ){
                    messagesRef1=db.collection("messages/"+document.getId()+"/inner_messages");
                    gonderenAdi=document.getString("msg_alan_adi");
                    messagesRef1
                        .orderBy("msg_tarih",Query.Direction.DESENDING)
                        .limit(1)
                        .addSnapshotListener(new EventListener<QuerySnapshot>() {
                            @Override
                            public void onEvent(@Nullable QuerySnapshot queryDocumentSnapshots,
                                @Nullable FirebaseFirestoreException e) {
                                if(e!=null){
                                    Toast.makeText(mesaj_goruntule.this, "Hata !",
                                        Toast.LENGTH_SHORT).show();
                                    return;
                                }
                            }
                            for (DocumentSnapshot document1 :
                                queryDocumentSnapshots.getDocuments()) {

                                exampleList.add(new Messages(
                                    document.getString("msg_alan_adi"),
                                    document.getString("msg_gonderen_adi") ,
                                    document1.getString("msg_icerik"),document.getId() ));
                            }
                        })
                }
            }
        }
    });
}

```

### 3. KAYNAKLAR

URL <https://www.pomelosoft.com/blog/java-nedir>

URL [https://java.com/tr/download/help/whatis\\_java.html](https://java.com/tr/download/help/whatis_java.html)

URL <https://developer.android.com/studio/intro>

URL <https://www.metinpolat.net/blog/android-studio-nedir-nasil-kurulur/>

URL <https://devnot.com/2017/web-ve-mobil-uygulamalar-icin-firebase/>

URL <https://firebase.google.com/docs/firestore?authuser=2>

URL <https://firebase.google.com/docs/android/setup?authuser=0>

## 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?)

Hali hazırda kullanılmakta olan benzer ürünler mevcuttur. Bizim uygulamamız bu uygulamalarla aynı konuda hizmet etse de kullanıcı deneyimi, arayüzler ve işleyiş açısından muadillerinden ayrılmaktadır.

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

Tasarım aşamasında çözülmesi gereken bir mühendislik problemiyle karşılaşmadık.

3. Önceki derslerde edindiğiniz hangi bilgi ve becerileri kullandınız?

Nesne Yönelimli Programlama dersinde öğrendiğimiz nesne kavramını, Veritabanı Yönetimi dersinde öğrendiğimiz tablo ilişkilerini ve daha bir çok bilgiyi proje içerisinde kullanmaktayız.

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

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

a) Ekonomi

Firestore hizmetleri için belirli bir kullanım kapasitesinden sonra aylık ücret istemektedir.

b) Çevre sorunları:

Kısıt yoktur.



c) Sürdürülebilirlik:

Uygulamanın tüm Android cihazlarda çalışabilmesi için API ve SDK sürümleri kontrol edilerek işlemler yapılıyor.

d) Üretilbilirlik:

Yeni bir pazar uygulaması oluşturuldu.

e) Etik:

Kısıt yoktur.

f) Sağlık:

Kısıt yoktur.

g) Güvenlik:

Güvenlik problemi oluşturmamaktadır.

h) Sosyal ve politik sorunlar:

Kısıt yoktur.