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

**PROJENİN KONUSU:**

# Kan Bağışını Yaygınlaştırabilecek Mobil Uygulama ve Web Sitesi Geliştirme

## Yazılım Mühendisliği Mimari Tasarım Raporu

**Ufuk Bulut (394811)**
**Osman Can Aksoy (394797)**
**Hüdahan Altun (394753)**

İçindekiler

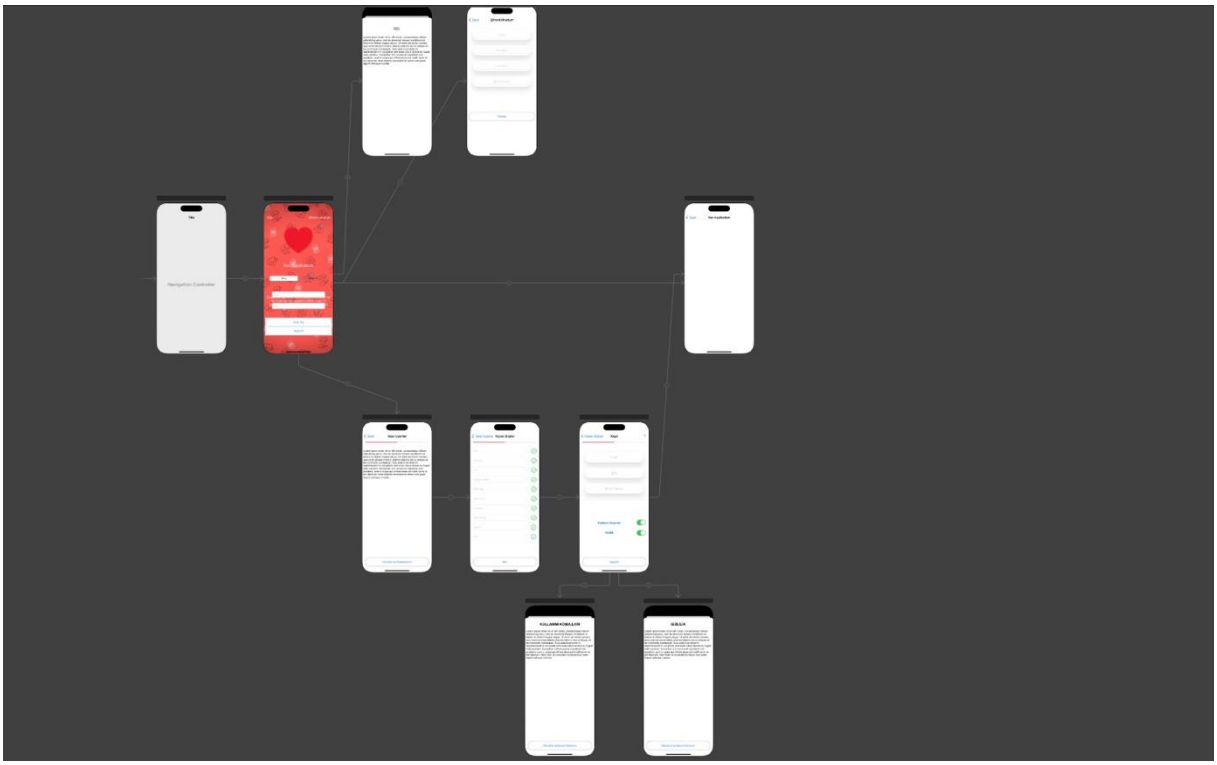
# 1.Ana Sınıflar

## 1.1 Mobil Uygulama Sınıfları

Mobil uygulamamızda bir kullanıcının ziyaret edebileceği kadar yaklaşık 14 adet ViewController sınıfı tanımlanması planlanmıştır. Ayrıca bu sınıflar arası hiyerarşiyi destekleyecek 3 adet swift sınıfı tanımlanması planlanmıştır.

Uygulama geliştirme süresince bu sınıfların içerikleri değişebilir. Yeni sınıflar eklenebilir veya mevcut sınıflar silinebilir.

Aşağıda şimdiye kadar geliştirilen sayfaların bağlantıları verilmiştir.



### 1.1.1 View Controller Sınıfları

Bu başlık altında uygulama geliştirme süresi boyunca geliştirilmesi planlana sınıflar belirtilmiştir.

- GirisVC

  Uygulamanın ilk karşımıza çıkan ekranıdır .Bu sınıfta temel olarak kayıt olma, giriş yapma, şifre sıfırlama ve sıkça sorulan sorulara ulaşabilirsiniz.

- SSSVC

  Kan bağışı ile ilgili sıkça sorulan soruların barındığı sınıftır.

- SifreYenVC

  Kullanıcının şifreyi unutması durumunda yenileyeceği sınıftır..

- YasalVC

  Kullanıcıların uygulamaya kayıt olurken kabul edeceği yasal uyarıları barındıran sınıftır.

- KisiselVC

  Kullanıcının uygulamaya kayıt olurken kişisel bilgilerinin alınacağı sınıftır.

- KayitVC

  Kullanıcın uygulamaya kayıt olması için gereken mail adresi ve şifresinin alındığı sınıftır.

- KullanimKosVC

  Kullanıcıya kullanım koşullarının gösterildiği sınıftır.

- GizlilikVC

  Kullanıcıya gizlilik esaslarının gösterildiği sınıftır.

- MainVC

  Kullanıcının kayıt olduktan sonra veya giriş yaptıktan sonra göreceği sayfanın sınıfıdır.

- ProfilVC

  Kullanıcının Kendine ait profil bilgilerinin görüleceği sınıftır.

- BagisVC

  Kullanıcının bulunduğu konumu temel alarak kan bağışı yapması için gereken kendine yakın kurumların bilgilerinin gösterildiği sınıftır

- BildirimVC

  Acil Kan İhtiyacı bildirimi  oluşunca kullanıcının bu bildirime tıklamasıyla uygulamada açılacak olan sayfanın sınıfıdır.

- AyarlarVC

  Hesap ayarlarının görüleceği sınıftır.

- HaritaVC

  Kullanıcının Bağış noktalarını göreceği sınıftır

### 1.1.2 Yardımcı Sınıflar

- Constants

  Uygulamada yazım hatalarının çıkartabileceği çökmeleri önlemek için oluşturulan sınıftır.

- Kisi

  Kullanıcının kişisel bilgilerinin nesnelerinin üretileceği sınıftır. Bu sınıfı veritabanına veri göndermek için kullanacağız

- Veritabanı

  Veri tabanı sorgularının yapılacağı sınıftır

## 1.2 Web Sitesi Mimarisi

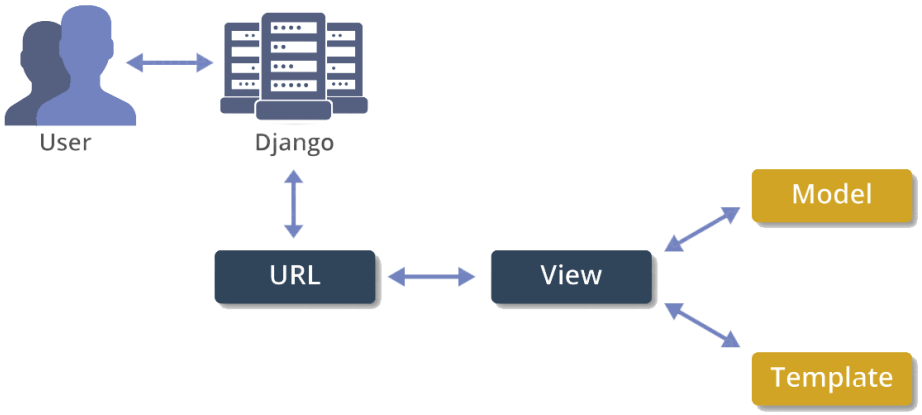
Web sitemizi geliştirirken Django Framework' ü ile çalışmaktayız. Bu frameworkü kullanmayı tercih etmemizdeki en önemli sebep az kodla az zamanda diğer programcılar tarafından kolayca okunabilir ve modüler bir yazılım ortaya çıkartmaktır. Django frameworkü MVT mimarisiyle kod yazmamızı sağlayan bir araçtır. MVT mimarisinde Model, View, Template katmanları kullanılmaktadır. Bu katmanlara kısaca değinecek olursak;

**Model :** Veritabanı işlemlerinin yapıldığı katmandır. Django'da SQL komutlarına ihtiyacınız yoktur. Django'ya özel bir dille veritabanınızı kolaylıkla oluşturabilirsiniz. Default olarak Sqlite veritabanı modeli ile karşımıza çıkmaktadır. Ancak bizim projemizde mobil tarafı ile haberleşmenin daha hızlı ve etkili yapılabilmesi için Firebase kullanılacaktır. Firebase kullanımı hakkındaki detaylı inceleme veritabanı başlığı altında açıklanacaktır.

**View :** Template ile Model arasında ki köprüdür. Uygulamadaki python kodlarımızın hepsi view de yer alır. Model ve Template de kullanmak istediğimiz yerleri python kodlarımızla çekerek view de kullanırız.

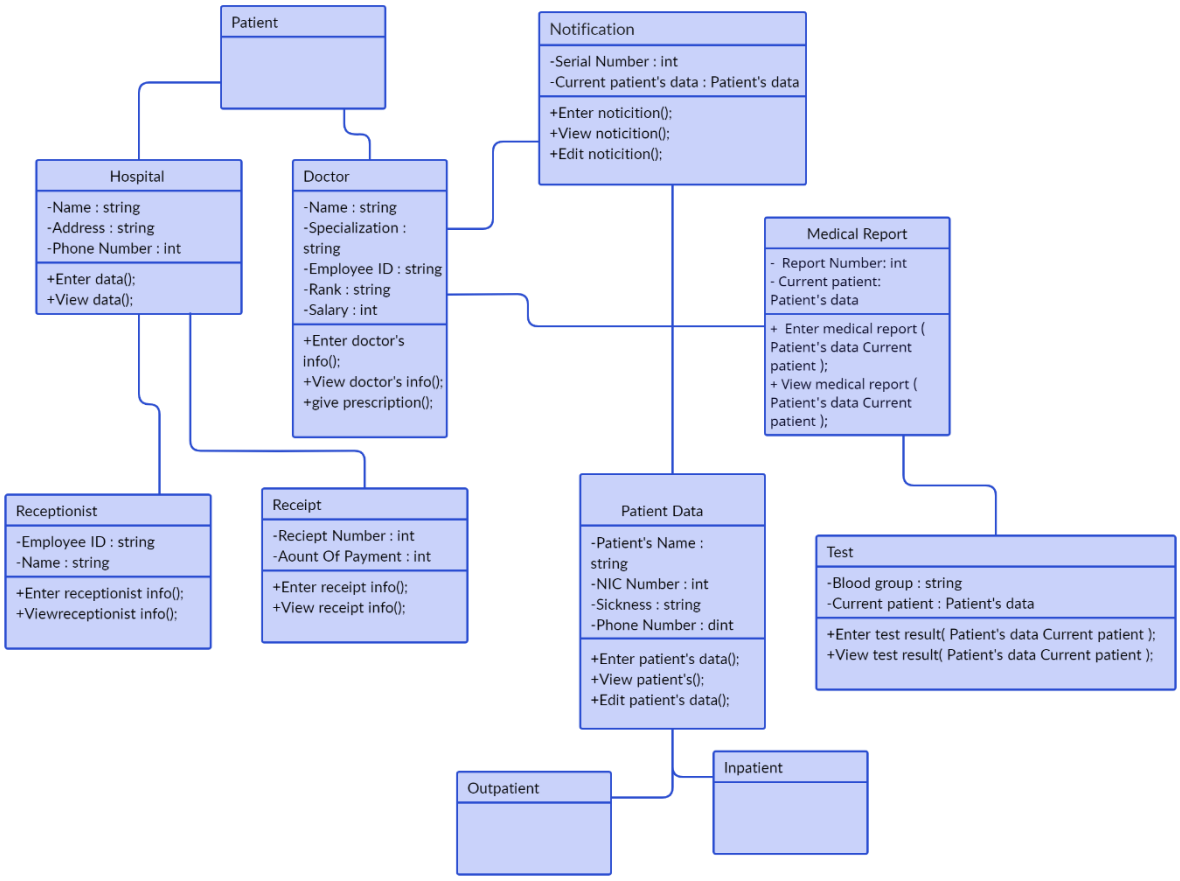
**Template :** Web programlamayla uğraşan kişiler için bu kelime yabancı değildir . Bu katman bizim sunum katmanımızdır .Sayfamızın nasıl görüneceğine dair kısımlar bu katmandadır. İçerisinde .html uzantılı dosyalar bulunmaktadır ve bu dosyalar aslında kullancı seviyesindeki uygulamanın arayüz tasarımını da oluşturmaktadır.

Django frameworkünün başlıca diğer avantajlarında ise Django ile karmaşık veritabanı işlemlerini kolayca halletmemizi sağlamasıdır, ayrıca büyük trafikler içerebilecek projemizde programları çalıştırabilme ve tüm bu yükü kaldırma konusunu da rahat bir şekilde başarıyla tamamlayacaktır. Ayrıca web sitesi kısmını hastanelere entegre edeceğimiz için yönetim panelinin dinamik olması gerekmektedir. Django içerisinde CRUD işlemlerini içeren dinamik bir yönetim paneli içerir.




### 1.2.1 Web Site Sınıfları

Django framework yapısında class bağlantıları ve kullanımları models tarafında Gerçekleştirilmektedir. Burada arkaplanda çalışacak ayrıca sistem girdilerindeki kontrolleri yapacak sınıflar bulunacaktır. Aşağıda kurgulanması hedeflenen sınıf yapılarının sınıf diyagramları verilmiştir. Hala proje geliştirilme aşamasında olduğu için aşağıdaki diyagram yapılması hedeflenen sınıfları göstermektedir.




## 2. Veritabanı

Veritabanı kısmında django frameworkünde default olarak sqlite gelse de bizim projemizde mobil kısım ile haberleşmenin daha sağlıklı kurulabilmesi için firebase teknolojisini kullanmaktayız. Kurmuş olduğumuz django site kodlarına aşağıdaki gibi firebase bağlantımızı gerçekleştirdik. Veriler firebase ortamına yazılıp okunabiliyor durumdadır. Bu aşamada karşımıza çeşitli sorunlar çıksa da bu sorunları çözebilmek için aktif olarak çalışıyoruz.

```
config = {
    "apiKey": "AIzaSyByNNvivikZvtOHq91g8wI_gnKLqb-t5EQ",
    "authDomain": "cpanel-47999.firebaseapp.com",
    "databaseURL": "https://cpanel-47999-default-rtdb.firebaseio.com",
    "projectId": "cpanel-47999",
    "storageBucket": "cpanel-47999.appspot.com",
    "messagingSenderId": "937692790342",
    "appId": "1:937692790342:web:0a9bab056360f3af805447"
}
```

```
firebase=pyrebase.initialize_app(config)
authe = firebase.auth()
database= firebase.database()
```

## 3. İş Kuralları

---

### 3.1 Genel Sistem Kuralları

Genel Yazılım sisteminin en temek kuralı kullanıcılara hızlı ve sürekli olarak hizmet vermektir. Bu kuralı yerine getirmeye çalışırken yazılım çalışacağı donanımların ve haberleşme ağının durumları göz önünde bulundurulur.

Projemizin en kritik İş Kuralı Emergency Blood Broadcast System (EBBS) dediğimiz sistemin sürekliliği esastır. Bu İş kuralında kan bağışı alan kurumların kullanacağı web sitesi ile bağışçıların kullanacağı uygulama arası iletişim gerçek zamanlı olmalıdır.

### 3.2 Mobil Sistem Kuralları

Mobil uygulamamızı kullanan kullanıcılar:
- Güvenli Şekilde Hesap oluşturabilmeli
- Güvenli şekilde kimlik doğrulama ile hesaplarına giriş yapabilmeli
- Kullanıcılar güvenli şekilde çıkış yapabilmeli
- Kullanıcılar Profil Bilgilerine erişebilmeli ve düzenleyebilmeli
- Kullanıcılar Bağış Noktalarını Görebilmeli
-EBBS ile gelen bildirimi görüp yanıt verebilmelidir.

### 3.3 Web Sistem Kuralları

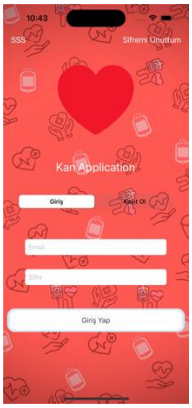
Hastanede Sistem Yönetiminden Görevli Personel:
-Güvenli şekilde hasta kayıt işlemi yapabilmelidir
-Personel güvenli şekilde kimlik doğrulama ile sisteme giriş yapabilmeli
-Çevresindeki Bağışçıların durumunu görebilmelidir.
-İhtiyaç olan durumda çevredeki kullanıcılara bildirim yollayabilmelidir.
-Bildirimi onaylayan kullanıcı bilgilerini görebilmeli ve kabul eden kişinin durumunu takip edebilmelidir

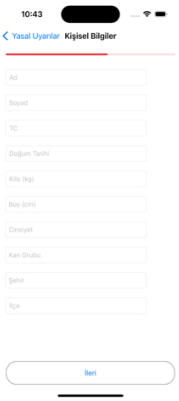
# 4. Kullanıcı Arayüzü

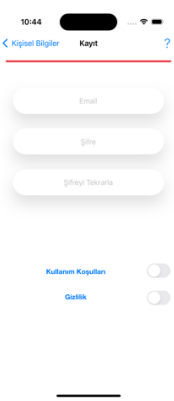
Yazılım sistemimiz mobil – web paylaşımlı bir sistemdir

## 4.1 Mobil Arayüz

Aşağıda uygulamamızın bazı sayfalarının arayüz tasarımları mevcuttur.



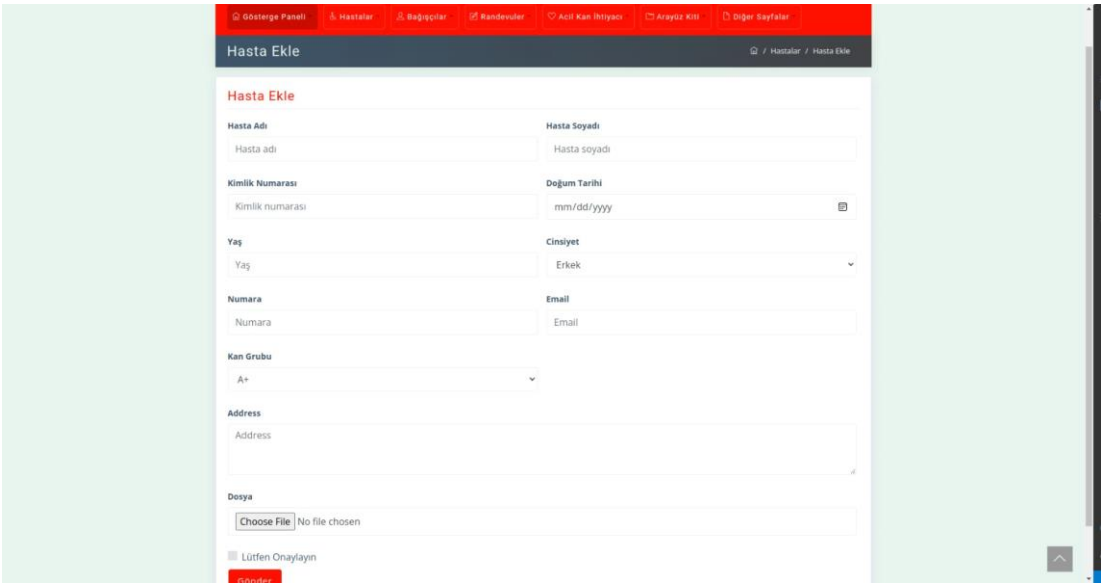




## 4.2 Web Ugulaması Arayüz








Web ve Mobil arayüz kısımları halihazırda geliştirilmeye devam edilmektedir. Bu arayüz tasarımları henüz prototip aşamasındadır.

## 5. Kaynak Yönetimi ve Performans Beklentileri

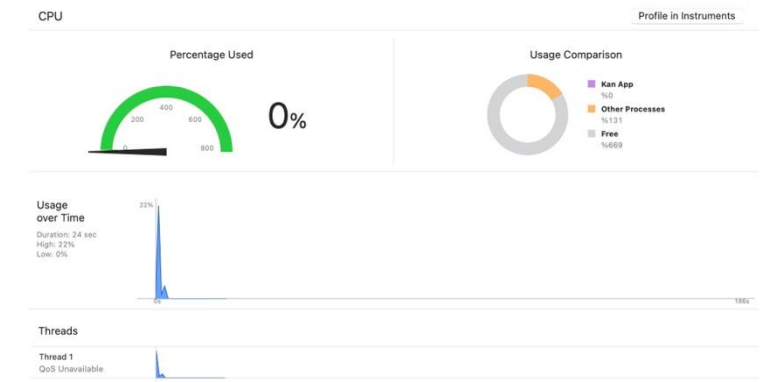
---

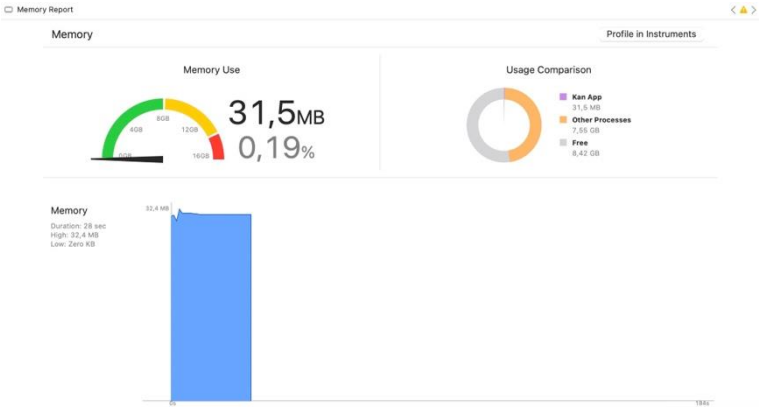
### 5.1 Mobil Uygulama Kaynak Kullanımı

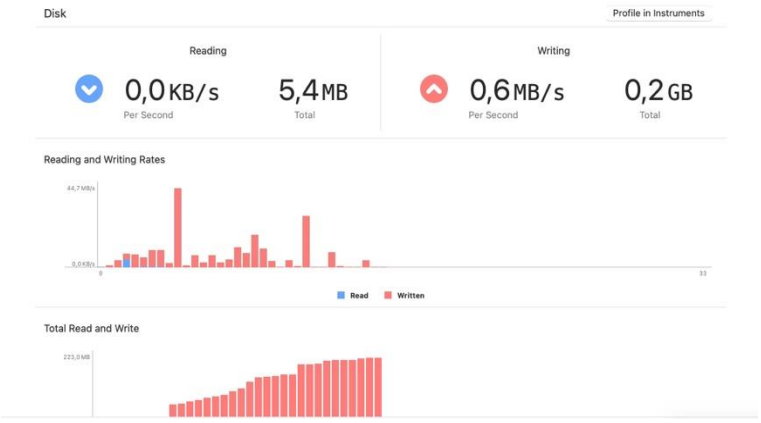
Amacımız uygulamamızın çalıştığı cihazlarda yüksek verimi elde etmesidir.

Şimdiye kadar olan geliştirme aşamasında XCode geliştirme ortamının bize sağlamış olduğu uygulamaya ait bellek, CPU ve disk kullanım miktarları aşağıda mevcuttur.

Geliştirme devam ederken veri miktarına bağlı olarak bellek, CPU ve disk kullanımlarında artış gözlenebilir. Threadler sayesinde bu kullanımları optimum seviyeye getirmeyi hedefliyoruz.



## 7.Genişletilebilirlik

Mobil uygulama geliştirirken mimarimiz MVC mimarisi olup bu mimaride kullanıcı arayüzü, veritabanı işlemleri ve sınıf kontrollerinin yapılacağı yapılar birbirinden büyük oranda bağımsızıdır. Web arayüzünde de benzer bir mimari olan MVT kullanılacağı için arayüz, veritabanı işlemleri ve sınıf kontrollerinin yapılacağı yapılar birbirinden farklıdır.

Katmanlar arasında ki bağımsızlık uygulamaya yeni özellikler eklerken avantaj sunar. Mobil uygulamamızda yeniliklere açık bir uygulamadır.

İleride Uygulamaya gelebilecek yenilikler:

-Kök Hücre Bağışı
-Kemik İlik Bağışı
-Organ Bağışı
-Bağışçının yaptığı bağış oranına göre sağlayacağı yasal avantajların gösterimi

## 8.Yerelleştirme

Projemizin testleri için seçilen pilot şehir Trabzon ilimizdir. Uygulamanın tam sürümü tüm Türkiye sınırları içinde aktif olacaktır. Uygulamamız Türkiye sınırları içinde çalışacağı için mevcut dil desteği Türkçedir.

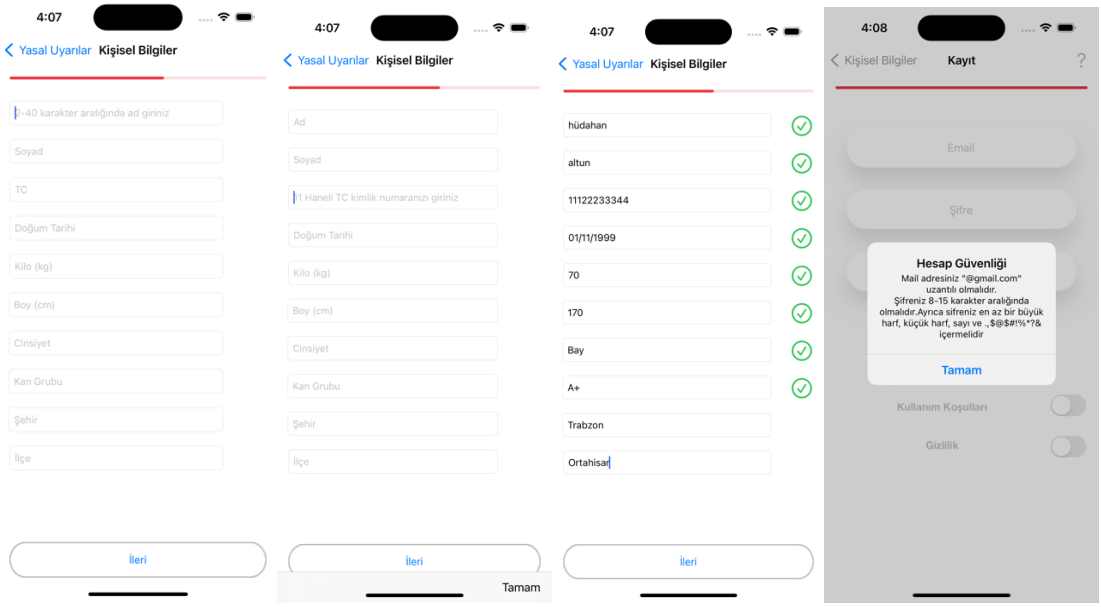
## 9.Hata İşlemleri

Uygulamanın kullanımında kullanıcının yapmasını istemediğimiz işlemleri yapması mümkündür. Bu yüzde uygulamanın o sayfasının arka planında çalışan sınıfında bazı kontrol işlemleri yapmak zorunludur. Kullanıcıya yapılmasını istemediğimiz durumlarda bildirim gösterilmesi veya yapmasını istediğimiz işlemlerle ilgili bildirim gönderilmesi gerekir.

Uygulamanın veri haberleşme işlemlerinde alınan veya gönderilen verinin doğru bir şekilde gerçekleşip gerçekleşmediğinin de bildirimleri yapılmalıdır.

Bu hata yakalama işlemini swift sınıflarında if bloklarıyla veya try-catch yapısıyla gerçekleyebilir ve bunun bildirimlerini kullanıcıya gösterebiliriz.

Aşağıda uygulamamızın kullanıcı bildirimlerinde bir kısmı gösterilmektedir




Ayrıca aşağıda kayıt sayfasının güvenlik kontrolünün gösterildiği kod bloğunun bir parçası mevcuttur.

```
extension KayitViewController{

    func kayitVerileriniAl(emailTF:UITextField,sifreTF:UITextField,sifreTkrTF:UITextField){


        if let mail = emailTF.text, let sifre = sifreTF.text, let sifreTkr = sifreTkrTF.text{

            if mailGlobalCheck == true && sifreGlobalCheck == true && sifreTekGlobalCheck == true{
                // Tf içerikleri artık dolu olduğu kesindir.1.asama güvenlik

                let check1 = alınanMailGuv(mail: mail)
                let check2 = alınanSifreGuv(sifre: sifre)
                if check1 == true && check2 == true && sifre == sifreTkr{
                    //2.asama güvenlik
                    //mail ve sifre istenen özelliklere sahip ayrıca sifre ile sifre tekrarı bir eşit.
                    performSegue(withIdentifier: K.kToMain, sender: nil)

//                    kayitOlButton.isEnabled = true // buton aktif artık kayıt ol butonuna basılabilir

                    print("diğer sayfaya güvenli gecis")


                }else{

                    bilgilerYanlisAlert()
```

Mobil kısıma benzer şekilde güvenlik kontrolleri web arayüzünde de yapılacaktır. Projenin ilerleyen süreçlerinde kullanım testleri yapıldıktan sonra ortaya çıkabilecek güvenlik zaafiyetlerini ortadan kaldırmaya yönelik iyileştirmeler yapılacaktır.

## 10. Dayanıklılık ve Mimari Fizibilite

---

Projemizden genel olarak beklentimiz EBBS siteminin doğru bir şekilde çalışmasıdır.

Mobil uygulamamızdan beklentilerimiz ise ios 12 ve üzeri sürümlerinin yüklü olduğu iphone cihazlarında stabil şekilde çalışması ve kullanıcı arayüzünün farklı cihaz boyutlarına göre yapılandırılması.

Web uygulamasında sistem gereksinimleri minimum  düzeyde tutularak sistem yöneticisi olan kişinin kolay ve sade bir arayüz ile işlemlerini hızlıca gerçekleştirebilmesini sağlamaktır.