**MMOT-225 - Bulut Bilişim**

**Haber Uygulaması**

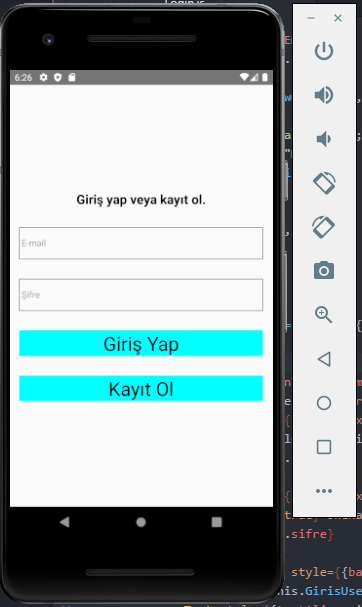
**Öğretim Görevlisi: Levent Yıldız**

**Öğrenci: Batın Çetin**

**Öğrenci No: 19MY03004**

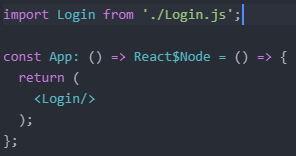
**Uygulamaya kayıt olan kullanıcı için; haberleri listeleyip içlerinden seçilenlerin beğenilmesi, daha sonra okumak için kaydedilmesi gibi işlevleri barındıran haber uygulaması.**

**Login Ekranı**

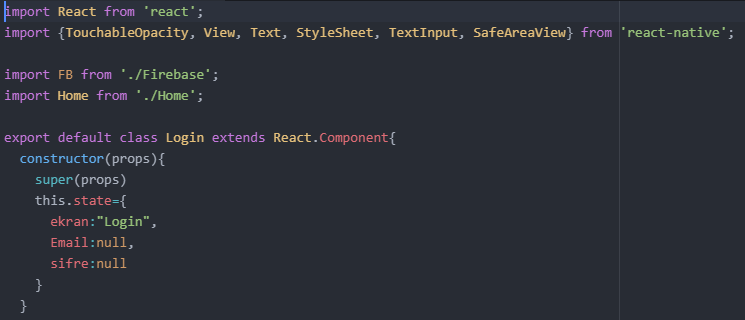


**Kullanıcıyı ilk başta Login ekranı karşılamaktadır. Kullanıcı kayıt olduktan sonra giriş yaparak uygulamaya erişim sağlayabilir.**

**Login Ekranı Kodları**



**Login Ekranı, App.js içerisinde Custom Component olarak çağrılıyor.**



**Gerekli componentler import ediliyor. Login işlemi için Firebase, Login işleminden sonra ilerlemek için Home ekranının componenti importlanıyor. Constructor içerisinde default sayfa olan Login ekranı ve kullanıcının giriş bilgilerini tutmak üzere Email ve sifre değerleri this.state içerisinde tanımlanıyor.**

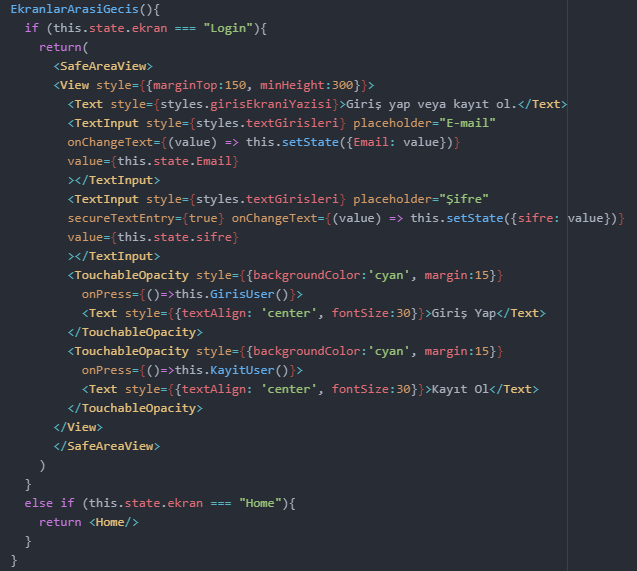
**Login Ekranı Kodları - Fonksiyonlar**



**Firebase’e login olmamızı sağlayacak olan fonksiyonlar oluşturuluyor. KayitUser, kayıt olmak için; GirisUser ise giriş yapmak için. Authentication işlemleri yapılıyor. Giriş başarılı olursa ekran değeri “Home” olarak ayarlanıyor.**

**Login Ekranı Kodları**

**Ekranlar Arası Geçiş Fonksiyonu**



**EkranlarArasiGecis fonksiyonu içerisinde state içerisinde bulunan ekran değeri kontrol ediliyor ve eğer “Login” ise, login ekranında kalınıyor. “Home” ise haberlerin bulunduğu navigasyon ekranına geçilecek.**

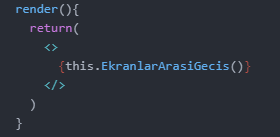
**Login Ekranı Kodları**

**Ekranlar Arası Geçiş Fonksiyonu**

**Return Açıklaması**

**Fonksiyonun Return()’u içerisinde bir SafeAreaView içinde bir View bulunuyor. Bu View içerisinde Email ve Şifre için 2 adet TextInput Component, 2 adet de “Kayıt Ol” ve “Giriş Yap” butonları için TouchableOpacity bulunuyor.**

**Login Ekranı Kodları - Render(){}**



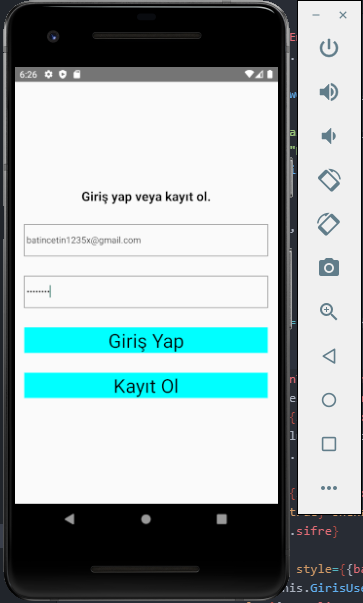
**Login ekranının default Render(){} metodu içerisinde EkranlarArasiGecis fonksiyonu çağrılıyor.**

**Login Ekranı Kodları - StyleSheet**



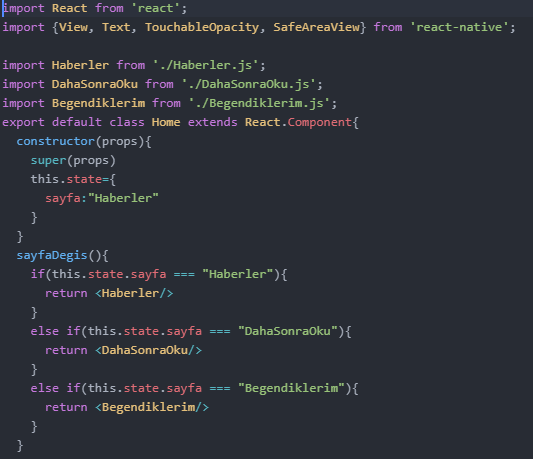
**Login ekranında Email ve Şifre yazılan TextInput Componentleri “textGirisleri” stili ile şekillendiriliyor. Giriş ekranında bulunan “Kayıt olun veya giriş yapın.” yazısı da “girisEkraniYazisi” stili ile şekillendiriliyor.**

**Login Ekranı - Giriş Yapıyorum**



**Yukarıda verilen değerlerle önce kayıt olup sonra giriş yapıyorum.**

**Home Ekranı Kodları**



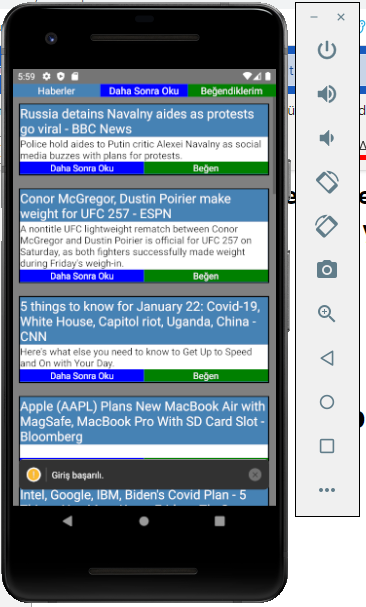
**Gerekli bütün componentler import edildikten sonra, this.state içerisine sayfa değeri tanımlanıyor, SayfaDegis fonksiyonu ile bu değere göre sayfa değişimi sağlanacak.**

**Home Ekranı Kodları - Render()**



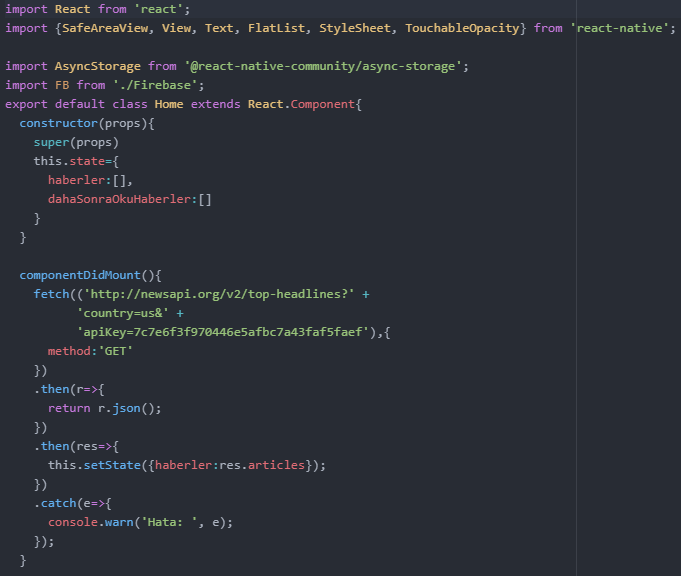
**SafeAreaView içerisinde bulunan View’da 3 adet TouchableOpacity butonu bulunuyor. Bu butonlar sayfalar arası geçiş yapmayı sağlıyor. En sonda sayfaDegis() metodu çağrılıyor, böylece herhangi bir butonu basıldığında sayfa değişecek ve butonların altında o sayfa gözükecek.**

**Haberler Sayfası - İlk Giriş**



**Kullanıcı ilk giriş yaptığında, onu 3 sayfadan biri olan Haberler sayfası karşılamaktadır. Yukarıdaki resimde ayrıca, girişin başarılı olduğunu gösteren mesaj bulunmaktadır.**

**Haberler Sayfası Kodları**



**Gerekli şeyler import edildikten ve this.state ayarlandıktan sonra, componentDidMount() fonksiyonu içerisinde fetch ile** [**http://newsapi.org**](http://newsapi.org) **adresinden haberler çekiliyor.**

**Haberler Sayfası Kodları**

**AsyncStorage**



**Daha Sonra Oku ekranına kaydedilmek üzere tıklanan haber AsyncStorage’ya kaydediliyor. Eğer haber zaten kayıtlıysa kaydedilmiyor. AsyncStorage’dan dönen null değerlerini engellemek için bir if else yapısı bulunuyor.**

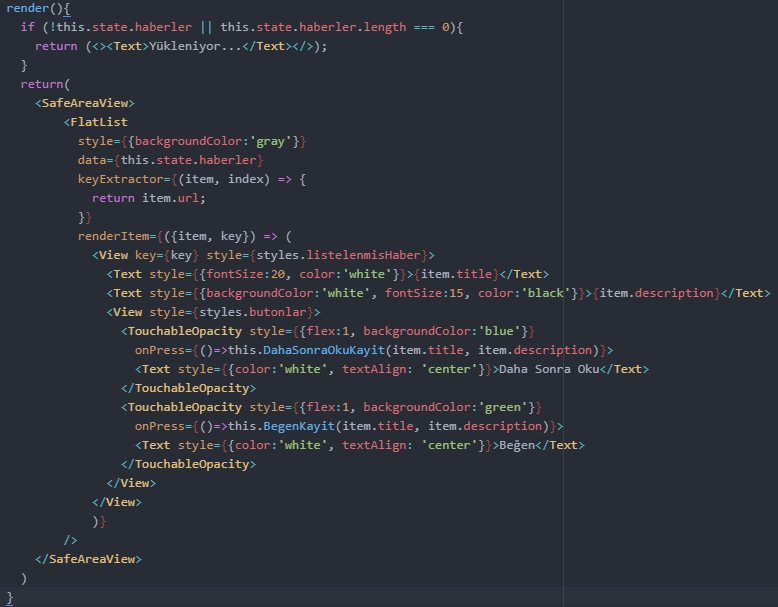
**Haberler Sayfası Kodları**

**Firebase Realtime Database**



**Beğendiklerim ekranına kaydedilmek üzere tıklanan haber Firebase Realtime Database içerisine kaydediliyor. Zaten beğenilmiş olan haber tekrardan beğenilmiyor.**

**Haberler Sayfası Kodları - Render(){}**



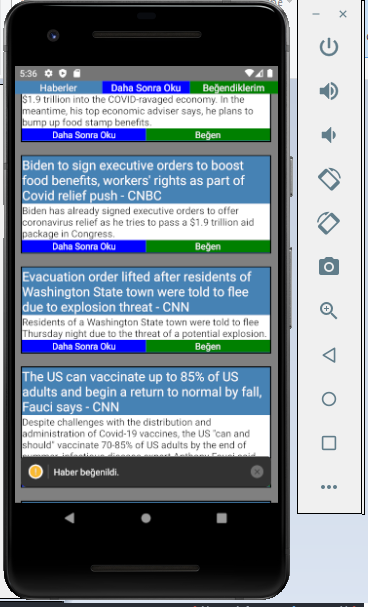
**Haberler yüklenene kadar bir conditional render yapılıyor, haberler yüklendiğinde bir FlatList içerisinde haberler listeleniyor. Her listelenen item, bir haber başlığı, bir haber içeriği ve 2 adet “Daha Sonra Oku” ve “Beğen” butonu eklenerek render ediliyor.**

**Haberler Sayfası Kodları - StyleSheet**



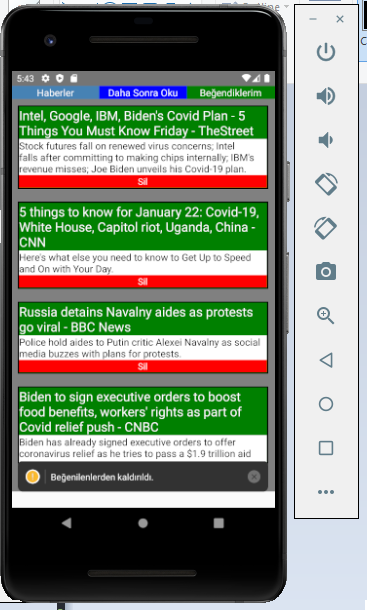
**Haberler sayfasında bulunan listelenen haberlerin View’ları için stiller belirleniyor, varolan 2 butonun yatay olarak hizalanması sağlanıyor.**

**Haberler Sayfası - Beğen**



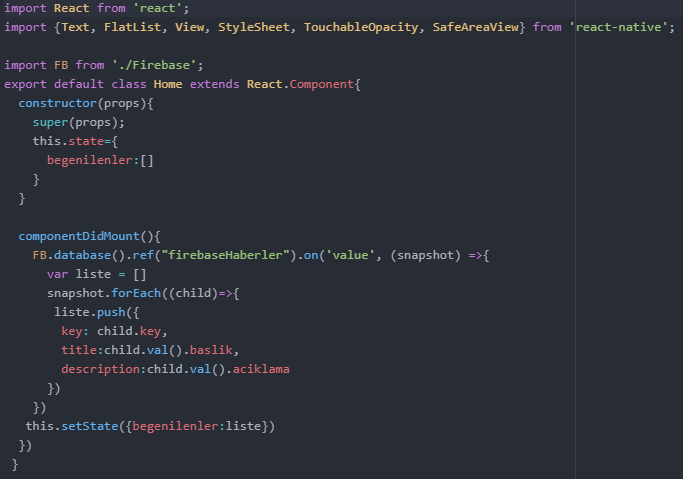
**Herhangi bir haber beğenildiği zaman, beğenilen haber, “Beğendiklerim” sayfasına kaydedilir ve kullanıcıya o haberi beğendiğine dair bir mesaj gösterilir. Eğer kullanıcı o haberi daha önceden beğenmiş ise, herhangi bir mesaj gösterilmez ve tıklanan haber tekrar kayıt edilmez. “Daha Sonra Oku”ya tıklanırsa da aynı işlemler gerçekleşir.**

**Beğendiklerim Sayfası**



**Yukarıda “Beğendiklerim” sayfasının içeriği gösterilmektedir. Kullanıcının daha önceden beğenmiş olduğu haberler arasından biri silindiğinde yukarıda gösterildiği gibi, haberin silindiğine dair bir mesaj gösterilir.**

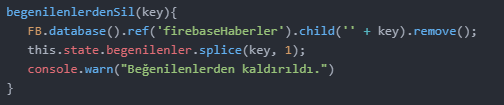
**Beğendiklerim Sayfası - Kodlar**



**Gerekli componentler import edildikten ve this.state ayarlandıktan sonra componentDidMount() fonksiyonu içerisinde Firebase Database’den kayıtlı bilgiler listelenmek üzere çekiliyor ve state içerisine kaydediliyor.**

**Beğendiklerim Sayfası - Kodlar**

**Beğenilen Haberi Kaldır**



**Beğendiklerim sayfasında listelenmiş olan haberlerden birine tıklandığında o haberin kaldırılması için bu fonksiyon kullanılıyor. Önce Firebase Database içerisinden silinecek olan haber key value’sine göre silinip, daha sonra state içerisinden haber silinerek sayfanın güncellenmesi sağlanıyor.**

**Beğendiklerim Sayfası - Render(){}**



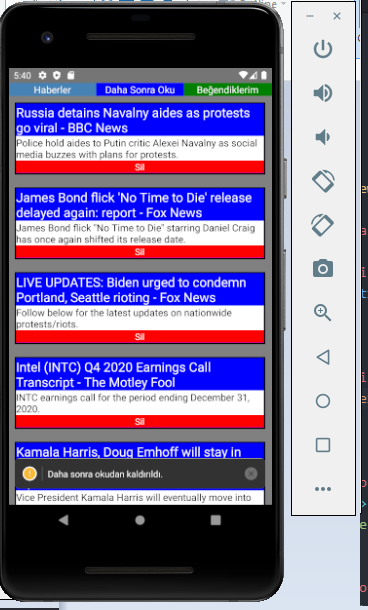
**Beğenilen haber henüz bulunmuyorsa, conditional rendering ile bu belirtiliyor. Eğer Firebase Database değerleri state’a yüklendikten sonra, state içerisinde, beğenilen haber var ise, Firebase Database içerisinde bulunan beğenilmiş haberler FlatList component ile renderlanıyor. Her listelenen item için, başlık, içerik ve sil butonu bulunuyor.**

**Beğendiklerim Sayfası - StyleSheet**



**Firebase Database içerisinden çekilen ve listelenen her haber için haberin bulunduğu View, “listelenmisHaber” stili ile şekillendiriliyor.**

**Daha Sonra Oku Sayfası**



**“Daha Sonra Oku” sayfasının içeriği gösterilmektedir. Daha sonra okumak üzere kaydedilmiş haberler burada listelenir. Bir haber buradan silindiği zaman yukarıdaki gibi mesaj gösterilir.**

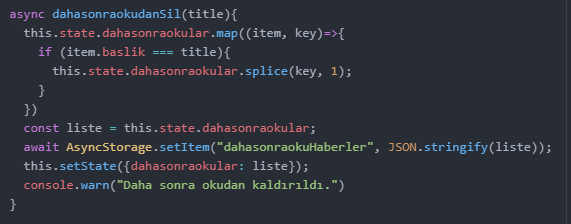
**Daha Sonra Oku Sayfası - Kodlar**



**Gerekli componentler ve AsyncStorage import edildikten sonra constructor içerisinde verileri tutmak üzere tanım yapılıyor. Daha sonra componentDidMount() metodu içerisinde async ve await ile AsyncStorage içerisinde kayıtlı bulunan veriler çekilip state’a aktarılıyor.**

**Daha Sonra Oku Sayfası - Kodlar**

**Daha Sonra Oku’dan Kaldır**



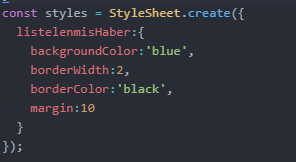
**Fonksiyon içerisinde state içerisinde bulunan haberler mapleniyor ve eğer fonksiyona parametre olarak verilen başlık state içerisinde bulunuyorsa, bu veri state değerinden splice ile kaldırılıyor. Daha sonra AsyncStorage içerisine yeni state değeri setleniyor.**

**Daha Sonra Oku Sayfası - Render(){}**



**Conditional Rendering yapıldıktan sonra, AsyncStorage’den alınan değerleri barındıran state bu sefer FlatList’e aktarılmak yerine ScrollView içerisinde maplenerek ekranda gösteriliyor. Maplenen her item bir haber başlığı, bir haber içeriği, Daha Sonra Oku’dan kaldırmak üzere TouchableOpacity ile bir sil butonu barındırıyor.**

**Beğendiklerim Sayfası - StyleSheet**



**Daha Sonra Oku’ya kaydedilmiş olan her item maplendiğinde kullanılan View componentini şekillendirmek üzere “listelenmisHaber” stili oluşturuluyor ve her maplenen item bu stil ile şekillendiriliyor.**