T.C.
CUMHURİYET ÜNİVERSİTESİ
MÜHENDİSLİK FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

TARIMSAL YÖNETİM SİSTEMİ(TYS)

RAMAZAN SÖNMEZ

LİSANS BİTİRME PROJESİ

DANIŞMAN
Dr. Öğr. Üyesi Halil ARSLAN

HAZİRAN-2019
SİVAS

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

İmza

Ramazan SÖNMEZ

Tarih: 03.07.2019

ÖZET

TARIMSAL YÖNETİM SİSTEMİ(TYS)

Ramazan SÖNMEZ

Danışman: Dr. Öğr. Üyesi Halil ARSLAN

Tarımsal Yönetim Sistemi (TYS), tarlası olan kullanıcıların kendilerine ait oluşturdukları hesap ile sahip oldukları tarlalarına ait genel bilgilerini girerek tarlalarını oluşturup daha sonra bu oluşturulan tarlalarına ait bilgileri istedikleri zaman takip edebildiği react native uygulama çatısı ile program kodları yazılmış android ve ios işletim sistemlerinde çalışabilen mobil uygulama projesidir.

Anahtar Kelimeler: mobil uygulama, android, ios, tarım, react native, javascript

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

Kısaltmalar

TYS	:Tarımsal Yönetim Sistemi
UI	:User Interface / Arayüz
SDK	:Software Development Kit/ Yazılım geliştirme kiti
IDE	:Integrated Development Kit /Tümleşik geliştirme ortamı
SQL	:Structed Query Language / Yapılandırılmış sorgu dili
JSON	:JavaScript Object Notation

1. GİRİŞ

1.1. Projenin Amacı

Tarımsal Yönetim Sistemi (TYS), belirledikleri konumdaki tarlası olan çiftçilerin tarlalarına ektikleri ürünlerin genel durumunu gözlemleyebilmeyi hedef alan bir mobil uygulamadır.

1.2. Projenin Hedefi ve Başarı Kriterleri

-Kullanıcının yeni tarla ekleyebilmesi

-Kullanıcının eklediği tarlaya ait konumunu belirleyebilmesi

-Belirlediği konumdaki tarlaya ürün istediği ürün bilgilerini girerek kaydedebilmesi

- Sahip olduğu tarlaları görüntüleyebilmesi

-Tarlalarında ekili olan ürünleri ve ürünler hakkında detaylı bilgileri(Ekim yeri, ekim tarihi, hasat tarihi, ekildiğinden itibaren geçen süre, hasat için kalan süre, maliyet, ekilen kg, tarım tipi, gübre çeşidi, ekildiği alan) görebilmesi

-Tüm tarlaların maliyet ve karının hesaplanması

2. KAYNAK ARAŞTIRMASI

Proje ile ilgili gerekli teorik çalışmalar internetteki kaynaklardan ve ilgili kişilerle görüşülerek yapıldı.

3. MATERYAL VE YÖNTEM

3.1. Mobil uygulamada kullanılacak teknolojiler nelerdir?

3.1.1. React-Native

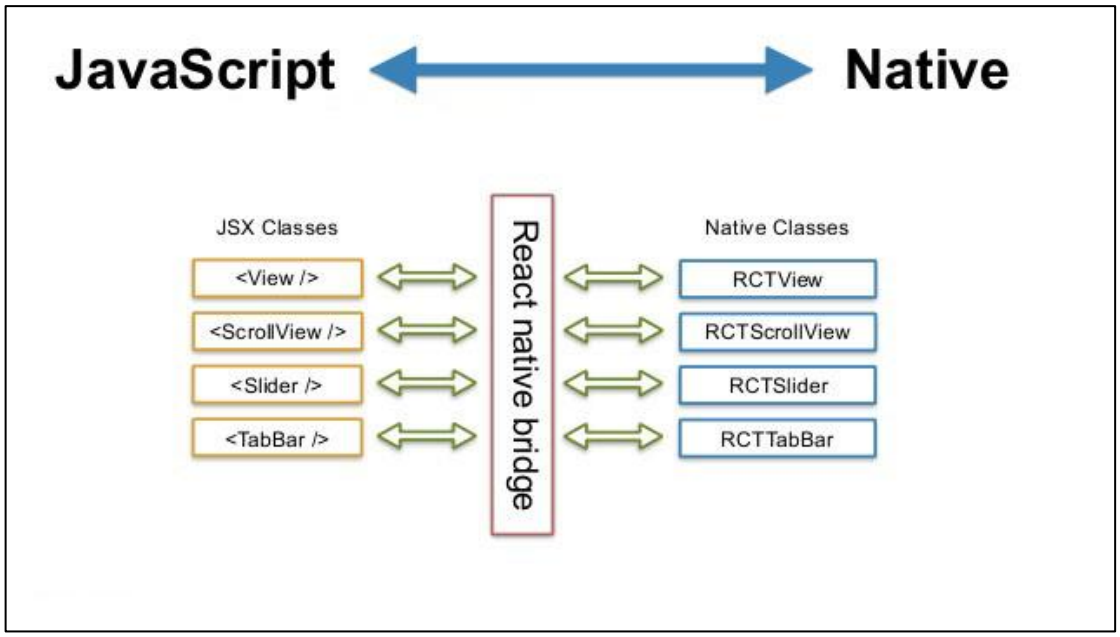
JavaScript programlama dilini kullanarak hem android hem de ios uygulamaları geliştirebileceğiniz facebook tarafından geliştirilen bir native mobile platformudur.

React Native, facebook'un web tarafında kullandığı react sistemi üzerine inşaa edilmiş, javascriptin her zaman programlama hatası gibi görülen tarafına farklı bir yaklaşım göstermiş bir kütüphanedir. Sanılanın aksine react native hybrid mobil uygulama geliştirme değildir.

React Native aslında adı üzerinde, ReactJS çatısını baz alarak Native ara yüzler aracılığıyla mobil uygulama yazılmasını sağlayan bir geliştirme ortamıdır. Native ara yüz bileşenlerini barındırdığı için klasik React'teki component yapısında kullanılan bileşeni yerine Android ve IOS'teki arayüzü karşılığına dönüşecek olan <View> bileşeni, <img>yerine ise <Image> bileşeni kullanılır. Aslında <View> ve <Image> gibi bileşenler Facebook tarafından Android ve IOS karşılıklarının kodlanmasıyla oluşturulmuş yapılardır. Yani siz bu bileşenleri kullandığınızda, çalıştırılan işletim sistemine göre arka planda gerçekten de native karşılıklarına dönüştürülürler.

3.1.1.1. React Native'in Mimarisi

WebView üzerinde görüntülenen HTML bileşenlerin aksine native olan bu yapılar tabi ki büyüsel bir şekilde oluşturulmuyor. Native tarafında eşleniklerinin üretilmesini sağlayan "bridge" dediğimiz React kodu ile native ortamı arasında bir köprü oluşturan bu yapı aslında React Native'in başrol oyunculuğunu üstleniyor.



Şekil 3.1. React Native, JavaScript kodları iletişimde bulunma şekli.

Yazılan JSX kodu, uygulamanın çalışma zamanında Android ve IOS platformları için ayrı olarak tasarlanan köprüler yardımıyla ilgili platformun Native bileşenlerine dönüştürülüyor ve uygulamanın native olarak yazılmasıyla arasında bir fark kalmıyor. Bu olayın bir dezavantajı da var elbette, çalışma zamanında native bileşene render etme operasyonu aslında CPU tüketen bir işlem. Buna bağlı olarak React Native uygulamaları, native uygulamalara kıyasla biraz daha fazla pil tüketimine sahip olacaktır. Bu fark çok küçük olduğu için genel anlamda son kullanıcının hissedeceği bir etki oluşturmayacaktır.

Aslında React Native'in, sadece UI bileşenleri ve cihaz API'larına erişim için "Native" özellikler taşıdığını belirtmekte yarar var. Zira yazdığınız JS kodu aslında uygulamanın çalıştığı cihazın JS motoru tarafından çalışıyor ve Xamarin'de olduğu gibi native kodlara dönüştürülmüyor. Bu nedenle görüntü işleme ve oyun gibi daha spesifik uygulamalar oluşturacaksanız React Native yerine ilgili platformun native geliştiriminden devam etmek çok daha mantıklıdır. Bu kullanım alanları haricinde, web servisten veri çekme, gelen veriyi UI'a basma ve çeşitli birtakım "basit" operasyonlarda React Native daha kullanışlı olacaktır.

Yazılan kodun JS kodu olmasının native koda göre en büyük avantajı belki de derleme gerektirmemesi. Bu özelliği çok iyi kullanan React Native, uygulamayı bir kez telefona atmanız halinde yapılan kod değişiklikleri "hot reloading" denilen yöntemle derleme gerektirmeksizin uygulamaya push edilebiliyor ve bu sayede yazılımcı için derle-çalıştır işleminden doğan süre kısalmış oluyor.

Web'e göre aslında React Native'in bir avantajı daha var. JS kodunun ayrı bir Thread'de çalıştığı bu sistemde, JS kodunun çalışma süresi uzasa dahi UI Thread'i çalışmaya devam ettiği için uygulama arayüzü bu durumdan etkilenmiyor ve takılmadan devam edebiliyor.

3.1.2. Proje Gereksinimleri

React Native uygulaması geliştirmek için öncelikle sisteminizde Node'un yanı sıra Android ve IOS SDK'lerinin de kurulu olması gerekiyor. Daha sonrasında react-native-cli aracını npm üzerinden indiriyoruz.

```
npm install -g react-native-cli
```

Şekil 3.2. React bileşenlerinin terminal üzerinde indirilmesi

İlgili bağımlılıklar kurulduktan sonra aşağıdaki komut ile ilk React Native projenizi oluşturuyoruz. Bu komut ile React Native projesi için gereken tüm npm kütüphanelerini yükleniyor ve bir başlangıç projesi oluşturuluyor.

```
react-native init ReactBaslangic
```

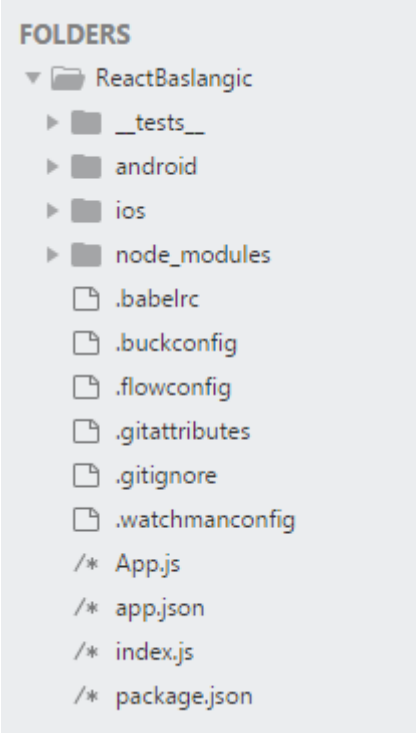
Şekil 3.3. React Native projesinin oluşturulması.

IOS projesi derlemek istiyorsak OSX işletim sistemine yani bir Mac bilgisayara ihtiyacınız var. Aksi takdirde Windows veya Linux işletim sisteminde IOS uygulaması geliştirmeniz şu an için mümkün değil.

3.1.3. Proje Yapısı

- **tests**: Facebook'un JavaScript test aracı olan Jest ile test edilecek ilgili bileşenler bu kısımda yer alıyor.
- **android/ios:** Sırasıyla Android ve IOS platformları için ayrı ayrı proje dosyaları burada tutuluyor.
- **node_modules:** Projenin bağımlı olduğu ilgili npm paketlerini barındırıyor.
- **.babelrc:** Babel JavaScript derleyicisi ile ilgili yapılandırma dosyası olan .babelrc, `"presets": ["react-native"]` satırı ile node_modules dizininde yer alan `babel-preset-react-native` plugin'inin kullanılacağını belirtiyor.

- **.buckconfig:** Facebook'un Buck isimli build sistemi ile ilgili yapılandırma dosyası olan .buckconfig, Android uygulaması için build sürümü ve maven repo url'ini barındırıyor.
- **.flowconfig:** Yine Facebook'un Flow isimli statik tip kontrolcüsü ile ilgili yapılandırma dosyası olan .flowconfig, React Native projesi ile ilgili çeşitli yapılandırma ayarlarını içeriyor.
- **.gitattributes/.gitignore**: Git versiyon kontrol sistemi ile ilgili yapılandırma dosyaları.
- **.watchmanconfig:** Facebook'un Watchman isimli dosya izleme aracı ile ilgili yapılandırma ayarlarını tutar.
- **App.js:** Örnek bir React Native uygulama dosyasıdır.
- **app.json:** Uygulama ile ilgili manifest dosyasıdır.
- **index.js:** Uygulamanın başlangıç noktasıdır diyebiliriz. İlgili bileşenler ve başlangıç bileşeni burada belirtilir.
- **package.json:** proje ile ilgili bağımlılıkların npm'e belirtildiği yapılandırma dosyasıdır. Proje adı, versiyonu, lisans bilgisi ve diğer ilgili ayarlar burada tutulur.



Şekil 3.4. React Native proje yapısı.

Node js: Uygulama geliştirirken çok fazla kullanılan yardım aracıdır. Kullanılma amacı javascript kodunu makine koduna çevirmektir.

Android Studio: Uygulamanın android kısmını ayrı olarak düzenlemek istediğimizde kullanabiliriz.

Android Sdk: React Native, Android 6.0 (Marshmallow) sürümünü istemektedir.

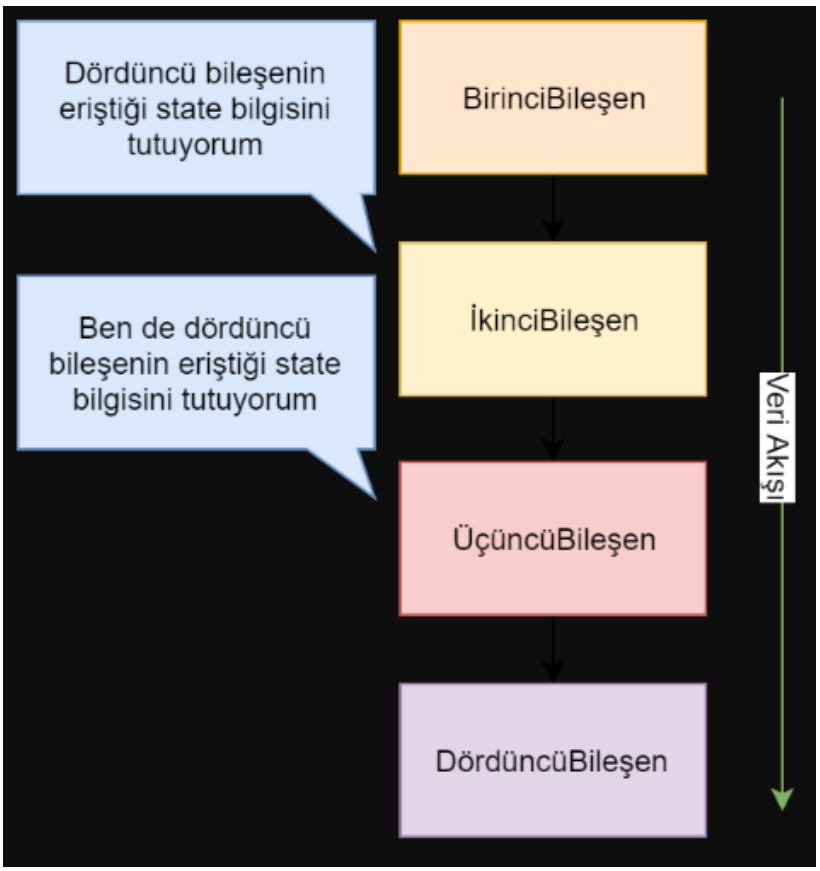
Eslint: Uygulama geliştirme aşamasında çok fazla ihtiyaç duyulan eklentidir.

Hataları derleme yapmadan yazım aşamasında görmemizi ve hız kazanmamızı sağlar.

Front-end Ide: Uygulama geliştirmek için ide olmazsa olmazlardandır. Ancak özel bir ide mevcut değildir. Öneri olarak Windows için Sublime Text ve Atom'u Mac için ise Xcode'u örnek olarak verebiliriz.

3.1.3.1. React Uygulamalarında Karşılaşılan Temel Problem

React ile kolayca başlayıp ilerlettiğiniz projede sonraki aşamalarda uygulama karmaşıklaştıkça, iç içe bileşenler arasındaki state yönetimi gittikçe güçleşir. Hangi verinin hangi bileşenden geldiğinin takibi zorlaşır. Aşağıdaki örnekte ;

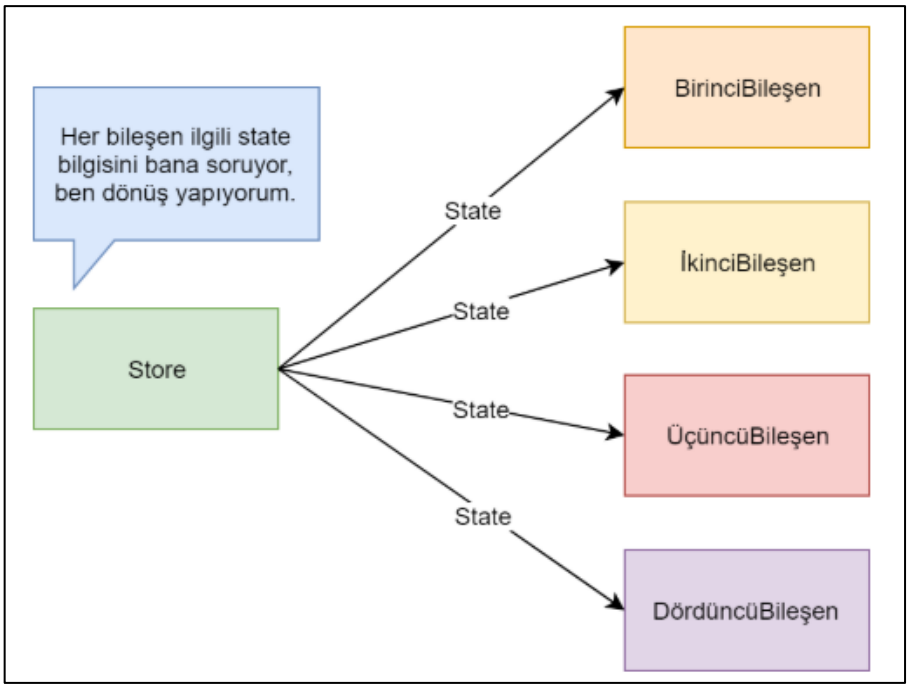


Şekil 3.5. React state bağımlılıkları

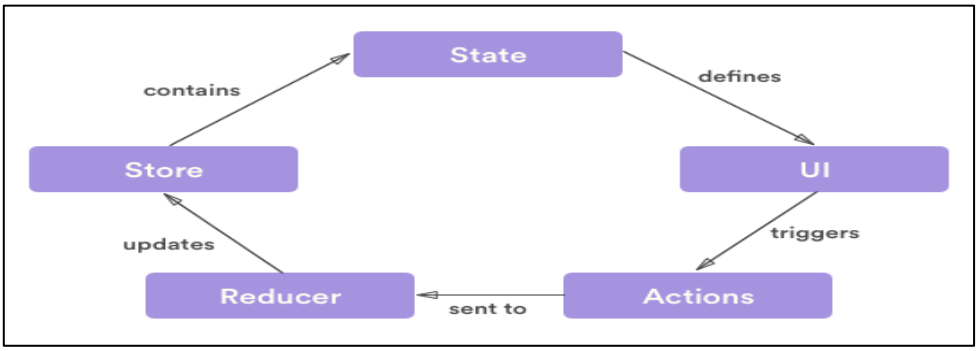
Şekil 3.5'te 4 adet bileşen bulunmaktadır. React'te tek yönlü veri iletimi söz konusu olduğundan dolayı, bu bileşenler ihtiyaç duydukları veriyi bir üst bileşenden almak zorunda. 4. bileşenin ihtiyaç duyduğu veri 2. bileşende ve 3. bileşende bulunuyor. 4. bileşen 2. bileşendeki state'e direkt olarak erişemediği için 3. bileşenden alma gereksinimi doğuyor. 3. bileşen de hiç gerekmediği halde 2. bileşenin state'ini içermek durumunda kalıyor. Bu nedenle de uygulamadaki state yönetimi güçleşiyor. React'e en büyük sorun state yönetimi bunu da Redux frameworkü ile çözebiliriz.

3.1.4. Redux

Redux JavaScript uygulamaları için state yönetimi sağlayan bir frameworktür.Redux, veri erişiminin tek bir yerden yönetilmesinin yanında, yukarıda anlattığımız state aktarımından doğan problemi de çözmek üzere tasarlanmıştır. Redux'ta bütün state verileri store adı verilen bir yerde tutulur. Böylece üst seviye bileşenden alt seviye bileşene state aktarımına gerek kalmaz. Her bileşen, uygulama state'ine store aracılığıyla erişim sağlar. Veriler, tek bir sorgulama kaynağından (store) getirildiği için, hangi state nereden gelmiş gibi bir veri takibine gerek kalmamış olur. Bu sayede, oluşan kod Redux ile daha temiz hale gelmiş olur.



Şekil 3.6. Redux state yönetimi



Şekil 3.7. Redux frameworkün genel işleyiş yapısı.

3.4.1. Redux Bileşenleri

ActionCreator: Bir state'in güncellenmesi için action tarafından tetiklenmesi gerekir. Nasıl güncellenmesi gerektiğini hangi fonksiyonların çalışması gerektiğini burada belirleriz.

Store: Statelerin değerlerinin tutulduğu bir nevi depodur. Flux'da birden fazla store bulunurken Redux'da sadece bir tane store bulunur ve her iş burada yapılır.

Reducer: Reducer'lar store'da yapılacak değişiklikleri state'i değiştirmeden state'in kopyasını alarak değişiklikleri bunun üzerinde yapar. State direk olarak işlenmez bunun yerine state'ler parçalanır ve her bir parça kopyalanır daha sonra bütün bu parçalar yeni

bir state içinde birleştirilir. Bu sayede eski ve yeni state bir arada saklanır. Bu teknoloji eski sürüme dönmeye olanak sağlar.

Provider: Store'un tüm uygulamaya etki etmesini sağlayan , uygulamanın etrafını sarmalayan bir yapıdır.

Connect: Eğer bir bileşen bir state'in güncellenmesini isterse connect()'i kullanarak kendini sarmalar.

3.5. MongoDB

Doküman tabanlı bir NoSQL veritabanıdır. Verileri yapısal olarak Json formatında ve dokümanlar halinde koleksiyonlar(collection) içerisinde tutmaktadır. Herhangi bir alana yahut aralığa göre sorgu yazılabilmektedir.

3.1.5.1. MongoDB'nin Özellikleri şu şekildedir

- Ölçeklenebilirdir (Scalable) . Veri boyutu arttığı durumlarda veya performans sıkıntısı yaşadığımız durumlarda makine ekleyebiliriz
- Veriler document (belge) biçiminde saklanır. Burada JSON verilerini kullanabiliriz
- Veriler JSON şeklinde saklandığı için gelen veri yapısı değişse bile kaydetme işleminde sıkıntı yaşanmaz.
- Verilerin birden fazla kopyası saklanabilir ve veri kaybı yaşanmaz (Replication)
- Veriler üzerinde index oluşturarak verilere hızlı bir biçimde ulaşabiliriz.

Genel kavramlar şu şekildedir;

- Database: Veritabanı
- Collection: Veritabanında kullandığımız tablo burada Collection olarak adlandırılır
- Document: Her bir kayıt document olarak isimlendirilir
- Field: Tablodaki her bir kolon field olarak adlandırılır

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

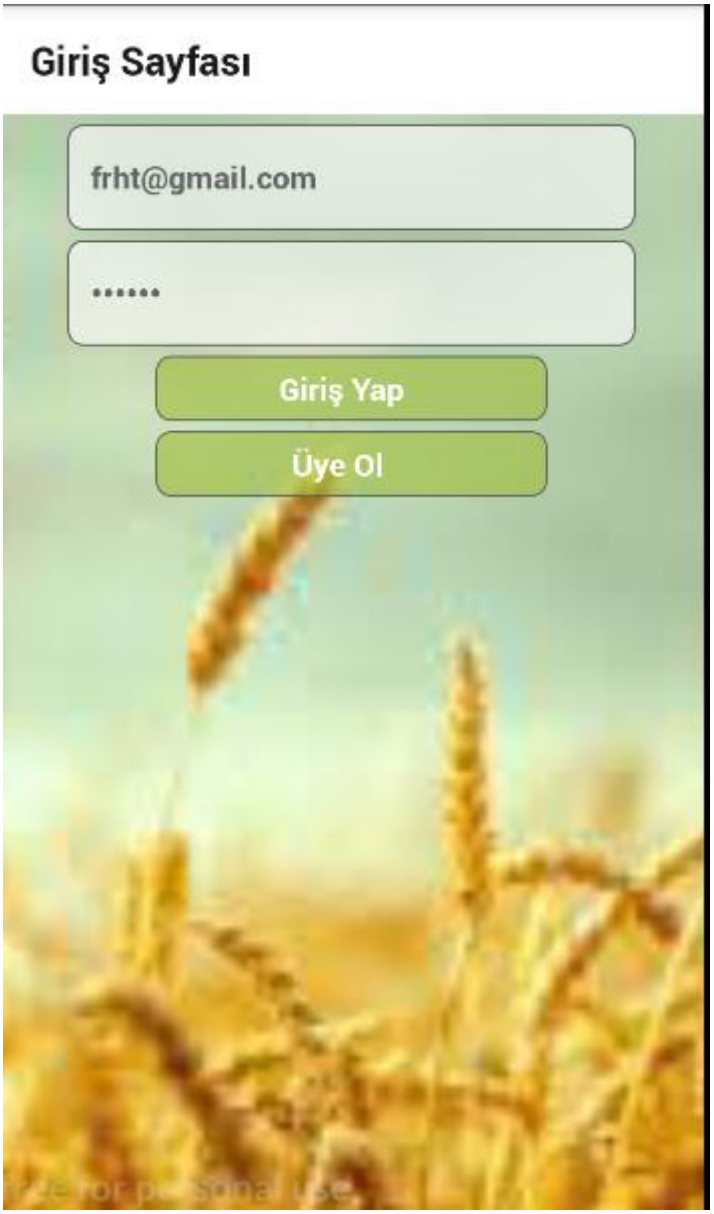
4.1. Kullanıcılar İçin Böyle Bir Uygulama Gerekli Midir?

Tarımla ilgilenen insanlar tarlalarıyla ilgili ekim zamanına ve tecrübelerinin getirdiği bilgi birikimleriyle tahminde bulunabilir fakat günümüzde sürekli gelişen teknoloji ve herkes tarafından akıllı cihazların kullanılmaya başlanmasıyla birlikte özellikle tarımla ilgilenen ve tarla sahibi olan insanlar için böyle bir mobil uygulama gereksinimi de oluşmaktadır.

Kullanıcılar sahip oldukları tüm tarlalarına görebilecek tarlalarıyla ilgili genel bilgileri anlık olarak takip edecek duruma gelebileceklerdir.

5. SONUÇLAR VE ÖNERİLER

5.1. Sonuçlar

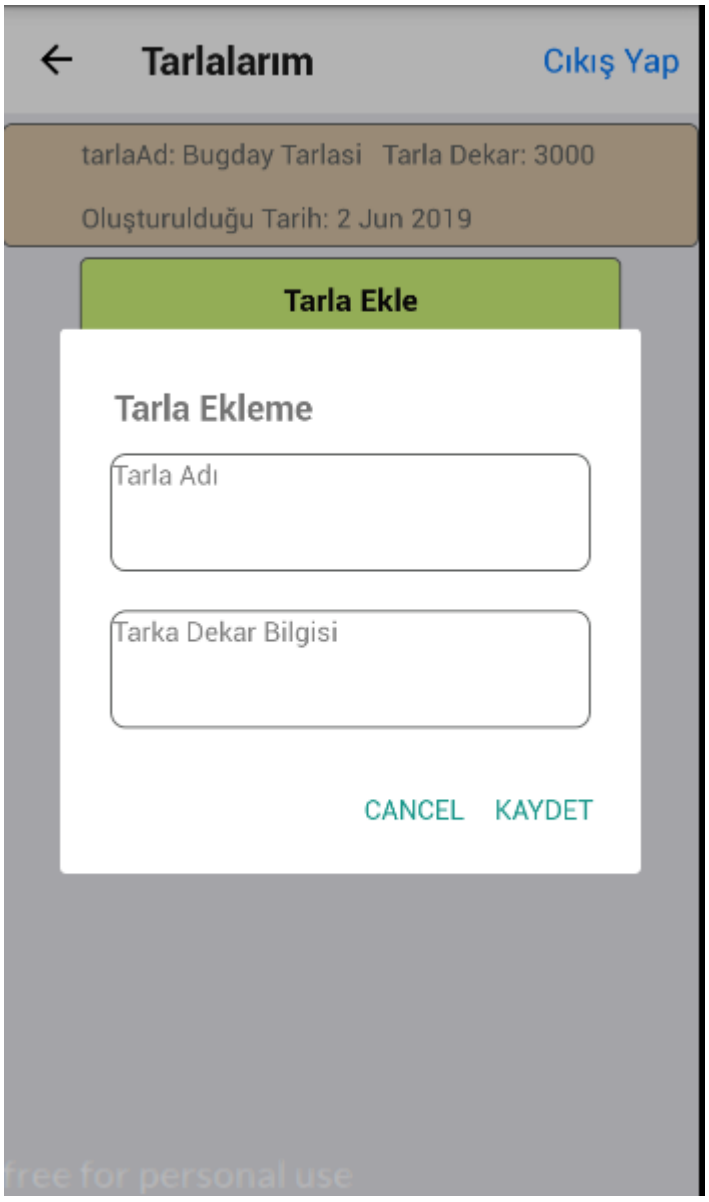


Şekil 5.1. Uygulamada kullanıcıların karşılaşacağı ilk ekran

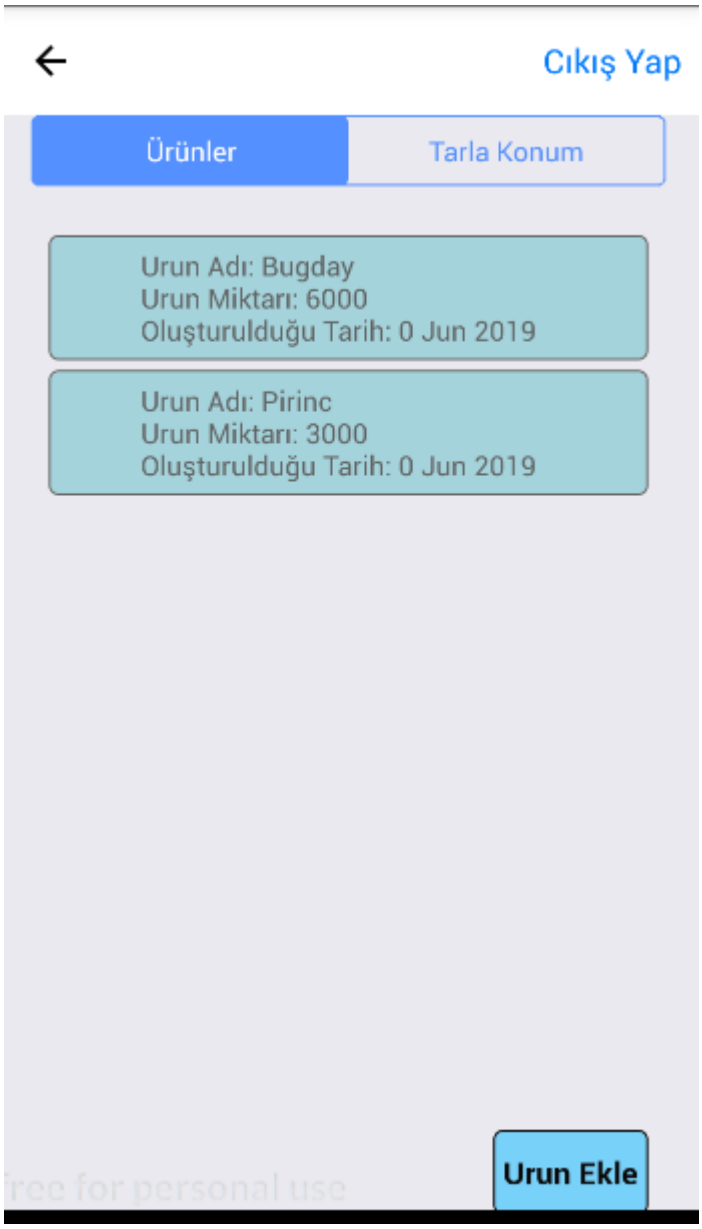
Şekil 5.2. Uygulamada üye ol sayfası.

Şekil 5.3. Uygulamada sezon ekleme ve sezonların listeleneceği sayfa.

Şekil 5.4. Uygulamada tarlaların listeleneceği sayfası.

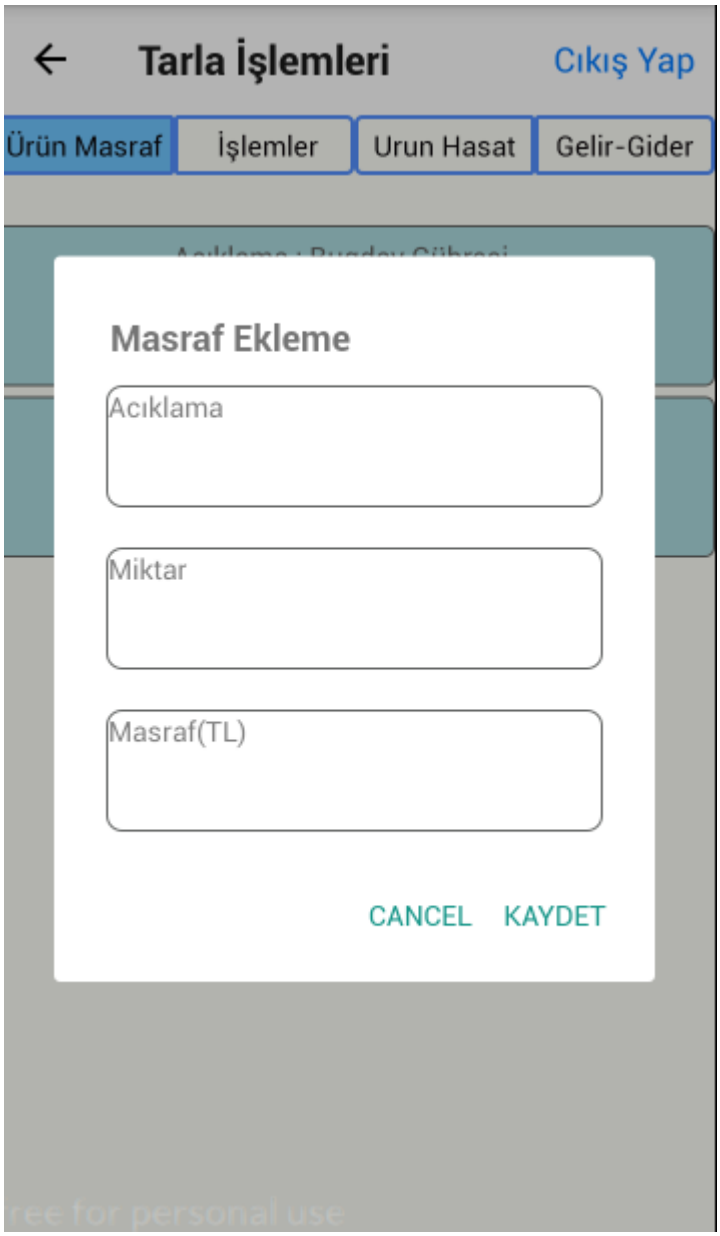


Şekil 5.5. Uygulamada tarla ekleme penceresi.



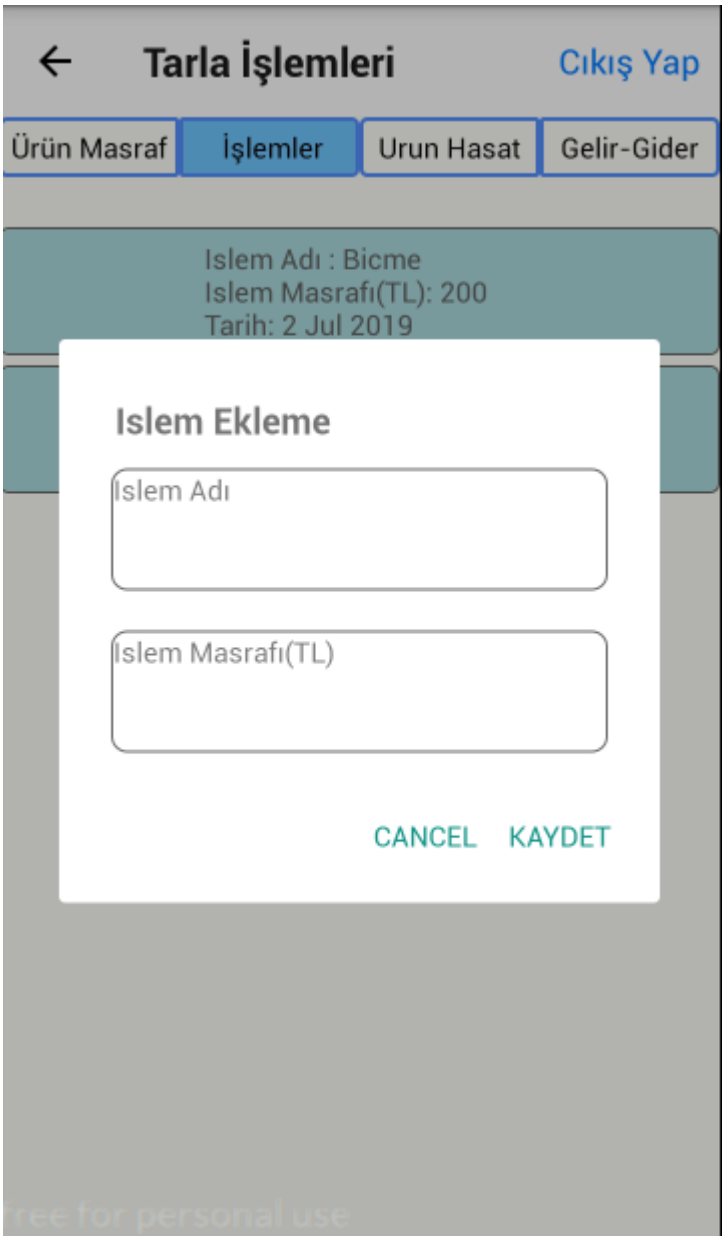
Şekil 5.6. Uygulamada tarlaya eklenecek ürünlerin listeleceği ve ürün ekleneceği sayfa.

Şekil 5.6. Uygulamada tarla üzerinde yapılacak masrafların listeleneceği sayfa.



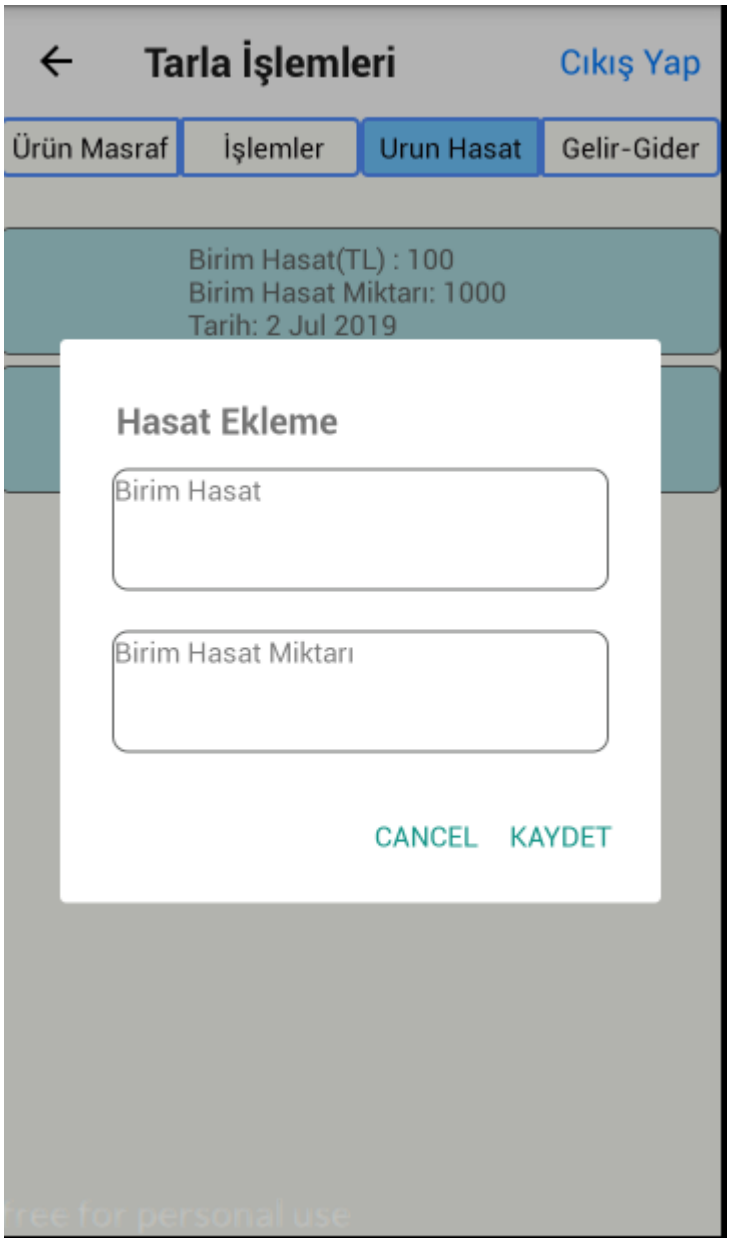
Şekil 5.7. Uygulamada tarlaya masraf ekleme penceresi.

Şekil 5.8. Uygulamada tarla üzerinde yapılacak işlemlerin listeleneceği sayfa.



Şekil 5.9. Uygulamada tarlaya işlem ekleme penceresi.

Şekil 5.10. Uygulamada tarla üzerinde yapılacak hasatların listeleneceği sayfa.



Şekil 5.11. Uygulamada tarlaya hasat ekleme penceresi.

Şekil 5.12. Uygulamada tarla üzerinde yapılan tüm işlemlerin gelir gider hesabı.

5.2. Öneriler

Mobil uygulamamızda kullanıcı tarlaları navigasyonda görülebilir olmalı, kodlamanın her adımından sonra düzenli olarak birim testi ve Play Store ve App Store'a konulmadan önce performans , uyumluluk , arayüz , fonksiyonel testleri ve bakımı yapılmalıdır.

KAYNAKLAR

[1] https://www.frmartuklu.org/konu/tarim-%C3%9Cr%C3%9Cnler%C4%B0-ve-yet%C4%B0%C5%9Eme-ko%C5%9Eullari-nelerdir.25483/
[2] https://facebook.github.io/react-native/
[3] https://marvelapp.com/project/2510749/
[4] http://www.salihk.info/react-native-nedir
[5] https://kodcu.com/2018/01/react-native-dunyasina-genel-bakis/
[6] https://gelecegiyazanlar.turkcell.com.tr/konu/web-programlama/egitim/402-mongodb/mongodbye-giris
[7] https://www.tohumgubre.com/sayfa/bugday-yetistiriciligi-ve-puf-noktalari
[8] https://nfcztyo.wordpress.com/2016/06/12/redux-nedir-nasil-kullanilir/

EKLER

Uygulama kaynak kod deposu:

https://github.com/RamazanFerhatSonmez/Cross-PlatformReactNative-Tar-m-Uygulamas-