Semih Kalaycı ToDo List App ReadMe

Kullanım:

Uygulama liste ve detay olmak üzere iki ekrandan oluşmaktadır. İlk açıldığında liste ekranından açılan uygulama önceden kayıtlı verileriniz varsa onları liste halinde size gösterir. Herhangi kayıtlı bir verinizin olmaması ya da yeni yapılacak görev eklemek istemeniz durumunda ekranın sağ üst kısmında bulunan + butonu ile ekleme yapabilirsiniz.

Liste ekranında + butonuna tıklanmasıyla detay sayfası açılır. Bu sayfa, yeni yapılacak kaydı eklemek ve daha önce eklenmiş bir kaydın detayarını göstermek amacıyla kullanılır. Eğer detay ekranına liste ekranındaki + butonu ile geldiyseniz "Delete" butonu gözükmez ve sağ üst köşede "Save" butonu ile kayıt yapabilirsiniz. Eğer liste ekranında bulunan kayıtlardan birinin üstüne tıkladıysanız detay ekranın daha önce kaydettiğiniz bilgilerle dolu olarak gelir. Aynı zamanda detay ekranının altında veriyi silmek için bir "Delete" butonu gözükürken sağ üsttei butonumuz "Update" olarak gözükür.

Detay ekranında yeni bir veri kaydederken ya da eski veriyi güncellerken başlık girmek zorunludur fakat altında bulunan çok satırlı ve daha büyük olan alana istediğiniz gibi açıklama ya da uzun notlarınızı girebilirken bu alan herhangi bir veri girişi için zorunlu tutulmamıştır(Bir koşul koyarak bu alanda zorunlu hale getirilebilir fakat sadece tek satır halinde ana ekranda görülmesi yeterli olan yapılacaklar için tercihen bu alanı zorunlu tutmadım).

Detay ekranından yeni veri ekleme, eski veriyi gücelleme ve silme gibi işlemler yapılabilirken liste ekranından da kayıtlı veriyi silme işlemi yapılabilir. Kayıtlı veriyi silmek için bulunğu satıra basılı tutup bütün satırı sola doğru kaydırmanız ya da yarıya kadar kaydırdıktan sonra basılı tutmayı bırakıp sağ tarafta çıkan "Delete" butonuna tıklamanız yeteli olacaktır. Ayrıca kayıtların sağ tarafında bulunan "Switch" butonu ile de yapılacak işlerinizi "Done" hale getirebilirsiniz. "Done" a getirilmiş olan kayıtlar yeşile döner ve tekrar olarak "Done" durumu değiştirilebilir.

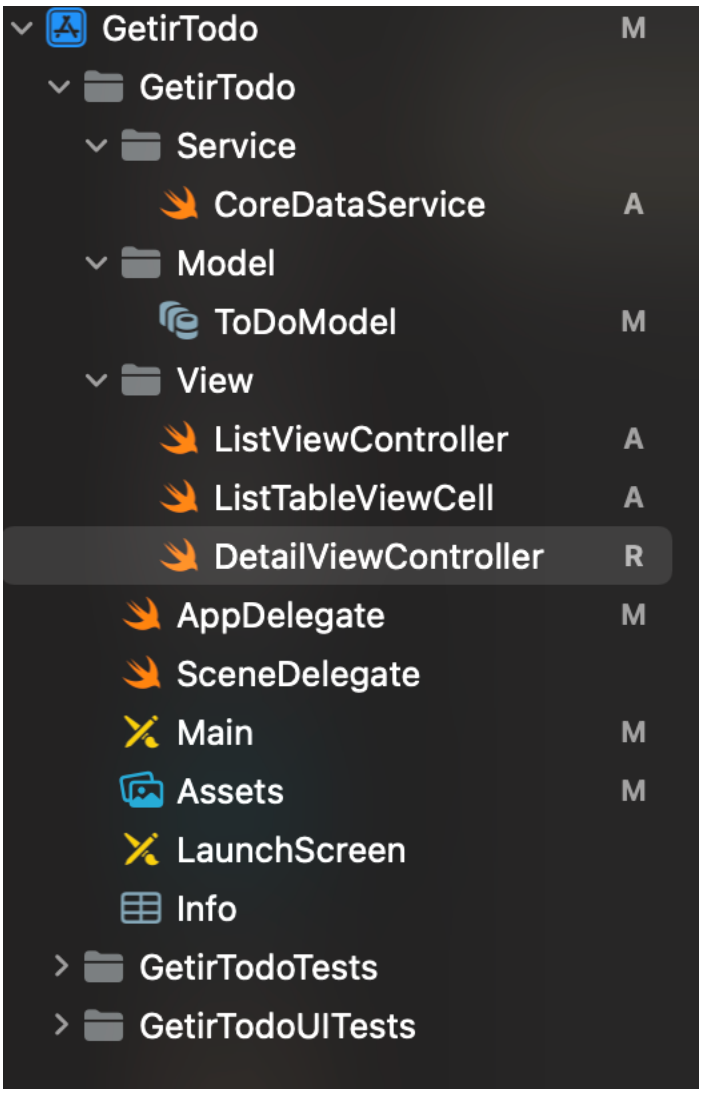
Amaçladığım:

Uygulamayı yaparken kullanışlı, temiz ve amacına uygun olmasını amaçladım. Uygulama içerisindeki kodlarımı elimden geldiği kadar kod tekrarına düşmeden, ortak func lar üzerinden ve temiz bir şekilde yazmaya çalıştım. Okunabilirlik ve kolay anlaşılabilirlik için dosya gruplandırmasına dikkat etmeye çalıştım.

Uygulama içerisinde:

- Sayfalar arası geçiş ve veri aktarımı
- NavigationBar kişiselleştirme ("+", "Save","Ubdate" butonları)
- TableView kullanımı
- CoreData kullanımı
- StoryBoard kullanımı
- TextField ve TextView(Çok satırlı metin desteklediği için) kullanımı
- Button Kullanımı
- Ortak fonksiyonların kullanımı
- Alert Kullanımı

Örnekleri yer almaktadır.



Dosya yapısı görseldeki gibidir.

Service:
İçerisinde CoreData veri işlemlerinin yapıldığı, kod tekrarını önlemek için yazılmış, bütün ekranlardan ulaşılabilen func ları barındırır.
Model:
GitHub üzerinden uygulama dosyasını çektiğimde içerisine CoreData eklemem gerekti. Bunun için oluşturulan, içerisinde table ve alanlarını eklediğim DataBase dosyasını içerir. Ayrıca CoreData Eklemek için AppDelegate üstünde de eklemeler yaptım.

View:
Ekranların ve custom eklenen TableViewCell in ViewControl dosyaları yer almaktadır

ViewDidLoad:
Ekranlar ilk yüklendiğinde yapmak istediğimiz işlemleri bu fonksiyonun içerisine ekleriz.

ViewWillAppear:
Bu fonksiyon da ViewDidload ile benzerdir fakat VewDidLoad sadece bir kere çağırılabilirken ViewWillAppear defalarcaçağırılabilir. Benim projemde kullanma amacım Detay ekranından Liste ekranına ekleme, silme ya da güncelleme işlemlerinden biri yapıldıktan sonra dönüldüğünde yapılacaklar listesi elemanlarının ve tableList in güncellenmesini salamaktır.

```swift
func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    return titleArray.count
}
func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    let cell = ListTableView.dequeueReusableCell(withIdentifier: "TableCell", for: indexPath) as! ListTableViewCell

    cell.TitleLabelinCell.text = titleArray[indexPath.row]
    cell.DoneSwitch.isOn = doneArray[indexPath.row]
    cell.idLabel.text = idArray[indexPath.row].uuidString
    cell.TitleLabelinCell.textColor = doneArray[indexPath.row] == true ? UIColor.systemGreen : UIColor.black

    return cell
}

func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
    choosenTitle = titleArray[indexPath.row]
    choosenId = idArray[indexPath.row]
    choosenIndexPath = indexPath.row
    performSegue(withIdentifier: "toDetail", sender: nil)
}
```

1. Fonksiyon:
   Listemizde kaç satır olacağını bu fonksiyonda belirtiriz. Genelde bu kuşumda return olarak ekrana basmak istediğimiz listenin uzunluğu verilir.
2. Fonksiyon:
   Oluşturduğumuz customCell i tanımlar ve içerisinde bulunan objelere hangi değerlerin atanacağını belirtiriz.
3. Fonksiyon:
   Table içerisinde bir satıra tıklanma işlemi yapıldığında çalışır. Gerekli atamalar yapılmış ve detay sayfasına gönderilmek için performSegue başlatılmıştır

```swift
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "toDetail" && sender == nil {
        let destinationVC = segue.destination as! DetailViewController
        destinationVC.selectedId = choosenId
        destinationVC.selectedTitle = choosenTitle
        destinationVC.selectedIndexPathRow = choosenIndexPath
    }
}
```

Sayfalar arası veri aktarımı ve geçiş:

Bu fonksiyon sayesinde sayfalar arası geçiş ve veri aktarımı sağlanır. 3. Fonksiyonda daha önce değişkenlere atadığımız seçilmiş değerler bu fonksiyonda gideceğimiz ekran içerisindeki değerlere eşlenir ve daha sonra da navigasyon işlemi gerçekleşerek sayfa değiştirme ve veri aktarımı gerçekleşmiş olur.

StoryBoard Öğeleri genellikle Outlet ve Action özelliklerini barındırırlar. Outlet özellikleri o objenin title, backgroundcolur vs gibi nesnesel özelliklerine ulaşmamızı sağlar. Action ise changeValue, onClick vs gibi olaylara vermesini istediğimiz tepkileri yazdığımız fonksiyonlardır.

Uygulamamın içerisinde bulunan, temel olarak farklı projelerde de karşımıza çıkabilecek öğelerin kullanım şekillleri ve kullanım amaçları da yukarıda belirttiğim gibidir. Bunlar haricinde notlar ve bilgilendirmeler kod içerisinde yorum satırları halinde yer almaktadır.