ECS189E Homework 4

Overview

In this assignment, we'll further simplifying the login process, manage accounts, and make transactions.

Directly Show Wallet

Normally if an auth token can be found, the whole login and verification process should be skipped for the user. As a result, check for the auth token even before loading the first view, and show the right view according to the result. Below is some sample code for a hint on how to do this, remember that for the majority of the app we use a UINavigationController which is *technically* a UIViewController

Hint: SceneDelegate

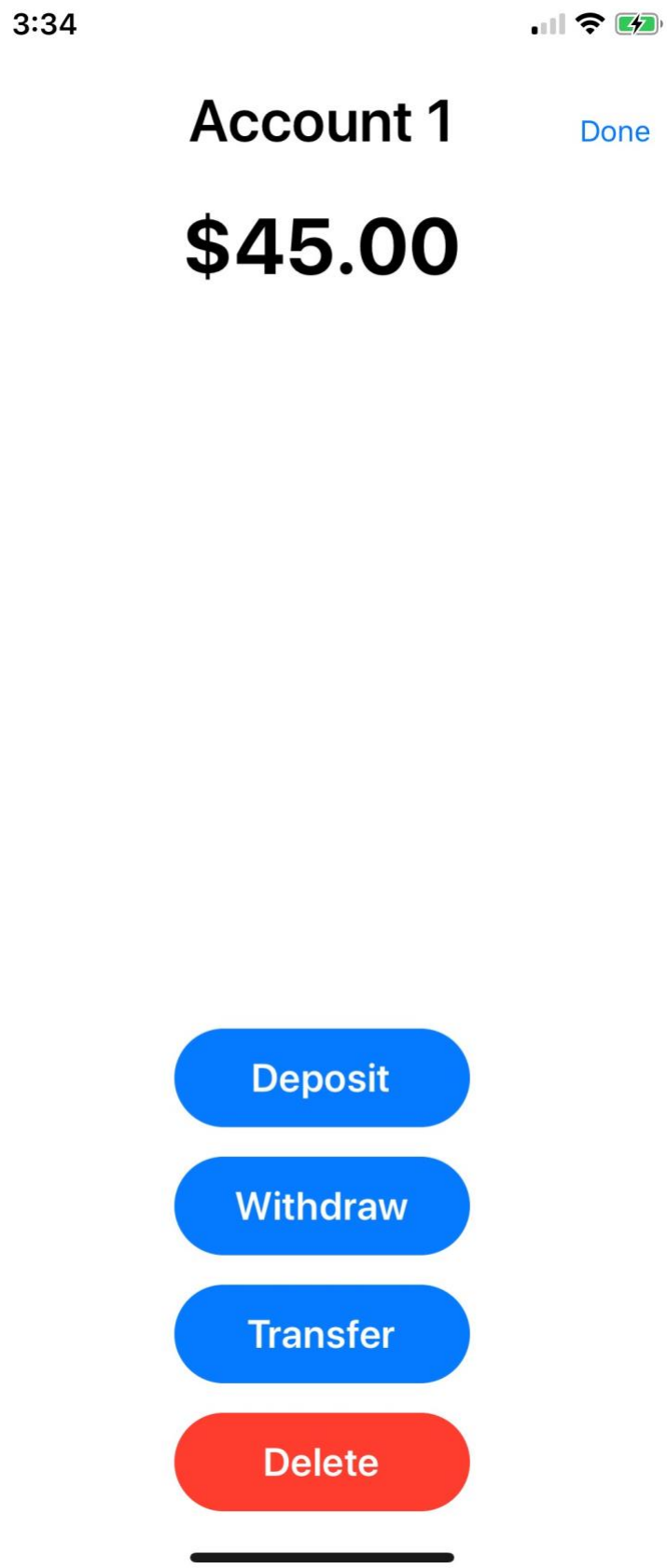
```
guard let windowScene = (scene as? UIWindowScene) else { return }

if Storage.authToken != nil {
    let storyboard = ...
    let navigationController = ...
    let viewController = ...

    // some code to set up the navigationController starting at viewController
    self.window?.windowScene = windowScene
    self.window?.rootViewController = navigationController
    self.window?.makeKeyAndVisible()
```

Account View

When user taps on a row in your wallet table, **your app is required to bring the user to an Account View that will present at least**:

1. A label showing account name
2. A label showing amount
3. Four acount-related buttons:
    - Deposit
    - Withdraw
    - Transfer
    - Delete
4. A Done button that will bring you back to HomeView

Here is a screenshot of mine app (the background is white, so you can't really see the border):


3:34
Account 1
Done
$45.00
Deposit
Withdraw
Transfer
Delete

Save the Accounts data to server

In the `init` of class `Wallet` in classes.swift file, if you pass `false` to `ifGenerateAccounts` , it will try to parse the account data from the `response` , and will give you an empty variable `[Account]` if there is no accounts data saved on server.

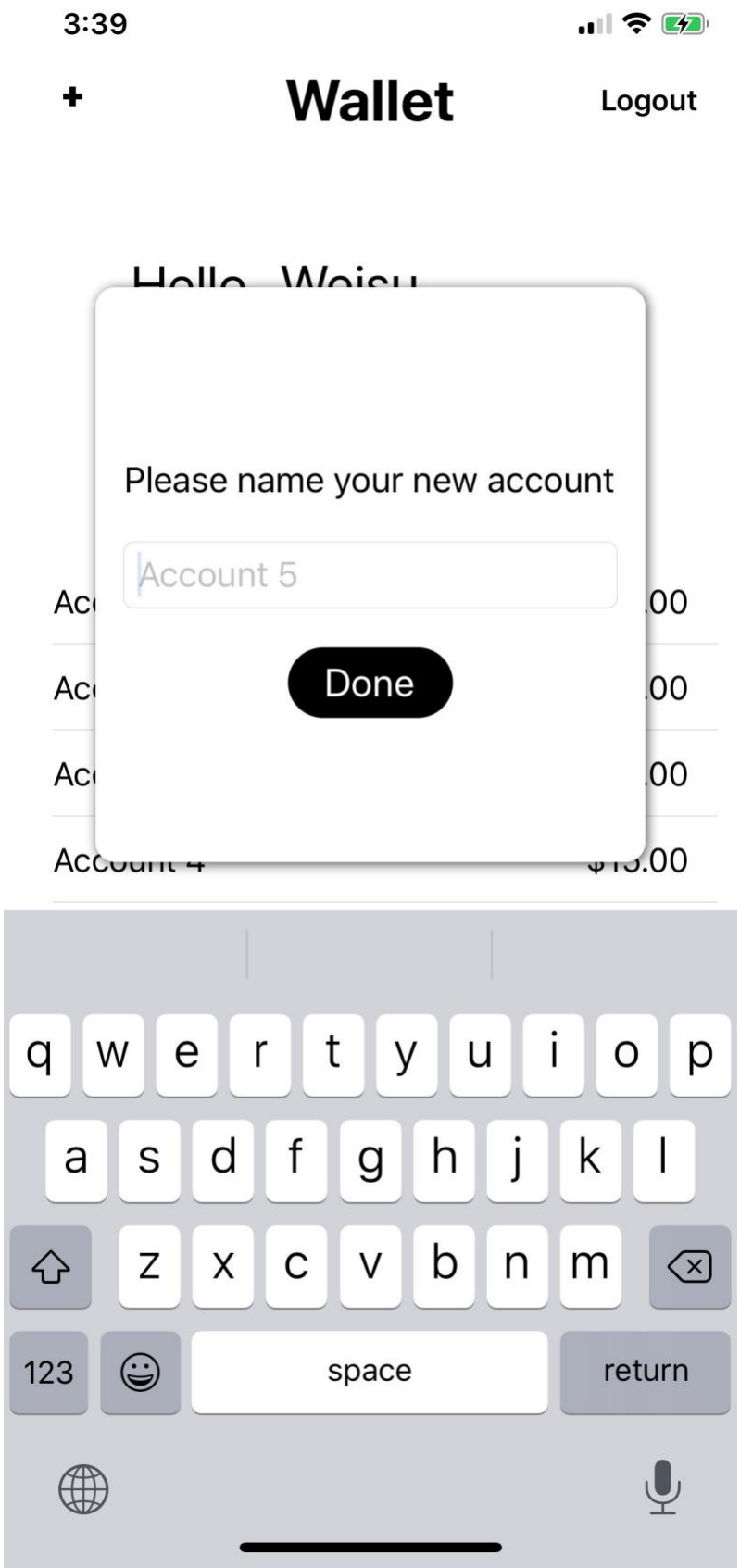
Use following Apis to create account, delete account, and make transactions:

```
static func deposit(wallet: Wallet, toAccountAt accountIndex: Int, amount: Double, completion: @escaping ApiCompletion)

static func withdraw(wallet: Wallet, fromAccountAt accountIndex: Int, amount: Double, completion: @escaping ApiCompletion)

static func transfer(wallet: Wallet, fromAccountAt fromIndex: Int, toAccountAt toIndex: Int, amount: Double, completion: @escaping ApiCompletion)

static func addNewAccount(wallet: Wallet, newAccountName name: String, completion: @escaping ApiCompletion)

static func removeAccount(wallet: Wallet, removeAccountat index: Int, completion: @escaping ApiCompletion)
```

You'll also notice that there are no longer any test functions in the Api. At this point they will cause more confusion than help so we didn't add them to the file, however you should very well be at the ability with Swift to create your own for testing purposes if you would like.

Allow Account Modifications

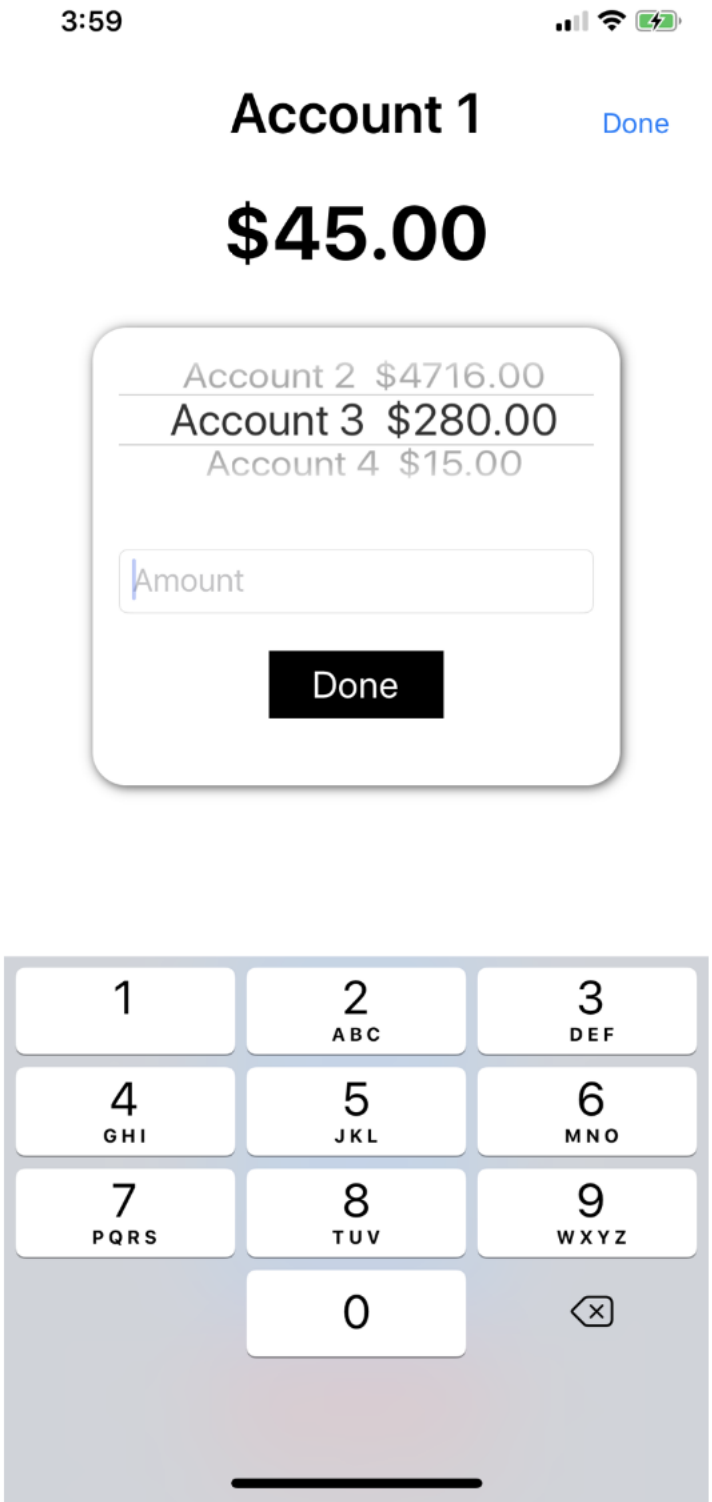
1. Allow user to delete an account.
    - Once the user deleted this account, dismiss Account View
    - **Your total amount in Home View should be updated accordingly.**
2. Allow user to create a new account (with amount 0.0).
    - Have a button in the Home View to allow user to create new accounts.
    - If user left text field blank, use "Account n+1" where n is the number of user's current accounts
      Write a custom popup instead of using UIAlertController
    - Hint: Hide the custom popup and show it when needed


3:39
+
Wallet
Logout
Please name your new account
Account 5
Done
Ac
.00
Ac
.00
Ac
.00
Account 4
$15.00
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
123
space
return

Allow Transactions

1. Allow user to deposit to an account.
    - You can use UIAlertController for this. Prompt user what's going on and provide a text field. **The text field should give number pad**.
2. Allow user to withdraw from an account.
    - **If the user entered an amount that's greater than current account's balance, use the balance as the amount**. For example, if the user's account only has 300 and the user tries to deposit 400, the app should only deduct 300.
    - You can use UIAlertController for this. Prompt user what's going on and provide a text field. **The text field should give number pad**.
3. Allow user to make transactions between two accounts.
    - **Write a custom popup containing:**
        - A UIPicker showing **all other accounts**
        - A text field taking transfer amount
        - A Done button
    - **Disable other UI to avoid bugs**
    - **Similarly, account that sends out the money shouldn't go under 0**

*NOTE: To clarify, we want you to practice creating a custom popup in this Homework. You **must** do this for both the creation of new accounts and transfer between accounts, however in withdraw and deposit you may use UIAlertController if you would like.*


3:59
Account 1
Done
$45.00
Account 2 $4716.00
Account 3 $280.00
Account 4 $15.00
Amount
Done
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
0

To be more specific, when it comes to the case of deposit and withdraw, the money is coming from "nowhere" and going to "nowhere". Like the ATM, take them as cash in and cash out. However, you need to limit the amount of money for withdraw and transfer. The account amount should not go under 0.

You are not allowed to directly change account amount on the client side. **It is required to use the functions provided in Api.swift and make the changes on server side.**

IMPORTANT: the amount in each account and Home View should always be in sync. Once the api call succeeds, reflect the amount change on your UI right away. You don't need to handle api call failures for this assignment.

Grading

1. Directly Show Wallet (5')
2. Allow Account Modifications (15')
3. Allow Transactions (15')
4. UI Functionality (15')

*Note: At this point it should be well known that the use of **any** force unwraps besides the few Professor King has okayed will be an automatic 0 on the Homework or any assignment, no questions asked.*

Submission

Push your files to the repository and **submit on Canvas**.