ECS189E Homework 4

Overview

In this assignment, we'll let the users to manage accounts and make transactions.

Update API

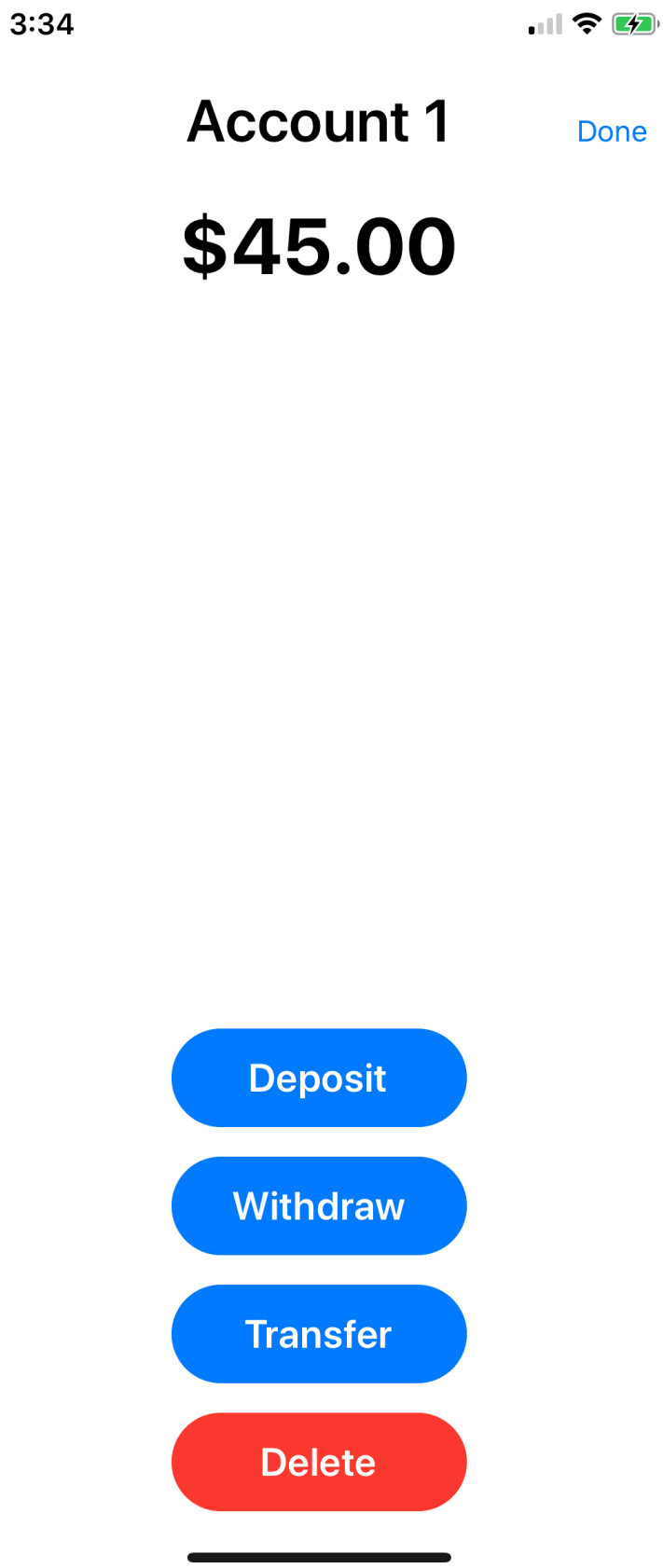
Update your API file like what you did in previous assignment.

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):



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 `[Account]` if there is no accounts data saved on server.

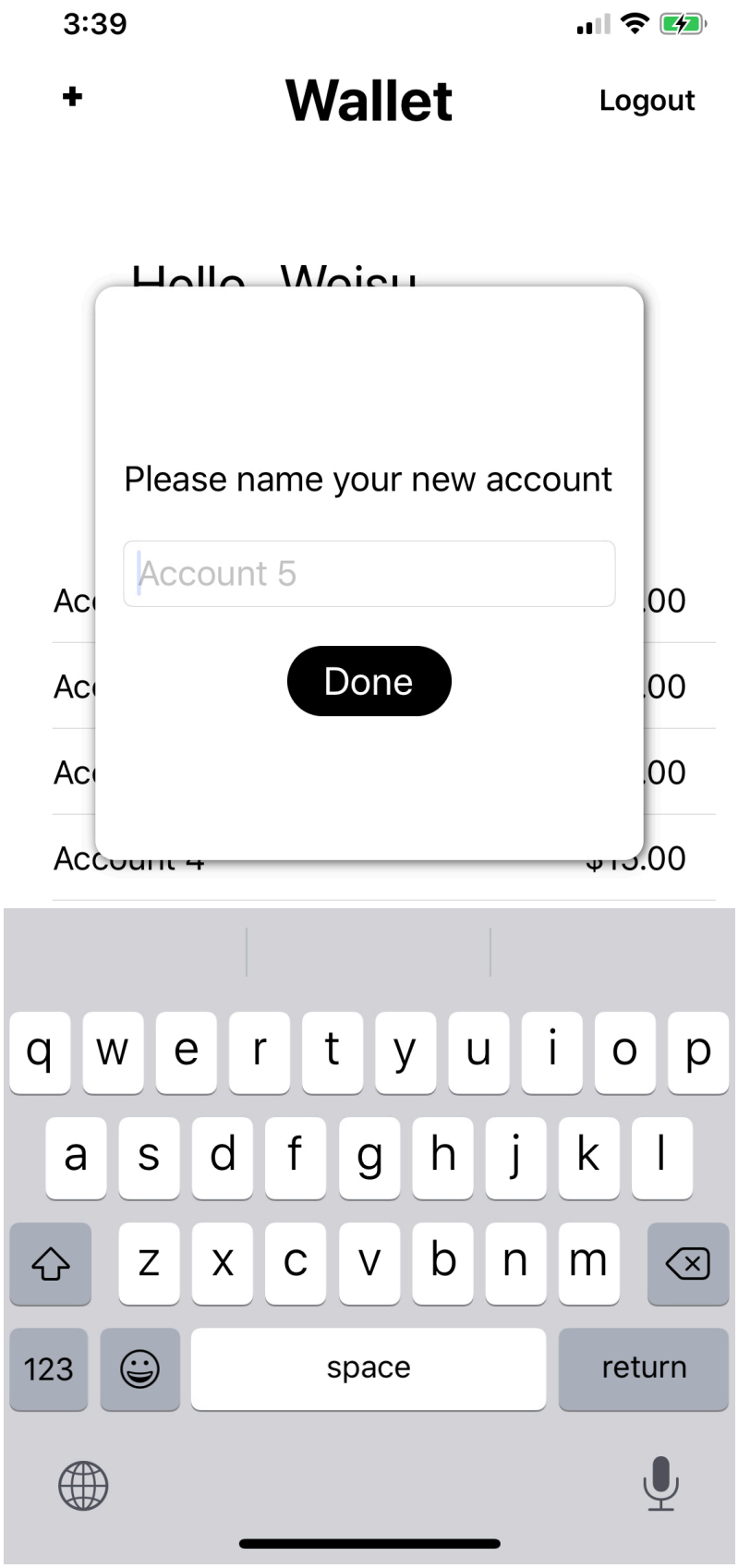
Use following Apis to create account, delete account, and make transactions:

```
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)
```

Allow Account Modifications

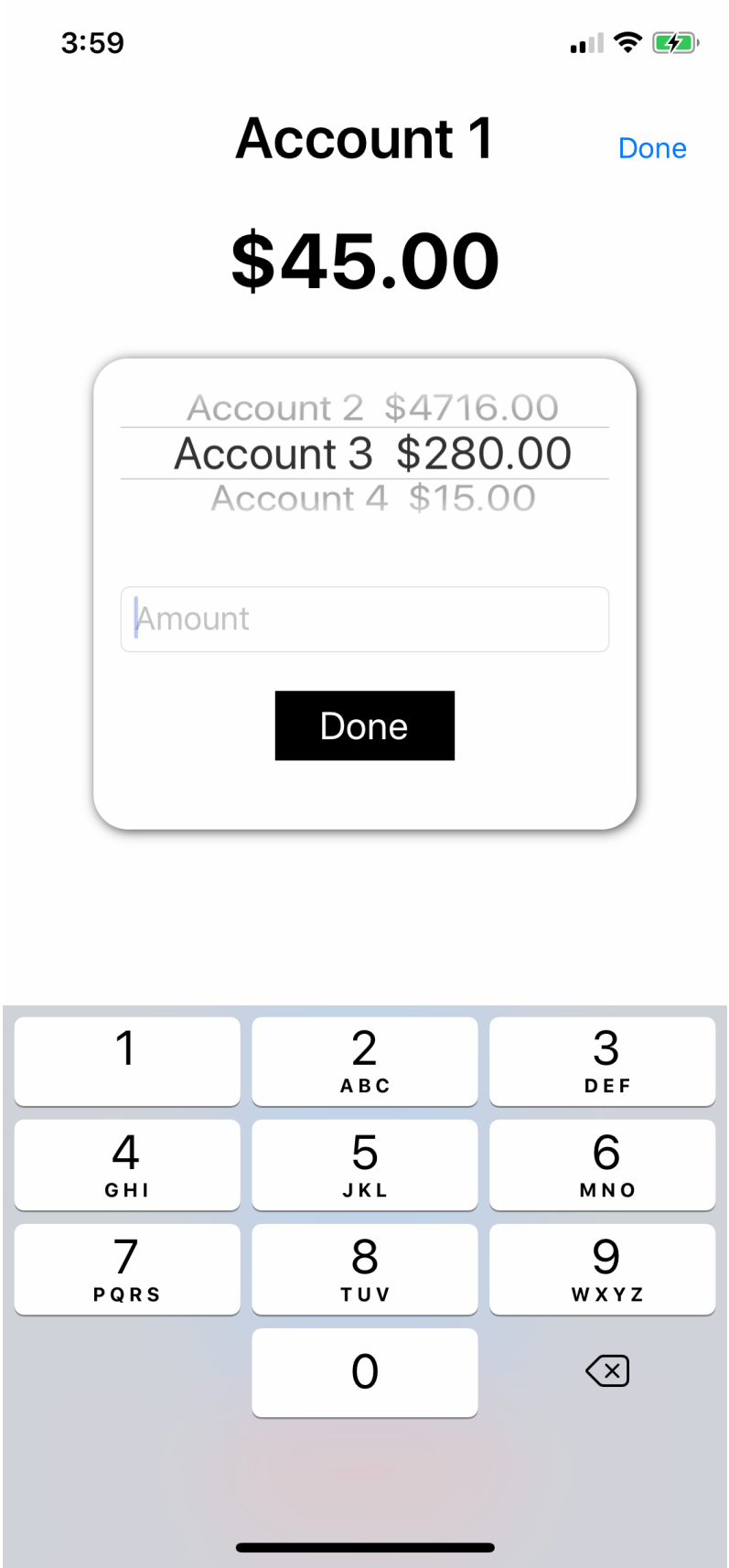
1. Allow user to delete an account.
    - Once the user deleted this account, dismiss Account View
    - **Your total amount in Home View should be updated accordingly.**
2. Allow user to create a new account (with amount 0.0).
    - Have a button in the Home View to allow user to create new accounts.
    - If user left text field blank, use "Account n+1" where n is the number of user's current accounts. Make sure the account name is unique though. Fix my bug while presentation :).
    - **Write a custom popup instead of using UIAlertController**

    Hint: Hide the custom popup and show it when needed



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**


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.

CardScan (Optional)

While withdraw, use the CardScan library to scan a credit card. After that, your popup(UIAlertAction) should prompt user he/she is depositing using "card number xxxxx"

Using CardScan requires a real iOS device, so we made it optional. However, we do encourage your to implement this as it's a good practice of using delegate.

You can find CardScan here. **You don't need to worry about the Api key.**

Grading

1. UI Functionality (15')
2. Allow Account Modifications (15')
3. Allow Transactions (15')
4. Style and Documentation (5')

Submission

Push your files to the repository.