ECS189E Homework 1

Overview

The app you are going to build for the homework is a wallet, which can be broken down into **three** parts: login, wallet, and transaction.

For the login part, your App will use the phone number authentication. To be more specific, your App will take a phone number from the user and send a verification code to this phone number by text message. Your App will then ask the user to enter the code and verify user's identity. You can assume that the user will use US phone number only. (Homework 1 and 2)

For the wallet part, your App will display the wallet, which has a table of accounts and the information of each account. For example, it might contain the name of the account, the available balance of each account and so on. (Homework 3)

For the transaction part, your App will allow the user to make transactions between the accounts in the wallet. It will also use card scan to "deposit" into your accounts(Homework 4)

Expectation

When building an App, there is not a single "correct" solution. As one of the purposes of this course, we do want you to be creative and put your unique thoughts and designs into your App. The homework instructions will list out the basic functions and objects that you have to include in this App. However, there are lots of things that you can build on top of these requirements.

Create and Setup the Project

If you have not already done so, clone this repository to your local device. Create a new "Single View App" project using Xcode under the directory you just cloned. This project should be for iPhone only and for Portrait only. **Change the User Interface option from SwiftUI to Storyboard.** SwiftUI is a new way of building UI introduced by Apple recently. We are not going to cover it in this class, but feel free to learn more about it.

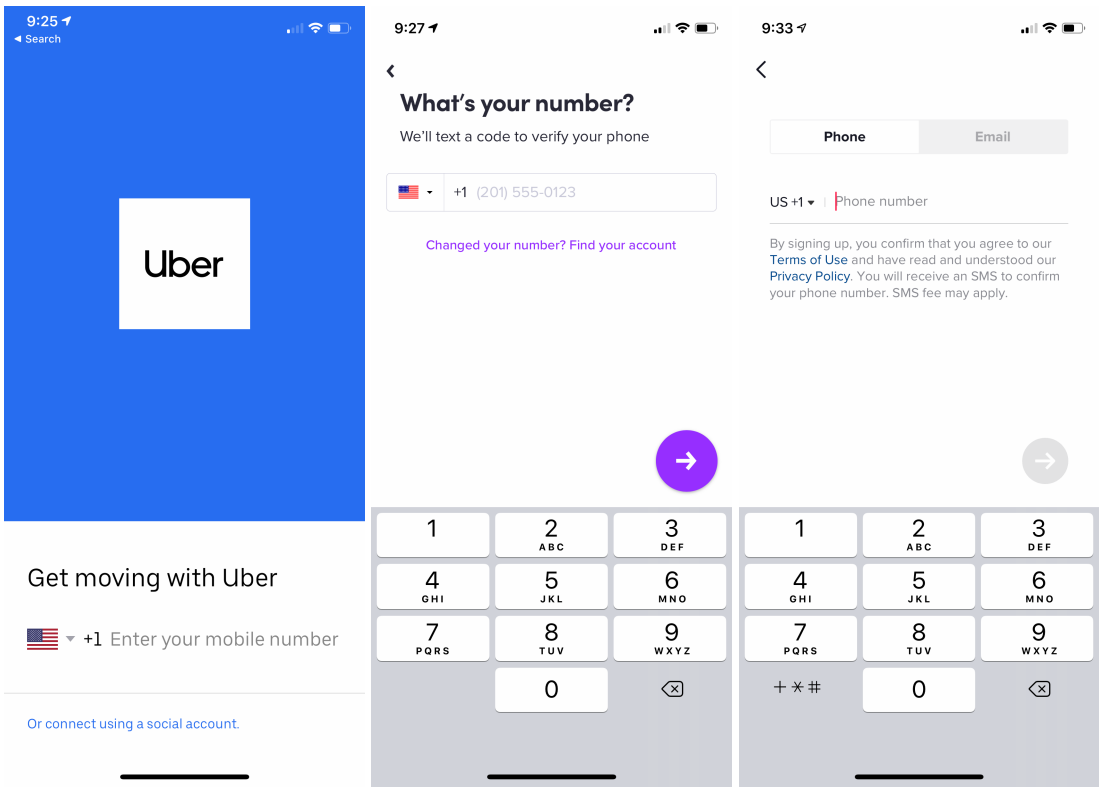
Following that, you need to create the workspace that you will be working on for all the homework. Follow this link to install the Pod for your project. The only Pod that is required to use for this homework is called "PhoneNumberKit". All the other guides on this web-page can be very helpful and might be used in the future homework, so make sure you go through them. After successfully installed the Pod (or Pods), close your created project and **open it with .xcworkspace file from now on.**

Launch Screen

Use the "LaunchScreen.storyboard" to design your launch screen. You should at least include a image on your launch screen. This link can be a great source for you to pick icons. Always remember to check the copyright when you use any resource.

Login View

There are a lot of Apps use the phone number authentication. Here are some examples:



UI Design

On the first view of the login part, you should at least have:

1. An instruction (or instructions) on the screen to tell users what to do
2. A text field showing the country code, i.e., +1 for US. You can assume the user will use US phone number only. User shouldn't be able to interact with the country code field for now.
3. A text field for users to enter the phone number
4. A button to send the code
5. A label to display errors and information ONLY when needed.

Besides these requirements, feel free to add anything or any function that you consider necessary.

Important: Use Auto Layout to make sure your app looks the same on different devices.

Display the Input in Phone Number Format

Instead of a list of digits, phone numbers are usually displayed in a certain format. Adjusting the format of the input while user is entering can be very helpful for the user. For the case of this homework, you can use the "PhoneNumberKit" Pod that was installed. Read the instructions provided here to see how to use this Pod for your App.

Hint: Check out the AsYouTypeFormatter

Dismiss keyboard

Your app's keyboard should also be dismissed when the user tapped outside the keyboard or tapped the send button. To detect gestures, use **UIGestureRecognizer**. You can learn more from here.

Hint: Create a gesture recognizer object and add it to your view

Error Detection and Display

When user tap the button, check if the input is valid phone number and display the error or information. Make sure you have at least two types of errors and an information for valid input. When the input from the user is valid, the label should display the data that will be forward to the next step. In this homework, you need to store the phone number in E164 format. For example, if user entered 0123456789, it should be displayed in the text field as (012)-345-6789 (UITextField), but should be stored and displayed as +10123456789 in the information (UILabel). This string is the data that you will use in the next homework. You should also be able to get the phone number in E164 format with the "PhoneNumberKit" Pod easily.

Coding Style

It is always very important to make sure that your code is easy to read and understand. The following points will be considered when your coding style is graded:

1. Reasonable names for ViewControllers, views and objects.
2. Reasonable names for files, functions and variables.
3. Necessary comments.
4. Reasonable order of the functions.
5. The use of optional (Question mark and exclamation mark).

Documentation

Write a README.md to briefly state what functions does the App have and how did you implement them. Imagine yourself working in a big company, and this document is for the people who will take over your work, or will become your partner and work on this App with you.

Important: Remember to include your name and student ID in the document.

Grading

1. Launch Screen (5')

2. UI Design and Auto layout (15')
3. Correctly use the Pod (the format of phone number while typing) (10')
4. Correctly dismiss the keyboard (5')
5. Error detection and display (10')
6. Coding style and documentation (5')

Submission

Push your files to the repository.