Forever Shopping App

Project Documentation

Ralph

March 2018

Project Information

Project title: *Forever Shopping App*

Overview

Online shopping is a new type of electronic commerce allowing consumers to buy goods or services from a seller over the Internet using a web browser. An online shop evokes the physical analogy of buying products or services at a regular "bricks-and-mortar" retailer or shopping center.

Nowadays, tons of shopping apps can be found in the mobile application store, since people are growing accustomed to the new style of shopping, given that 30% of all online shopping purchases now happen on mobile phones.

Similar to other online shopping apps, the *Forever Shopping APP* is built up for customers to shop clothes online, with basic features of browsing item according to different category, managing their shopping carts and creating user account for checking out.

Features

Features of this app include:

1. "***Let's go window shopping!***": select the category of your interest, explore, browse the item list to find those items you like, similar to the window-shopping;

2. "***Tell me more about it!***": if you are interested in one item in the list and want to know the details, click on the item and find more information in the detail page;

3. "***Manage your cart!***": click tab at bottom to check your shopping cart at any time, it contains all items you just added, remove items you added in by mistake or no long want;

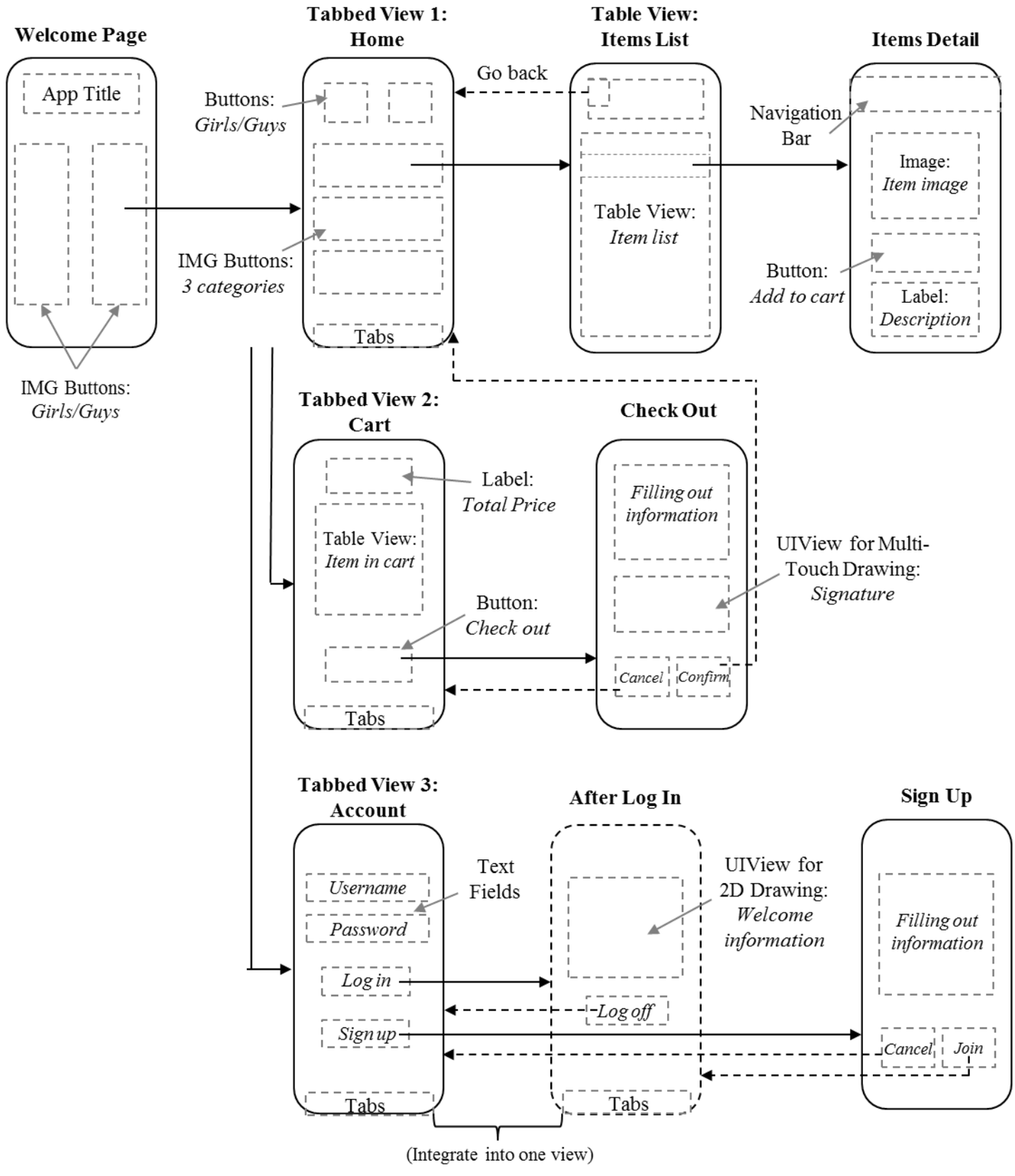
4. "***Checking out!***": page pumped up tells you how many items in your cart and their total prices, fill in your information include recipient name, mailing address and the card number for checking out, confirm your order and get system messages.

5. "***Join the club!***": click tab at bottom to see the account page, log in as a member or a guest, with a welcome messages shown up, or, create your own account with your name, email address and password.

App architecture

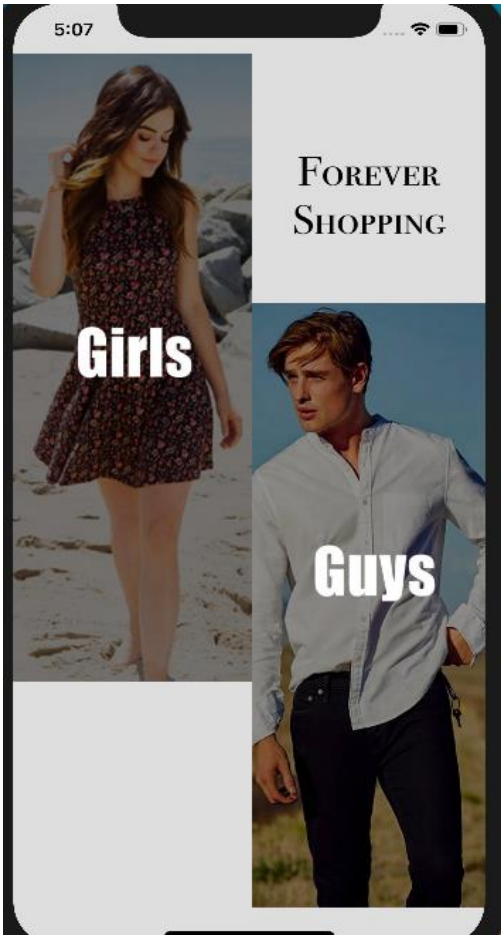
There are total 9 scenes designed for this APP (same scene lay-out may show different information).

Architecture Design Chart of this application is shown below.


Welcome Page
App Title
IMG Buttons:
Girls/Guys
Tabbed View 1:
Home
Buttons:
Girls/Guys
IMG Buttons:
3 categories
Tabs
Go back
Table View:
Items List
Table View:
Item list
Items Detail
Navigation
Bar
Image:
Item image
Button:
Add to cart
Label:
Description
Tabbed View 2:
Cart
Table View:
Item in cart
Label:
Total Price
Button:
Check out
Tabs
Check Out
Filling out
information
UIView for Multi-
Touch Drawing:
Signature
Cancel
Confirm
Tabbed View 3:
Account
Username
Password
Log in
Sign up
Tabs
Text
Fields
After Log In
Log off
Tabs
UIView for
2D Drawing:
Welcome
information
Sign Up
Filling out
information
Cancel
Join
(Integrate into one view)

Screenshot Illustrations

1 - Welcome page:

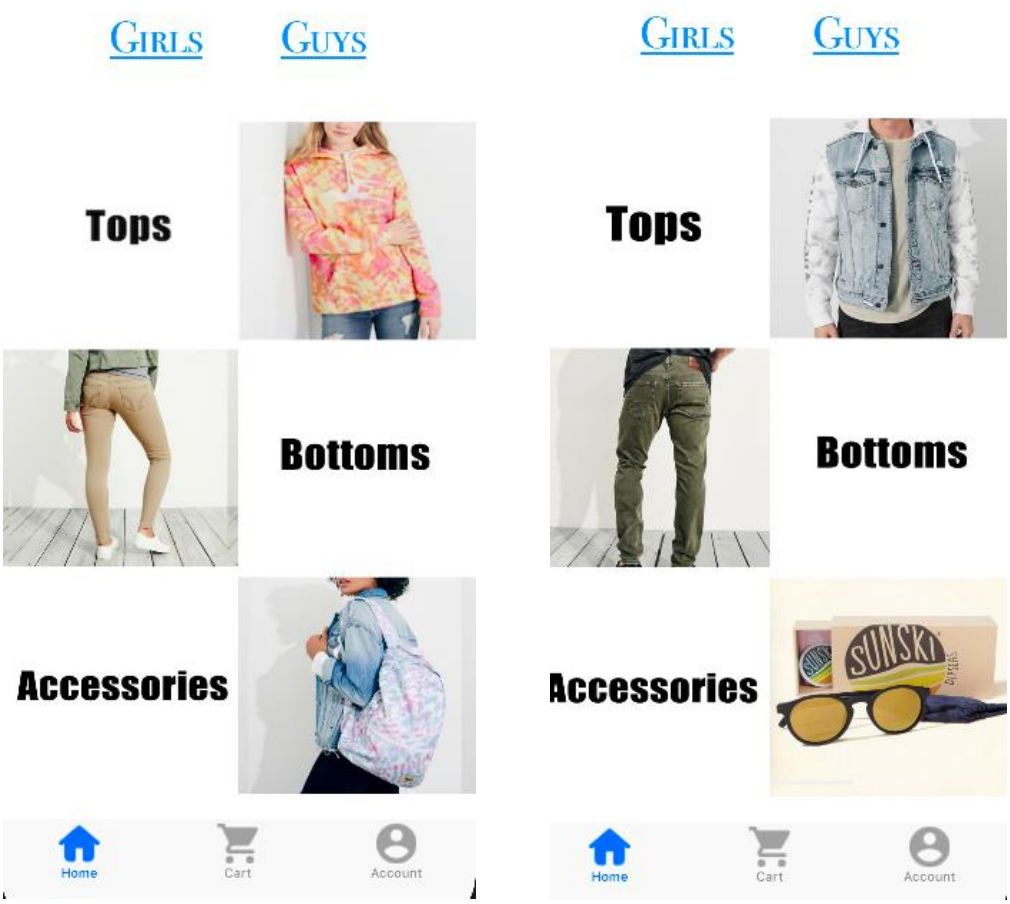


There are two Image Buttons on the screen, *Girls/ Guys*.

Click on *Girls* button, it will switch to **Home** page with girls' category.

Click on *Guys* button, it will switch to **Home** page with guys' category.

2 - Home page:



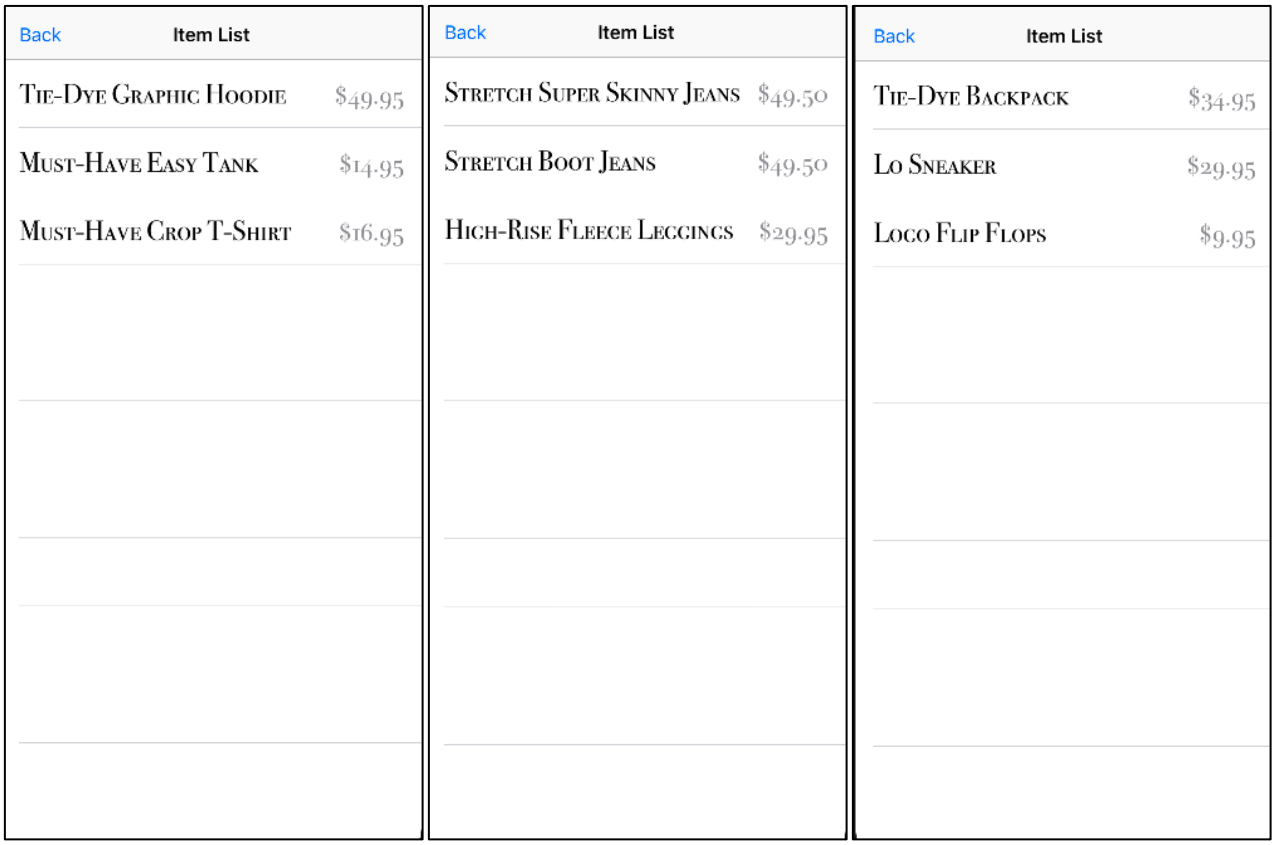
The shown up content on the page is depending on selection in **Welcome** page.

On this screen, click on *Girls* button, it will switch to girls' category; Click on *Guys* button, it will switch to guys' category.

There are three Image Buttons below. Click on *Tops/ Bottoms/ Accessories* button, it will switch to **Item List** page.

On the bottom there are three tabs for page switching.

3 – Item list page (e.g., Girl's category):



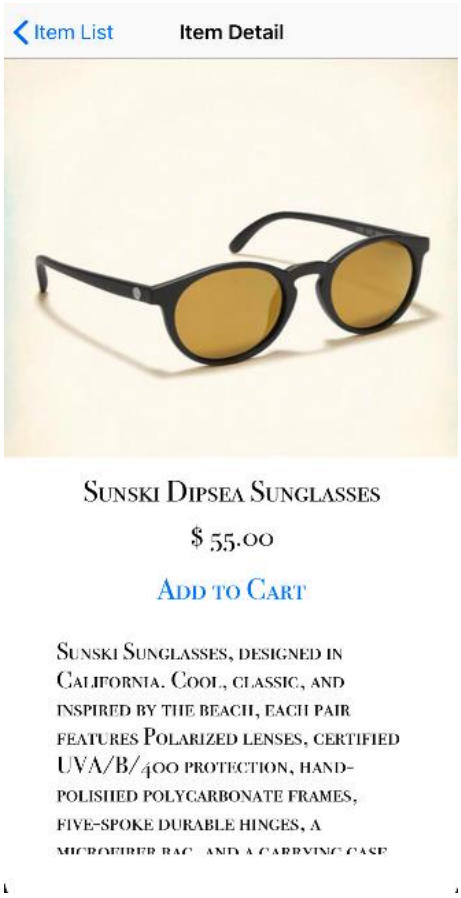
By clicking *Tops/ Bottoms/ Accessories* buttons on **Home** page, one of above page shows up.

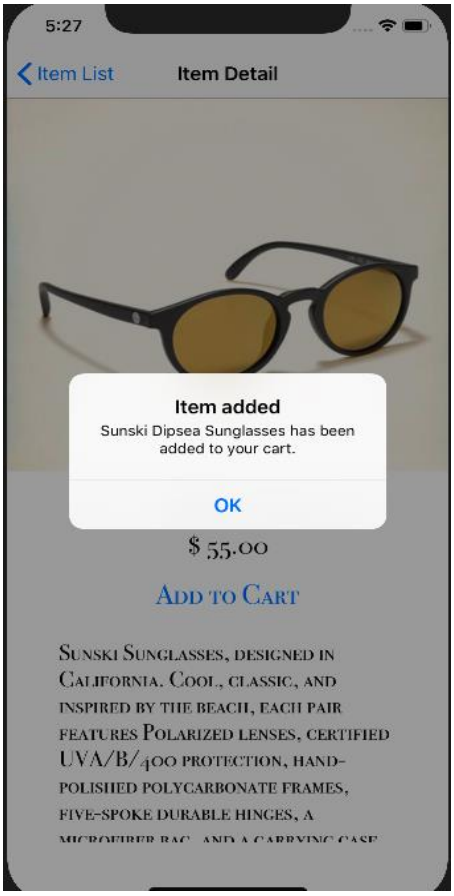
It provides a list of items in system, according to user selection of category.

On this screen, click on *Back* button, it will switch back to **Home** page.

Click on either *item*, **Item Detail** page will show up.

4 – Item detail page (e.g., Sunski Dipsea Sunglasses):





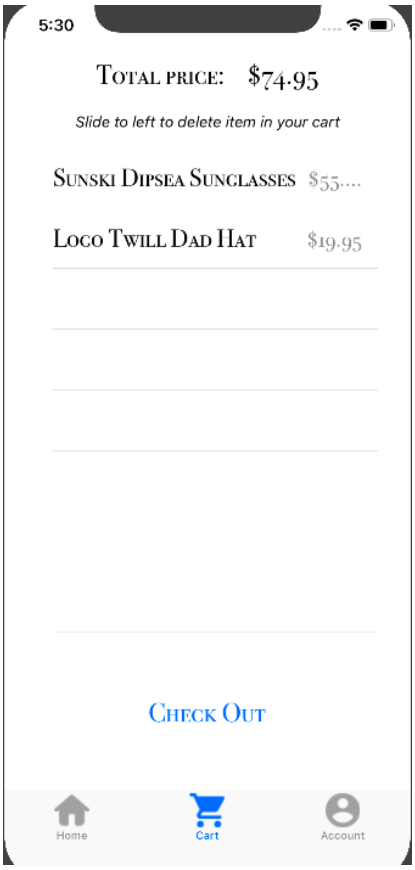
On this screen, click on Item List button on top left, it switches back to **Item List** page.

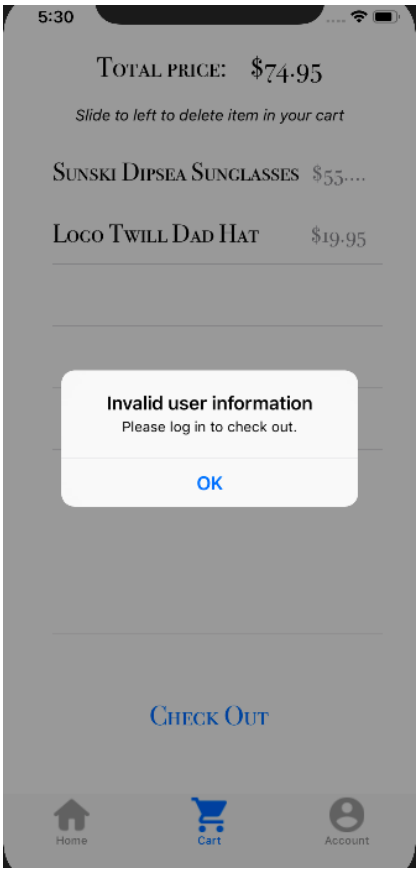
Click on *Add to Cart* button, the item will add to cart (which can be checked in **Cart** page). In addition, a message pops up showing the adding is successful.

Click on *OK* button on the pop up section, the pop up will dismiss.

The *Item Description* is scrollable.

5 – Cart page:





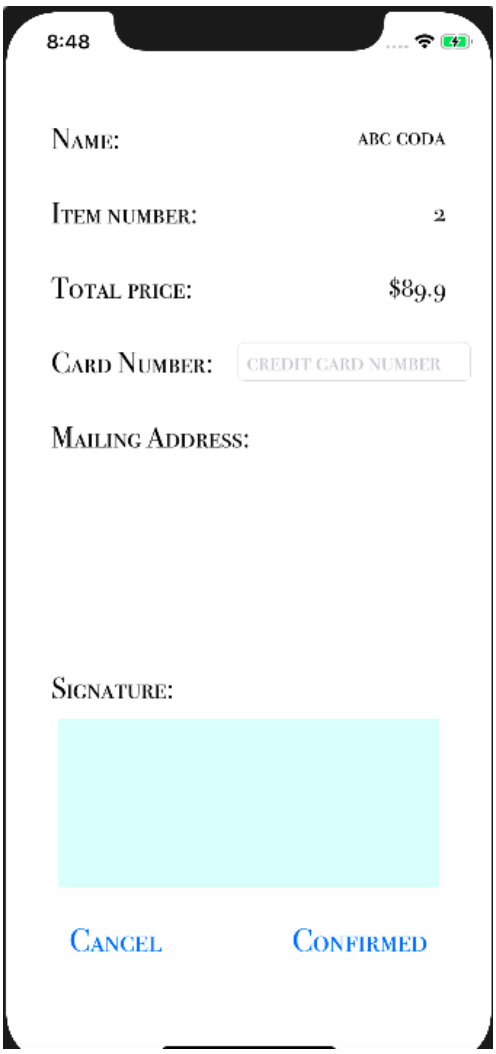
Click on the second tab on the bottom of **Home** page, **Cart** page shows up.

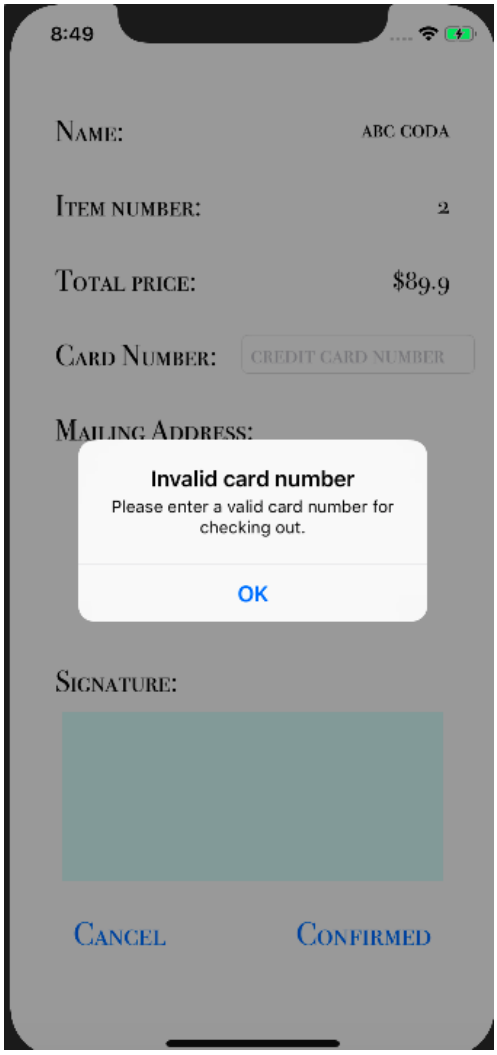
Click on the *Check Out* button on this screen, if the user haven't logged in, a warning message pops up asking user to log in first.

Click on *OK*, the pop up will dismiss.

Click on the *Check Out* button, if user has logged in, **Check Out** page shows up.

6 – Check out page:





On this screen, there are two text fields for user to input a card number and a mailing address.

On *Signature* section (with blue background), touch the area and sign. It applies Multi-touch event API to capture user's hand-writing signature.

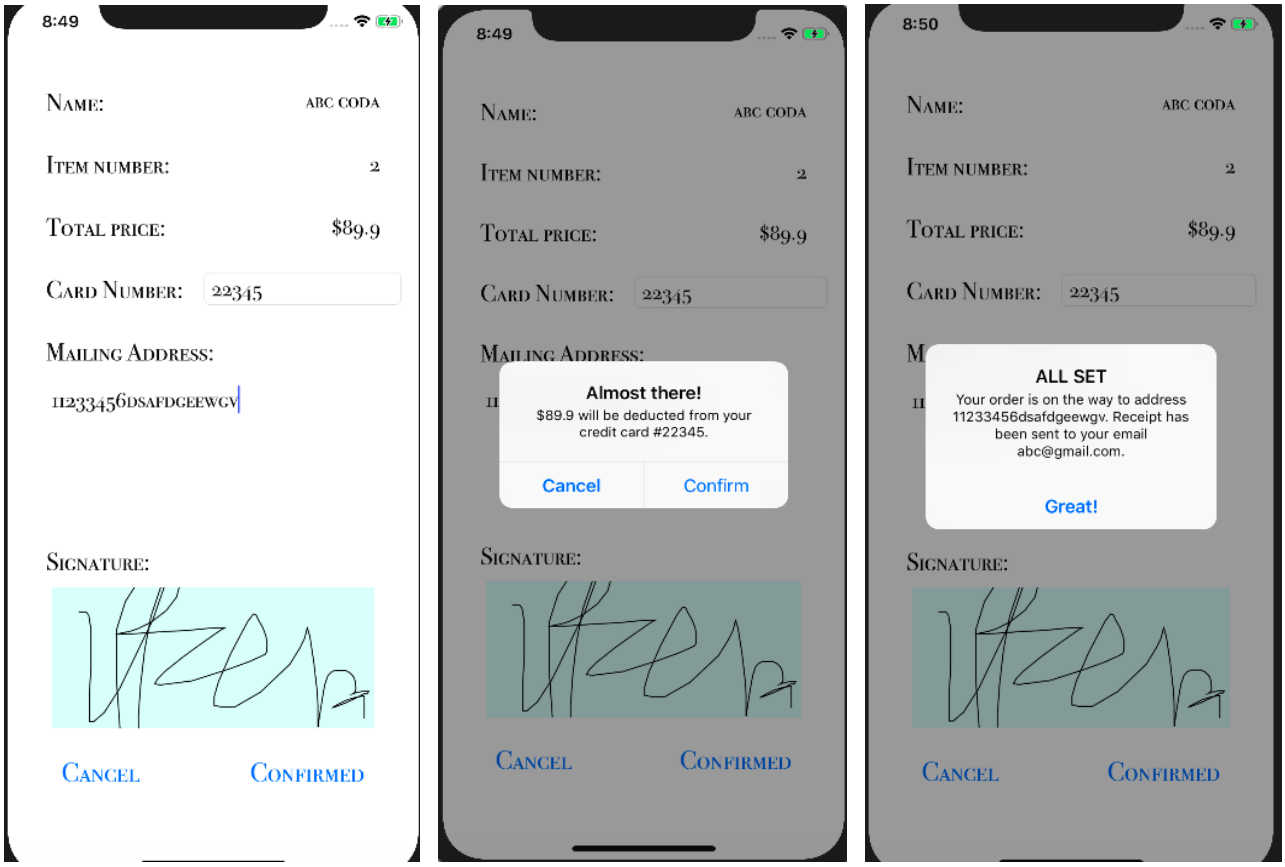
Click on *Cancel*, it will go back to **Cart** page.

Click on *Confirmed* button, if there is no card number entered, a warning message pops up.

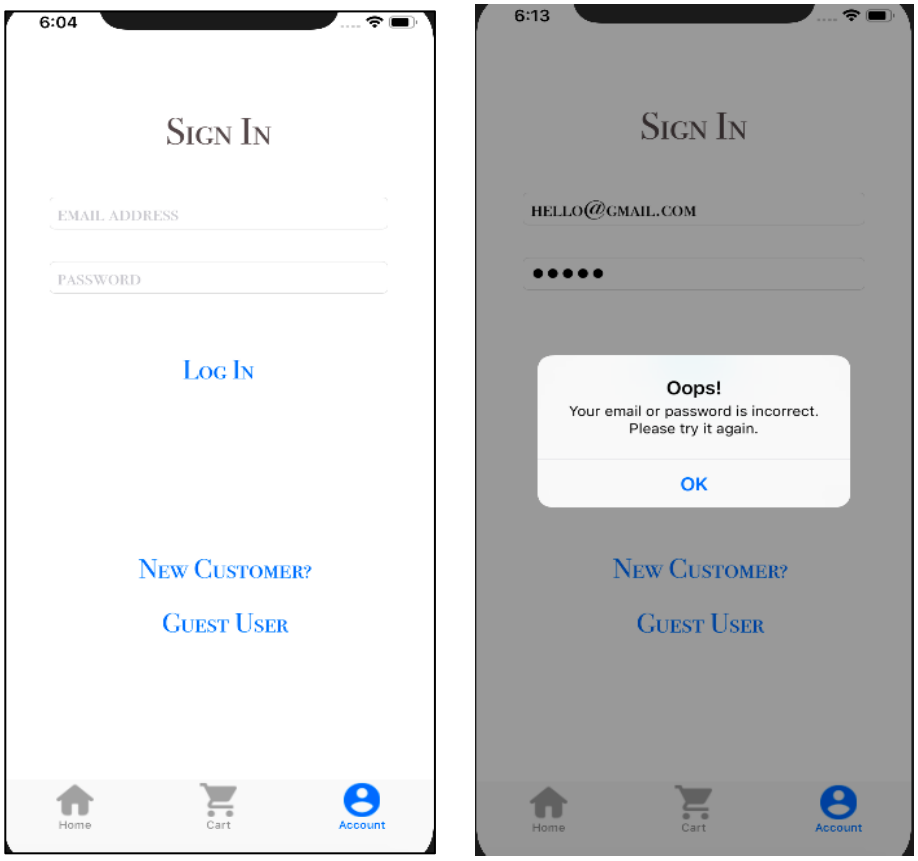
After all information filled in, click on *Confirmed* button, a message pops up for user to double check.

For the popping up message, click on *Cancel* button, the pop-up is dismissed. Click on *Confirm* button, the order is placed and a confirmation message pops up.

On the second pop-up, click on *Great!* button, app switches back to **Home** page.



7 – Account page (before log in):



On this screen, there are two text fields for user to input an email address (as username) and a password to log in.

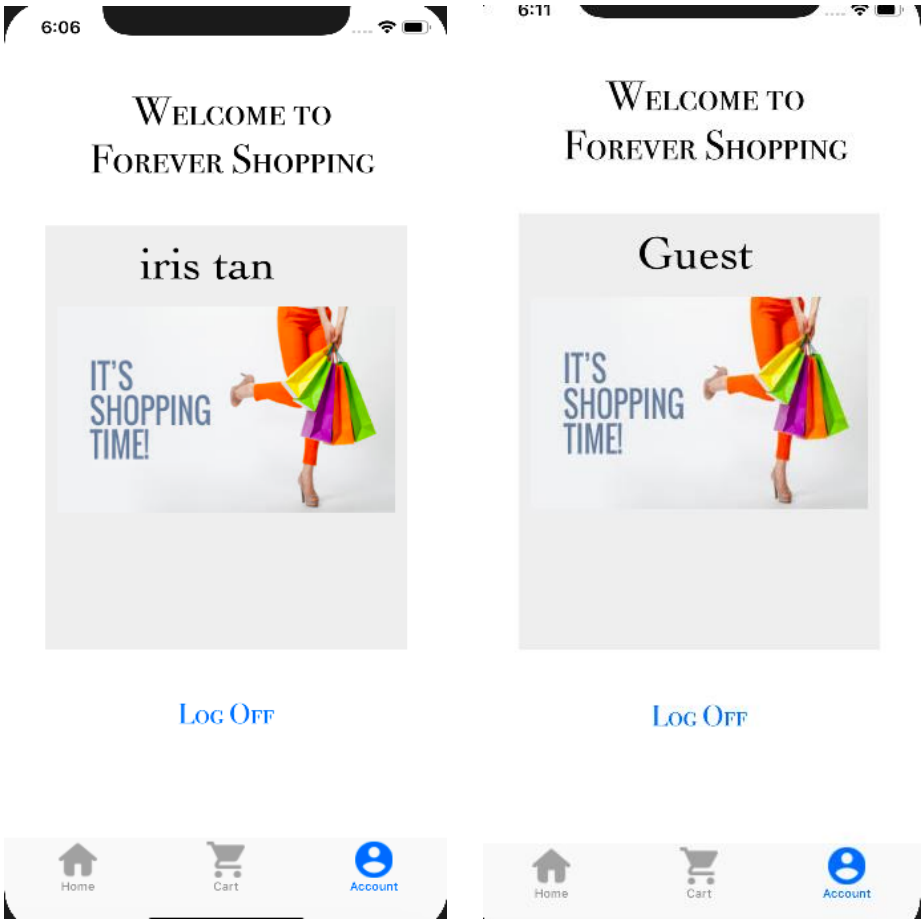
Click on *Log in* button, if the input username matches the password, the **User** page with welcome information shows up. If incorrect information is entered, a warming message pops up.

Click on *OK* button on pop-up, pop-up is dismissed.

Click on *New Customer* button, **Sign Up** page shows up.

Click on *Guest User*, guess user information is loaded into the system. It is designed for software testing or for user to try out its features.

8 – User page (after log in):

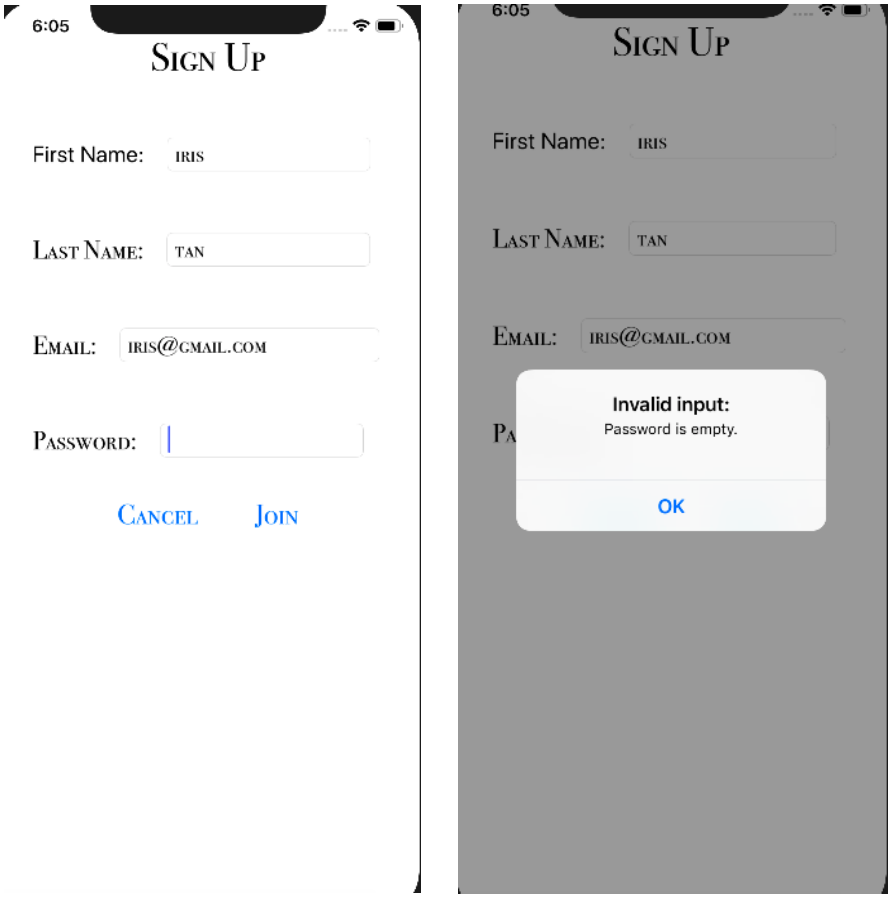


After signing in as a registered user, the User page shows up with a welcome message on screen. The message is created with Quartz 2D drawing API, consisting of the user name and a picture.

Screenshots above illustrate the different message for a registered user and Guest user.

Click on *Log Off*, it switches back to **Account** page.

9 – Sign up page:



On this screen, there are four text fields for user to input the first name, last name, email address and password.

Click on *Join* button when there is any missing information, a warming message pops up. Click on *OK* on pop-up to dismiss the message.

Click on *Join* button with all information filled in, it will switch to **User** page.

Click on *Cancel* button, it will switch back to **Account** page.

Project Discussion

Special API Features

1. **Multi-touch event API:**

   In **Check Out** page, APP requires user's hand-writing signature as a way for authentication. A UIView is created on the page (with blue background). Users can touch the area and sign, and their writing on the screen can be captured and recorded.

2. **Quartz 2D drawing API:**

   In **Account** page, after signing in as a registered user, there will be a welcome message on the upper area of screen. A UIView implementing the Quartz 2D drawing API is created for drawing the message (with grey background). APP reads from background information and draws the user name, following by a welcome picture in this area.

3. **Navigation Controller:**

   The **Item List** page is created based on navigation controller and table view controller. The default navigation bar on top can help switching between list view and detail view.

4. **Table View Controller:**

   There are table views in both **Item List** page and **Cart** page. Differently, table view in Item List page is based on UITableViewController, while in

Cart page there is a table view inside View Controller (not taught in class).

In **Item List** page, the page is only for displaying all items in specific category, so the entire view is a table view.

In **Cart** page, a table view is for listing the items in cart, and there are more functionalities need to be provided on this page based on label, button and tabs. So the table view is created within the page UI view. The implementation for the table view object is quite different from table view page in assigning the delegator, displaying the table cells, provide extra deleting features for items in cart, etc., which were not taught in class.

5. **Segue for page switching:**

   The APP implemented many segues to help scenes switching.

   On the one hand, there are normal segues created by ctrl+drag to target page, e.g., the *Back* button on **Item List** page to go back to **Home** page.

   On the other hand, there are segues integrated in the Button Click action, so that the switching through segue can be executed along with more complicated operations, such as writing information to global variables. For example, clicking *Image* buttons on **Welcome** page and **Home** page, or, clicking *Add To Cart* button on **Item Detail** page, the page switching are all implemented in this way, which was not taught in class.

Biggest Challenges

1. **Page navigation.** For an APP with multiple scenes, choose the right

approach to switch between scenes is very important to ensure its process flow.

Solution: implement different ways of scene switching to meet different requirements. Ways includes segue, dismiss operation and navigation controller.

2. **Passing values** back and forth between different scenes. Use only prepare() function for parameter passing with in-class variables makes the logic more complicated and requires a lot of work.

Solution: define global variables for parameter passing and use in-function segue (within button click action) for page switching. It makes life much easier.

3. **Functionalities in Cart page**. A table view is required for displaying the items that have been added to cart, but there needs to be more. The page should be a part of tabbed view, there are labels on top to display other information, and there is a button to process the checking out. A simple table view page makes it hard to meet all the requirements.

Solution: use a built-in table view object within the page, along with other widgets.

4. **Applying 2D draw and multi-touch API**. They were tough because there were too many stuff thrown out in the last lecture which were hard to catch one by one.

Solution: no easy solution. Just go through the lecture and sample code line by line. Figure out their ideas and try to apply them in this APP.

5. **Testing app functionalities**. Certain part of this APP requires user to log in first in order to process. Signing up/ logging in each time makes the testing tiring.

Solution: create *Guest User* button to allow a one-click log in.

Possible Further Improvements

1. **Back-end database system**: the *Forever Shopping APP* stores all users and their carts information temporarily in global variables, which will be initialized each time the APP starts over. But in real world, there is always a persistence layer applied to permanently store the information.

   It can only be done with understanding of some APIs for an iOS APP to interact with some back-end database system. Still, it is a very interesting topic and is going to be a huge improvement for this APP.

2. **Functionality of searching:** shopping app usually provides "search" functionality to make the shopping easier for users.

   I notice that in Xcode, we have a widget called *Search Bar*. I'm thinking we may be able to implement searching with this widget. We can start from it.

3. **On-page optimization**: some pages of this APP can still be optimized in some way. For example, in *Item List* table view, it will be great if user can see thumbnails of items sitting beside their names.

iOS Discussion

For mobile application, UI and code behind it are usually created separately, and they should be connected through specific operations. For example, Xcode requires developer to create so-called *outlet/ action* and *link* them to elements in view explicitly. When programming with Xcode, because of the connection issue, I found it was difficult to integrate different APP. It's hard to import scenes or features I created before into a new project, since simply copying and pasting views or controller classes does not work. And it makes the team collaboration even harder, as working on different parts of APP at the same time and integrating them together is troublesome.

Personally, I found it a little bit hard to catch up the class without my own Mac… But overall, I found iOS of much interest and fun. As an iOS developer, I am acting more like a full-stack software developer. Because I not only need to figure out how to architect the APP by myself, but also be responsible for designing the UI and business logic, as well as how to integrate them to work together. Meanwhile, the IDE of Swift provides very powerful built-in functions and features. They make life much easier as long as you learn the trick to use them and get used to that.

Personal Contributions

1. Collaborate with team member to draw up design of APP features and pages, complete the project proposal.

2. Architect the infrastructure of APP, create pages and build up the

connections between them to finalize the APP working flow.

3. Collect or compose item resources and images for APP UI and database.

4. Compose the UI for **Cart**, **Check out** pages and the corresponding business logic (the controllers); apply the multi-touch API for user signature.

5. Compose the UI for **Account** (**Log in/off), Sign up** pages, and the business logic behind it (the controllers).

6. Create class to accommodate global variables for information storage and passing.

7. Create **User** class for user information storage and display on page.

8. Compose APP's **Overview** and **Architecture Design Chart**, summarize its **Features**, and finalize the project documentation.