ITIS/CS 4180 – Mobile App Development
In Class 09

Basic Instructions:

1. In every file submitted you MUST place the following comments:
   a. Assignment #.
   b. File Name.
   c. Student Full Name.
2. This is an individual assignment, each student is expected to work alone and submit their own work.
3. Your assignment will be graded for functional requirements and efficiency of your submitted solution. You will loose points if your code is not efficient, does unnecessary processing or blocks the UI thread.
4. Please download the support files provided with this assignment and use them when implementing your project.
5. Create a zip file which includes all the project folder, any required libraries, and your presentation material.
6. Submission details:
   a. You should submit the assignment through canvas: Submit the zip file.
7. **Failure to follow the above instructions will result in point deductions.**

In Class 09 (100 Points)

In this assignment you will get familiar with Firebase to develop a contacts application which enable you to list, create, edit and delete contacts.

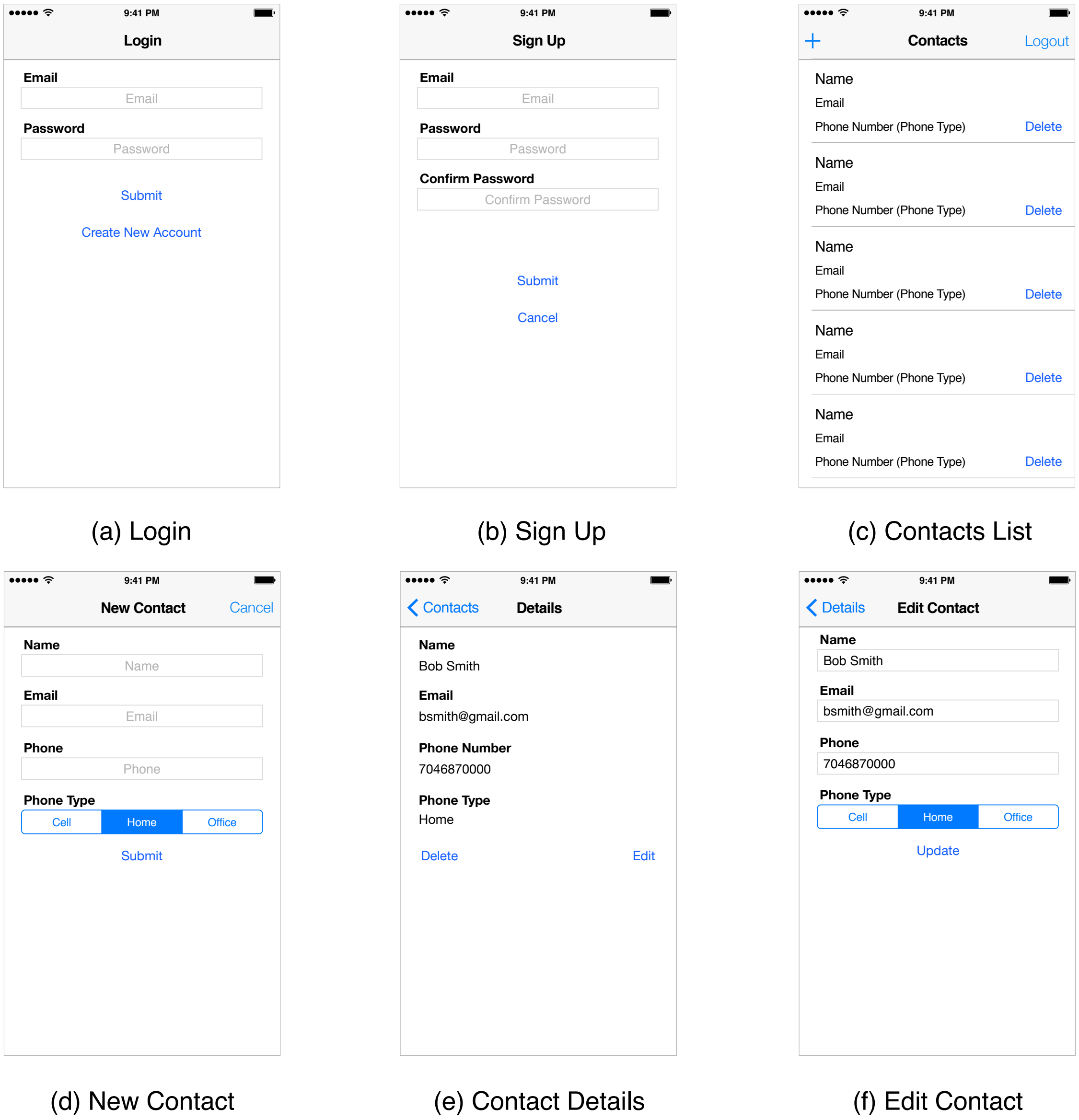

Figure 1, Application Wireframe

Part 1 : Login View Controller (10 Points)

The interface should be created to match the UI presented in Figure 1(a). The requirements are as follows:

1. Upon entering the email and password, clicking the submit button should:
   a. If all the inputs are entered correctly, you should attempt to login the user using Firebase. If the login is successful then transition to the Contacts View Controller. If login is not successful show an alert dialog highlighting the error.
   b. If there is missing input, show an alert dialog indicating missing input.

Part 2 : Sign Up View Controller (10 Points)
The interface should be created to match the UI presented in Figure 1(a). The requirements are as follows:

1. Upon entering the email, password and confirm password, clicking the submit button should:
    a. If all the inputs are entered correctly, you should attempt to signup the user using Firebase. If the signup is successful then transition to the Contacts View Controller. If signup is not successful show an alert dialog highlighting the error.
    b. If there is missing input, or if the passwords do not match show an alert dialog indicating the error.

Part 3 : Contacts List View Controller (30 Points)
The interface should be created to match the UI presented in Figure 1(c). The requirements are as follows:

1. This ViewController should be embedded in a navigation controller to provide the transition to the application ViewController.
2. This ViewController should retrieve the list of users from Firebase.
    a. You should create a Contact class to store the contact information.
    b. Make sure to setup the correct observer for the Firebase database reference to load the table when ever the data values change.
3. Clicking on a row item should segue to the Details ViewController. The Contact List ViewController should pass the required information to the Details ViewController in order to display the contact details.
4. Clicking the “Delete” button should delete the selected contact and should refresh the TableView to show the updated list of contacts.
    a. The app should delete the selected contact in Firebase, and should reload the list of contacts and refresh contacts table.
5. Clicking the “+” button should segue to the New Contact ViewController, you should use the “Present Modally” segue.
6. Clicking the “Logout” button should logout the currently logged in user and show the Login View Controller.

Part 4 : New Contact View Controller (15 Points)
The interface should be created to match the UI presented in Figure 1(d). The requirements are as follows:

1. Clicking the “Cancel” button should dismiss this ViewController and return back to the Contacts List ViewController.
2. Clicking the “Submit” button, the app should check if all the fields are entered and display an alert dialog if any of the entries is missing indicating that the missing field is required.
    a. If all the fields are entered, the app should store the new contact on Firebase. The upon completion should dismiss this ViewController and signal the Contacts List View Controller to refresh the list of users.
    b. Upon returning to the Contacts List View Controller the TableView should be refreshed and should include the newly created contact.

Part 5 : Contact Details Controller (20 Points)

The interface should be created to match the UI presented in Figure 1(e). The requirements are as follows:

1. This ViewController should display the contact details as shown in Figure 1(e).
2. Clicking the "Delete" button should perform the following steps:
    a. Delete the selected contact from Firebase. Upon completion, this ViewController should signal the Contacts List ViewController indicating that this contact was deleted successfully.
    b. Dismiss the current ViewController, which should show the Contacts List ViewController.
    c. The Contacts ViewController should reload the list of users.
3. Clicking the "Update" button should segue to the Edit Contact ViewController and should pass the contact object to the Edit Contact ViewController.

Part 6 : Edit Contact View Controller (15 Points)

The interface should be created to match the UI presented in Figure 1(f). The requirements are as follows:

1. This ViewController should display the edit form populated with the details of the contact that was passed by the Details ViewController.
2. Clicking the "Update" button, the app should check if all the fields are entered and display an alert dialog if any of the entries is missing indicating that the missing field is required.
    a. If all the fields are entered, the app should update the current contact on Firebase. After completion the app should dismiss this ViewController and signal the pervious Contact Details View Controller to refresh the contact details. In addition should signal the Contacts List View Controller that the contact information was updated.
    b. Upon returning to the Contact Details View Controller the contact details should show the updated contact information.
    c. Upon returning to the Contacts List View Controller the TableView should be refreshed and should include the newly created contact.