ITIS/CS 4180 – Mobile App Development
In Class 07

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 07 (100 Points)

In this assignment you will get familiar with iOS TableViews and making simple API requests. You will develop a contacts application which enable you to list, create, edit and delete contacts.

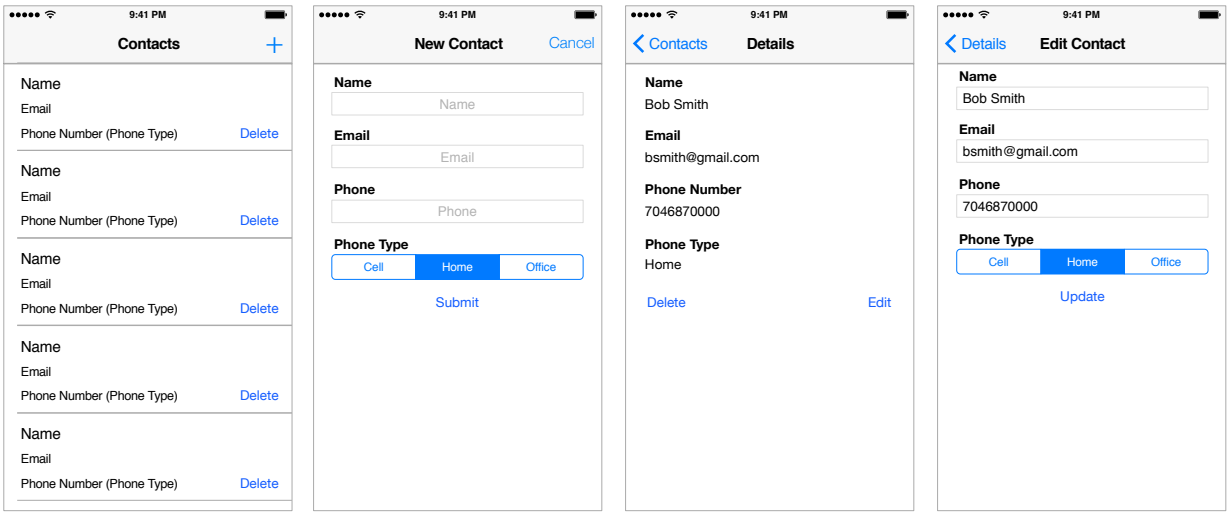
API Details:

Get Contacts API	
End Point	http://ec2-18-234-222-229.compute-1.amazonaws.com/contacts
Method	GET
Parameters	None

Create Contact API	
End Point	http://ec2-18-234-222-229.compute-1.amazonaws.com/contact/create
Method	POST
Parameters	- name: String value of the name - email: String value of the email - phone: String value of the phone - type: String value CELL, OFFICE or HOME

Update Contact API	
End Point	http://ec2-18-234-222-229.compute-1.amazonaws.com/contact/update
Method	POST
Parameters	- id: The contact id number - name: String value of the name - email: String value of the email - phone: String value of the phone - type: String value CELL, OFFICE or HOME

Delete Contact API	
End Point	http://ec2-18-234-222-229.compute-1.amazonaws.com/contact/delete
Method	POST
Parameters	- id: The contact id number



(a) Contacts List (b) New Contact (c) Contact Details (d) Edit Contact

Figure 1, Application Wireframe

Part 1 : Contacts List View Controller (30 Points)

The interface should be created to match the UI presented in Figure 1(a). 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 using the “Get Contacts” API.
   a. You should integrate AlamoFire library into your project and use it to make the API requests.
   b. You should create a Contact class to store the contact information.
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 call the “Delete Contact” API after the API is completed you should reload the list of users calling the “Get contacts” API and refresh the list.
5. Clicking the “+” button should segue to the New Contact ViewController, you should use the “Present Modally” segue.

Part 2 : New Contact View Controller (20 Points)

The interface should be created to match the UI presented in Figure 1(b). 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 call the “Create Contact” API. After the API is completed the app 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. This should be done by reloading the list of users by calling the “Get Contacts” API and refresh the list.

Part 3 : Contact Details Controller (25 Points)

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

1. This ViewController should display the contact details as shown in Figure 1(c).
2. Clicking the “Delete” button should perform the following steps:
   a. Call the “Delete Contact” API. After the API is completed 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 calling the “Get contacts” API and refresh the list.
3. Clicking the “Update” button should segue to the Edit Contact ViewController and should pass the contact object to the Edit Contact ViewController.

Part 4 : Edit Contact View Controller (25 Points)

The interface should be created to match the UI presented in Figure 1(d). 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 call the “Update Contact” API. After the API is completed 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. This should be done by reloading the list of users by calling the “Get Contacts” API and refresh the list.