CS 4520/5520: Mobile Application Development
Weekly Assignment 6 (100 points)

Basic Instructions:

1. **Submission details:**
   a. The project name should be: **WA6_<Your_Last_Name>_<Last_4_digits_of_your_NUID>**
   b. After you are done building the project, right-click on the project from Project Navigator.
   c. Click on Show in Finder.
   d. Select all the files and folders, including the .xcworkspace file, and compress them to create a zip file.
   e. Then submit the zip file in Canvas.
   f. (Please read the "How to Submit the Assignments" from Modules -> Quick Links)

Weekly Assignment 6

In this assignment, you will use the same API **(text API, not the JSON one)** as App10 and extend its functionalities. I will **not** provide you with any design for this assignment. You need to carefully create a usable design with the elements you learned so far.

Requirements:

1. We need to add three more functionalities to the current application.
   a. Display a contact.
   b. Delete a contact.
   c. Edit a contact.
2. **Display a contact:**
   a. The user should be able to see the details of a selected contact. Do not use AlertController for this. You can create a new screen to display the details.
   b. Do not send different variables: name, email, and phone, from one screen to another. Use a struct or class that includes those attributes to do that.
3. **Delete a contact:**
   a. The user should be able to delete a contact. You can do that in a couple of ways.
      i. You can do that with the help of menus and accessories on the main screen.

ii. You can also do that from the display screen.
    1. If you delete the contact from the display screen, you should **use the notification center** to notify the main screen so that it can load the updated contacts. Do not use delegates; use the observer handlers to reload data.

iii. **You must ask the user if they are sure to delete the contact.** Use AlertController to do that. AlertControllers can have multiple buttons and handlers together.

iv. Check the contacts app on your iPhone or the iPhone emulator for cues. You do not have to design it the same; focus on the basic requirements.

4. **Edit a contact:**
    a. This one is tricky 🙂There is no direct API endpoint to edit our contacts.
    b. So, how would you edit a contact?
        i. The easiest way is to use the **delete** and **add** API endpoints together.
        ii. First, you can retrieve what the user put while editing the contact, update the local contact object, and then call ‘delete’ to delete the old contact from the server.
        iii. Then, call ‘add’ to add a new contact to the server.
        iv. Important: You should not just call the ‘delete’ and ‘add’ functions one after another.
            1. **‘add’** should be called after you get a response for the **‘delete’** call through Alamofire.
            2. So the ‘delete’ call goes through first, then, when you get the response back, and if it is a 200-level code, then you will call ‘add’.
        v. Do not send different variables: name, email, and phone, from one screen to another. Use a struct or class that includes those attributes to do that.
        vi. Once the Edit Screen is loaded, it should display the attributes (name, email, phone) of the contact the user is editing.
        vii. Once the Edit is done, you should use the notification center to notify the main screen and the details screen that the contact list has been updated, and it should reload the data. Do not use delegates; use the observer handlers to reload data.

5. **Validations:**
    a. The email has to be valid.
    b. The phone number has to be an integer.
    c. None of the inputs should be empty.

6. **Display and Edit Contact Screens should be scrollable.**