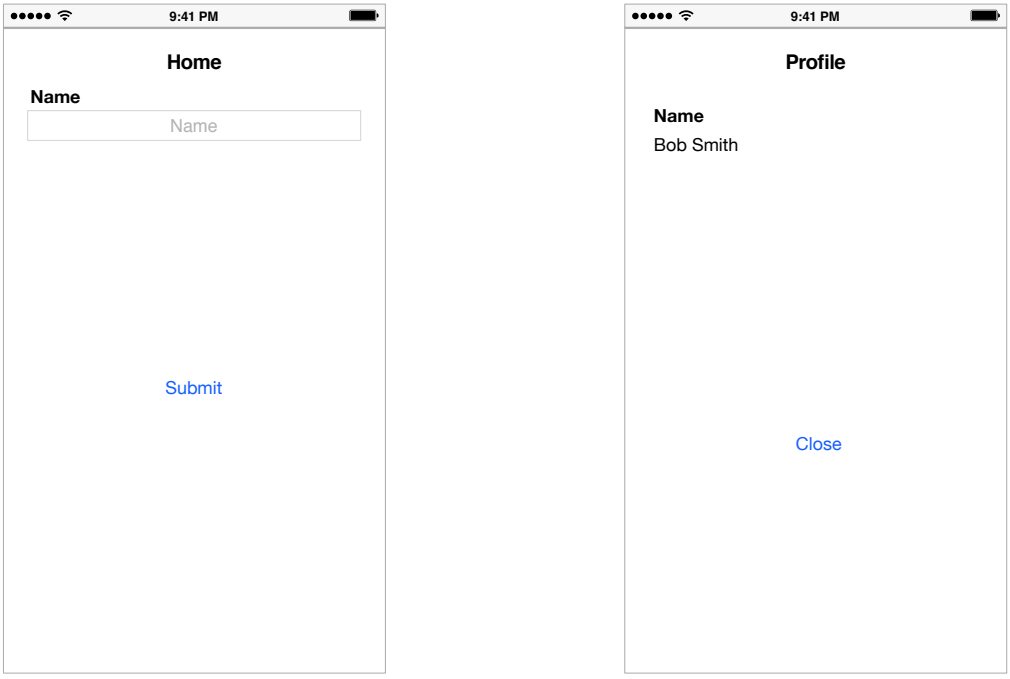
ITIS/CS 4180 – Mobile App Development
In Class 03

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 03 (100 Points)

In this the refresher assignment for this in-class, which you have to complete and demonstrate to the TA before proceeding to the assignment.



(a) Home View Controller **(b) Profile View Controller**

Figure 1: Application Wireframe

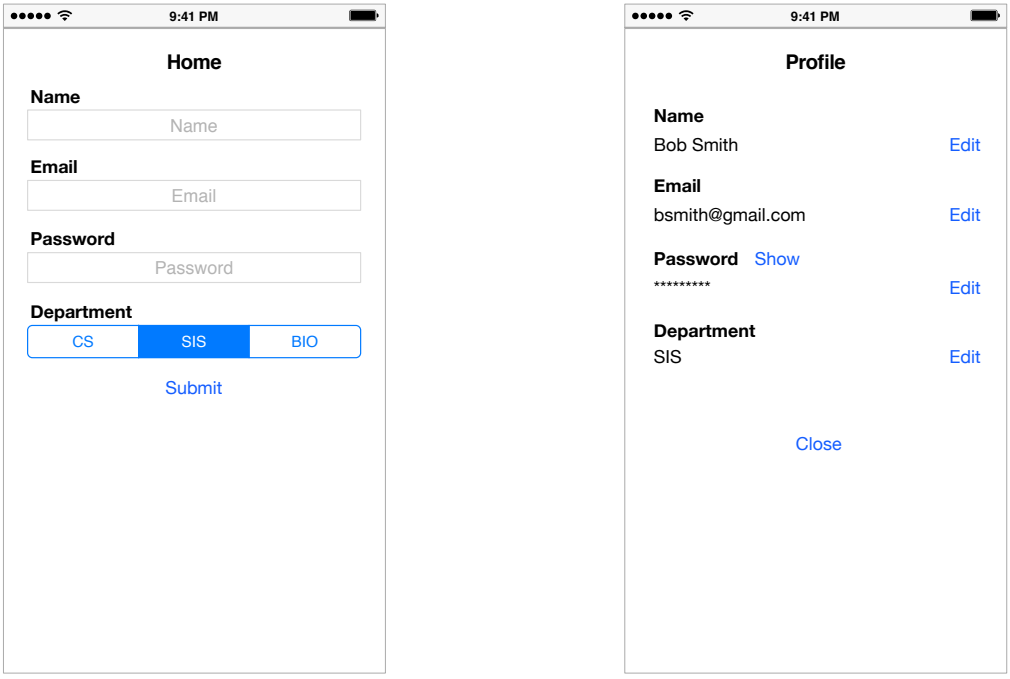
Part 0 : Refresher (5 Points)

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

- The user enters the name in the home view controller, then upon clicking “Submit”
  - The entered name should be retrieved.
  - The retrieved name should be sent to the profile view controller.
  - The profile view controller should be presented and should show the entered name field.

In Class 03

In this assignment you will build an iOS application that uses multiple view controllers and investigates communication and data passing between them.



(a) Home View Controller **(b) Profile View Controller**

Figure 1: Application Wireframe

Part 1 : Home View Controller (15 Points)

The interface should be created to match the UI presented in Figure 2(a). The requirements are as follows:

- Upon clicking “Submit”
  a. If any of the profile fields are missing display an alert indicating that the missing field is required.
  b. If all the fields are provided correctly, then store the provided data in a User Object. Then pass the User object to the Profile View Controller and then display the Profile View Controller.

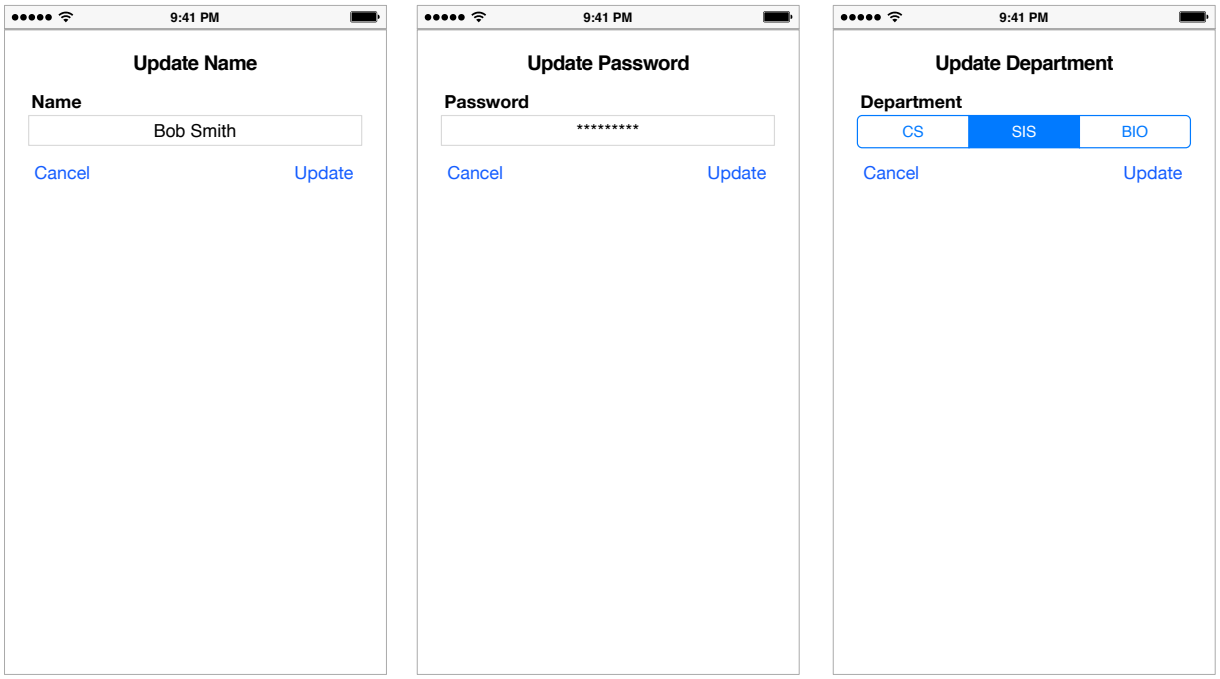
Part 2 : Profile View Controller (40 Points)

The interface should be created to match the UI presented in Figure 2(b). The requirements are as follows:

1. Upon loading the Profile View Controller, it should display the content of the User object as shown in Figure 2(b). It should display the name, email, password and department.
   a. Note that the Password Label is displayed as “****”, the number of stars should match the number of characters in the password field.
   b. If the “Show” button is clicked the Password Label should show the actual

password field. Then the “Show” button text should be changed to “Hide”, upon clicking “Hide” the password field should be hidden and shown as “*****”, and then the “Hide” button text should be changed to “Show”.

2. Clicking the “Edit” button beside each field should open the corresponding Edit View Controller and pass the required data.



(a) Edit Name **(b) Edit Password** **(c) Edit Department**

Figure 2: Application Wireframe

Part 3 : Edit View Controllers (40 Points)

The interface should be created to match the UI presented in Figure 3. Feel free to create multiple view controllers to provide the different functionality required for this part. The requirements are as follows:

1. The update view controller should be implemented for the different user profile fields as shown in Figure 3.
2. Clicking the “Cancel” button should simply dismiss the current view controller which should show the previously displayed view controller.
3. Clicking the “Update” button should pass the updated profile value back to the Profile View Controller and should dismiss the current view controller.
4. Note that the Edit View controller is displayed using “Present Modally”, you should consider using “**Unwind Segues**” to pass data back to the Profile View Controller.
5. Upon returning to the Profile View Controller, the updated value should be updated and displayed in the Profile View Controller.