**Institute of Vocational Education**
**Information Technology Discipline**

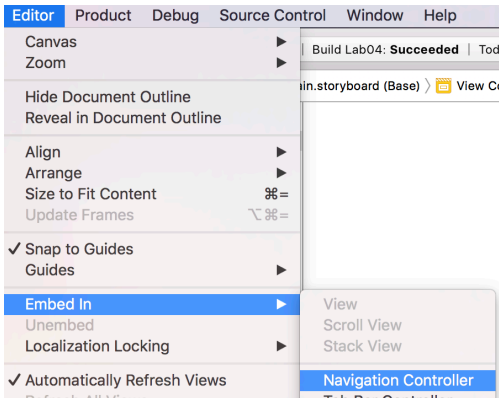
ITP4206 - Proprietary Mobile Application Development
Lab 4 - MVC and Multiple View Controllers

Description:
Study MVC Pattern and using multiple view controllers in iOS.

Task 1 - MVC Pattern

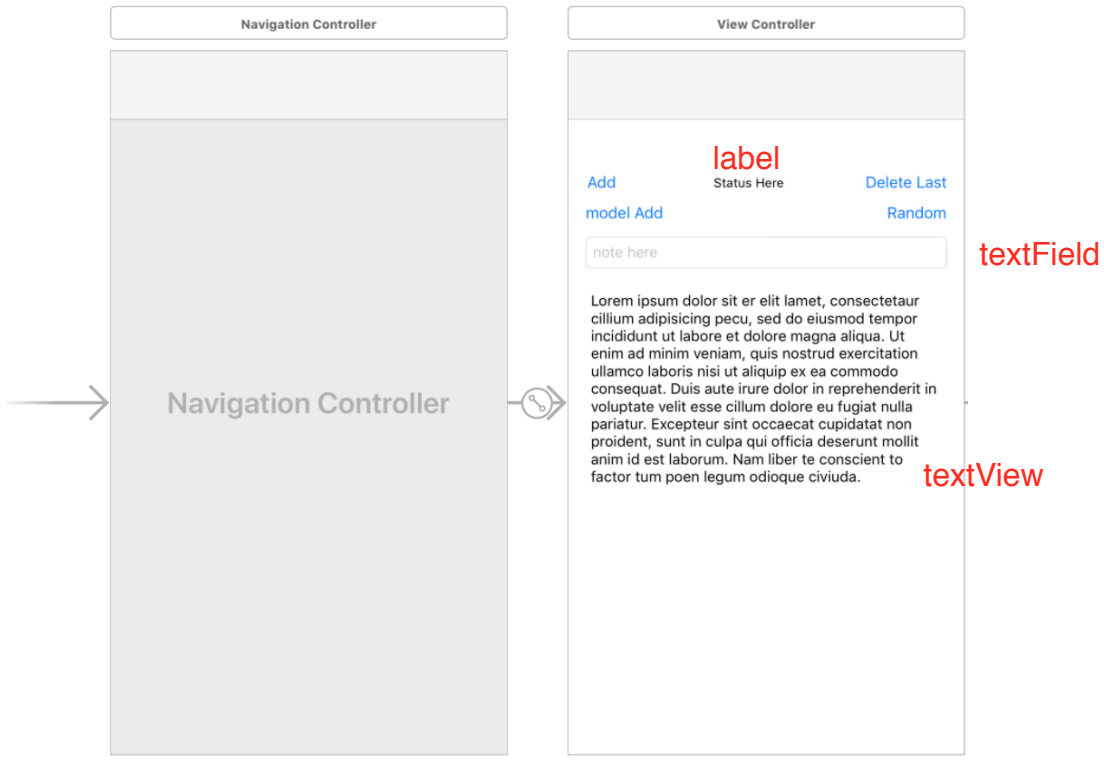
1. Create an iOS single view project, name: Lab04, company Identifier: edu.self, Devices: Universal. Don't forget setting the language to **Swift**.

2. Go to the main storyboard, complete the following UI:
   a. Select the preloaded View Controller, select **Editor -> Embed in -> Navigation Controller** in menu bar.




'

   b. Create the UI (in story file) and **@IBOutlet (in swift file, don't miss this!)** according to the given graph:



3. Create the model class NoteManager.swift (superclass: NSObject) as follow:

```
import UIKit

class Note : NSObject {
    var text : String
    init(text : String) {
        self.text = text;
    }
}

class NoteManager: NSObject {
    private var notes = [Note]()

    func appendNote(expense : Note) {
        notes.append(expense)
    }

    func removeLastNote() -> Note? {
        if notes.count < 1 {
            return nil;
        }
        return notes.removeLast()
    }

    func count() -> Int {
        return notes.count
    }

    func note(at index : Int) -> Note {
        return notes[index]
    }
}
```

4. Go back to **ViewController.swift**, define a noteManager object and check if the following outlets are created and linked (connect the UI components with storyboard if the outlet shows a hollow circle):

```
        var noteManager =  NoteManager()
        @IBOutlet var label : UILabel!
        @IBOutlet var textView : UITextView!
        @IBOutlet var textField : UITextField!
```

5. Create a **showNoteOnTextView** function for reuse purpose

```
func showNoteOnTextView() {
    var text = "";
    for i in 0..<noteManager.count() {
        let note = noteManager.note(at: i)
        if i == 0 {
            text = "\(note.text)"
        } else {
            text = "\(text)\n\(note.text)"
        }
    }
    self.textView.text = text;
}
```

6. Create and complete the following two IBAction functions (Don't forget to connect the IBAction with the two buttons (**Add** and **Delete Last**) in storyboard:

```
@IBAction func addBtnClick(_ sender : AnyObject){
    //get the text field text
    if let text = textField.text {
        //check if the textfield is empty or note
        if text.characters.count > 0 {
            //create a note object, append the note to the manager
            let note = Note(text: text)
            self.noteManager.appendNote(expense: note)
            //update the UI
            self.label.text = "\"\(note.text)\" added"
            self.showNoteOnTextView()
            self.textField.text = ""
        } else {
            self.label.text = "textfield is empy"
        }
    }
}

@IBAction func deleteBtnClick(_ sender : AnyObject){
    //remove and get the last note object
    if let note = self.noteManager.removeLastNote() {
        //update the UI
        self.label.text = "\"\(note.text)\" removed"
        self.showNoteOnTextView()
    }
}
```

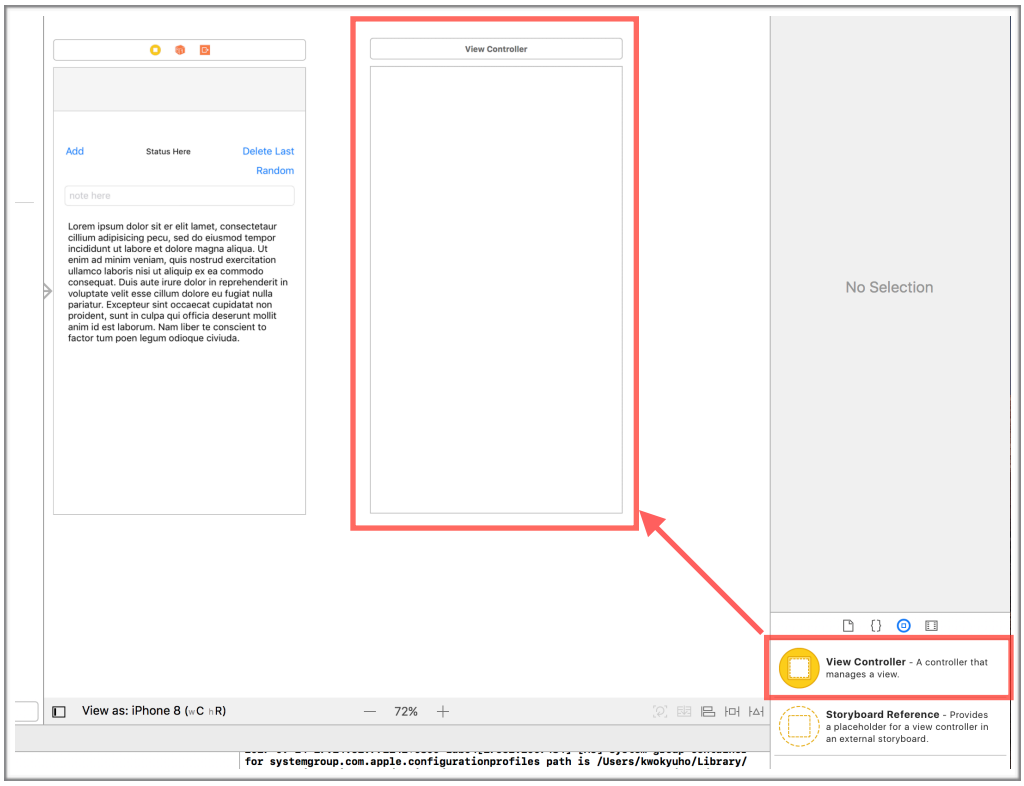
7. Override the viewWillAppear function, this function will call when the view is going to show up.

```
override func viewWillAppear(_ animated: Bool) {
    super.viewWillAppear(animated)
    self.textView.text = ""
    self.showNoteOnTextView();
}
```

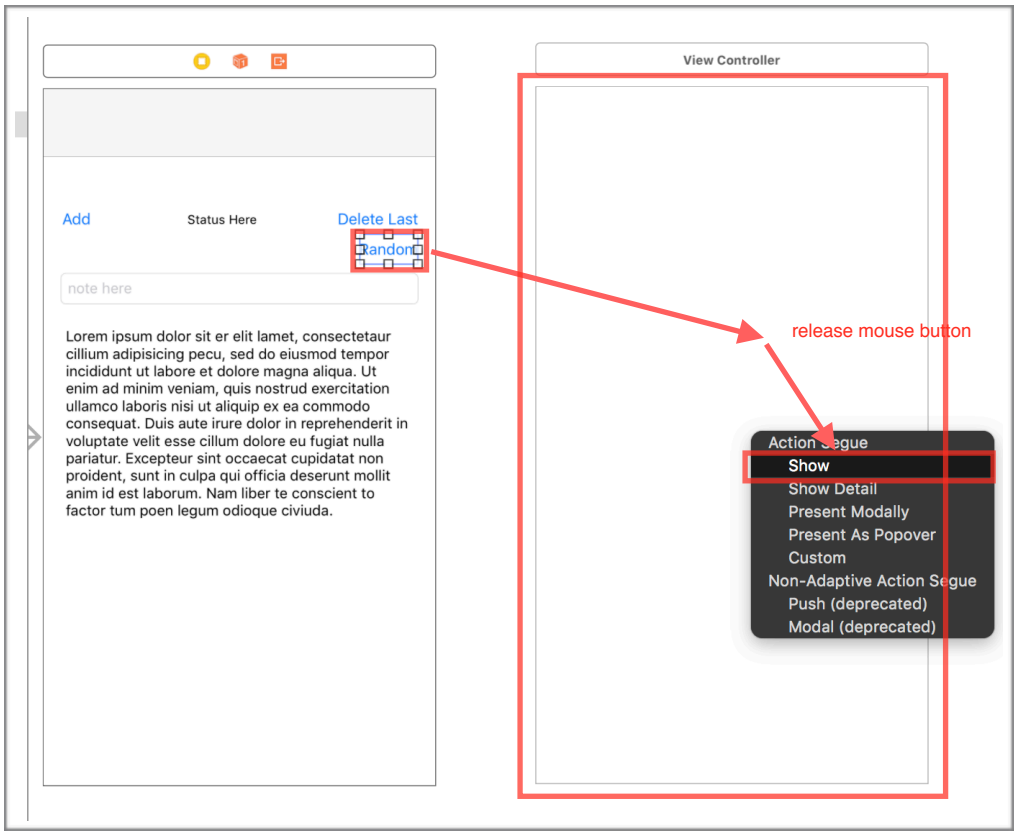
8. Build and Run. At this stage, you could be able to:
   (i) Input note in textfield, add the note and show all stored notes in textview.
   (ii) Remove lastly appended note and update the notes in textview.
   (iii) Show add/delete/empty status in label view.

Task 2 - Multiple VC with Navigation Controller

9. Drag a new View Controller into storyboard.

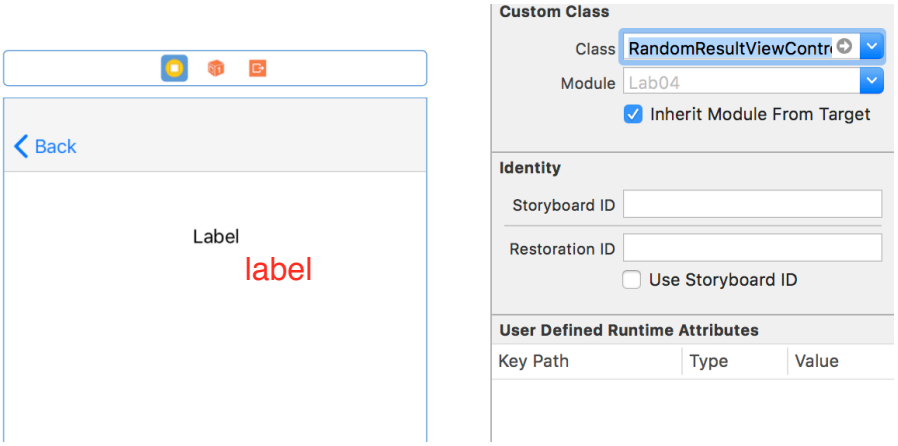



10. At “**Random**” button, **Ctrl-Left-Click-drag** (or **Right-Click-Drag**) a “**Show**” segue to the newly created View Controller

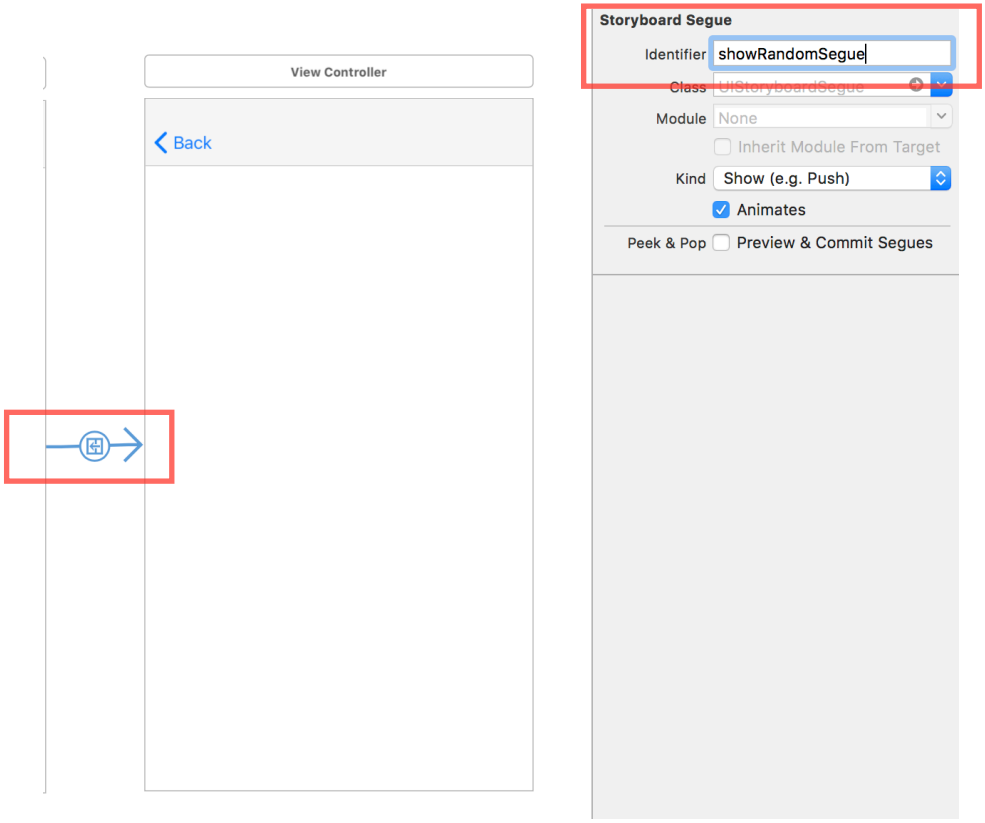


11. Create a new Swift Cocoa Touch Class named **RandomResultViewController.swift** (superclass: **UIViewController**).

12. Connect the Swift class with the new view controller using identity inspector. **Create and connect a UILabel outlet named label**.




13. Select the segue and give it an identifier. (name: **showRandomSegue**)



14. Go to **RandomResultViewController.swift**, update the class as follow:

```
import UIKit

class RandomResultViewController: UIViewController {

    var randomResult : String = "no result"
    @IBOutlet var randomResultLabel : UILabel!

    override func viewDidLoad() {
        super.viewDidLoad()

        // Do any additional setup after loading the view.
    }

    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
        self.randomResultLabel.text = self.randomResult;
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

15. Override the prepare for segue function in **ViewController.swift**. This function will call if the segue (showRandomSegue) is triggered:

```
//called before the transition
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    //check the segue id
    if segue.identifier == "showRandomSegue" {
        //check if the vc is RandomResultViewController
        if let destination = segue.destination as? RandomResultViewController {
            if self.noteManager.count() > 0 {
                //generate a random index and pick a note object from manager at random index
                //assign the text value of that note to RandomResultViewController
                let randomNo = Int(arc4random()) % self.noteManager.count()
                let note = self.noteManager.note(at: randomNo)
                destination.randomResult = note.text
            }
        }
    }
}
```

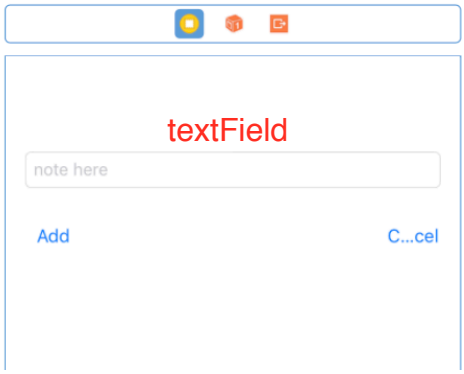
16. Build and Run. At this stage, you could be able to:
    (i) When the “Random” button is clicked (and the note array in NoteManager is not empty), the source will randomly pick a random note and assign the value to next page (RandomResultViewController).
    (ii) In the next page, the random text will be shown on the label.

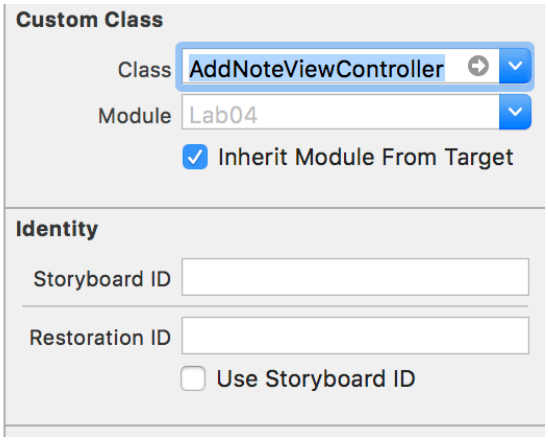
Task 3 - Modal View Controller and Unwind Segue

17. **Drag a new ViewController** into storyboard (place it above ViewController).

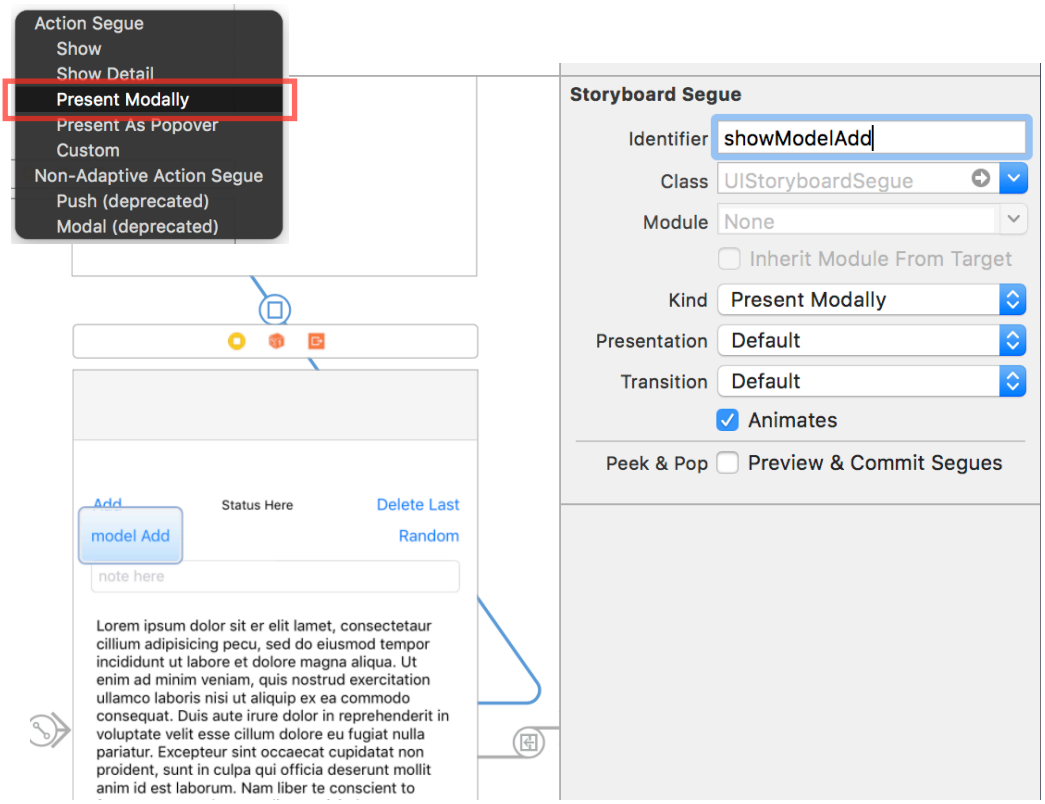
18. Create a new Cocoa Touch Class, **AddNoteViewController** (superclass: **UIViewController**).

19. Connect the class with the new view controller using identity inspector, construct the UI and IBOutlet as following as well:




20. Create a **"Present Modally" segue** from **ViewController** to **AddNoteViewController**, and name the segue as "**showModelAdd**".



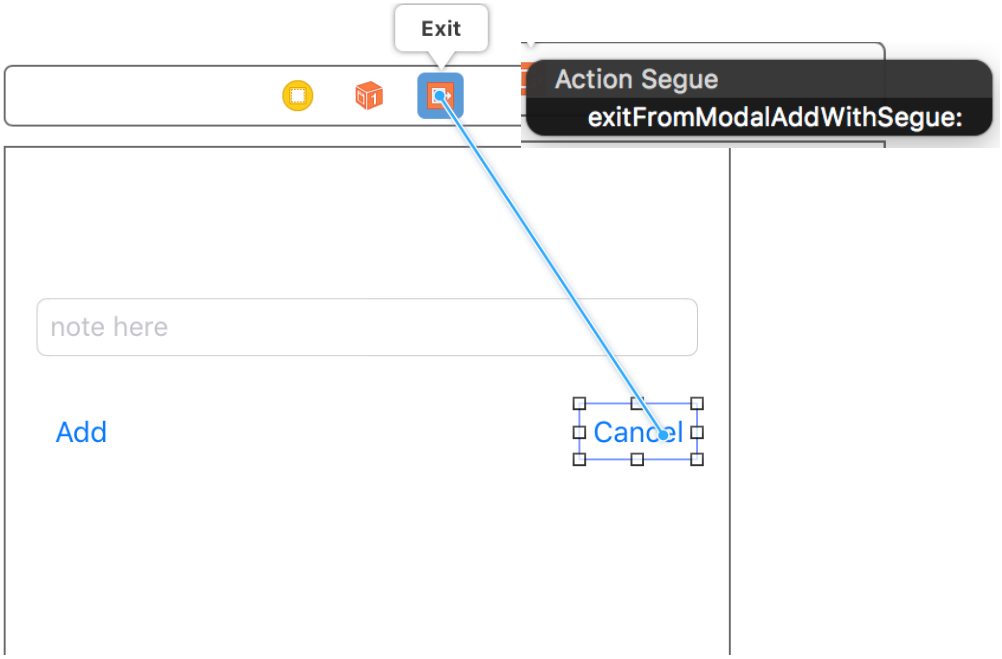
21. Update the **AddNoteViewController.swift** code as following:

```
import UIKit

class AddNoteViewController: UIViewController {

    @IBOutlet var textField : UITextField!

    override func viewDidLoad() {
        super.viewDidLoad()

        // Do any additional setup after loading the view.
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

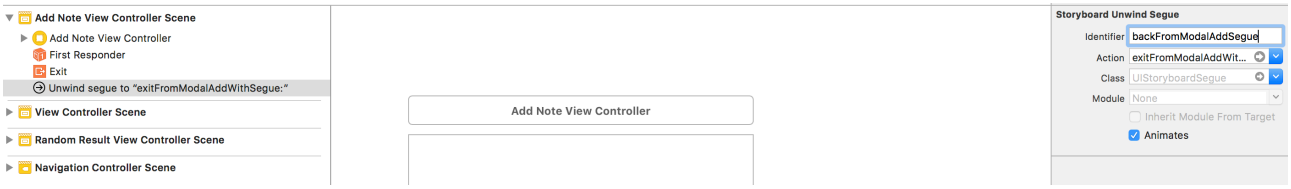
22. Go to **ViewController.swift**, create an **unwind segue** function named **exitFromModalAdd**.

```
//unwind segue function (take 1 argument with UIStoryboardSegue type)
@IBAction func exitFromModalAdd(segue : UIStoryboardSegue){
    if segue.identifier == "backFromModalAddSegue" {
        //we get the "source" instead of "destination" this time.
        if let source = segue.source as? AddNoteViewController {
            if source.textField.text!.characters.count > 0 {
                let note = Note(text: source.textField.text!)
                self.noteManager.appendNote(expense: note)
                self.label.text = "\"\(note.text)\" added"
                self.showNoteOnTextView()
            }
        }
    }
}
```

23. Go back to the storyboard, put your focus on **AddNoteViewController**. Connect the unwind segue from **“Add” (do it first)** and **“Cancel” (do it last)** button. Link it with **exitFromModalAdd** function.




24. Name the identify of the unwind segue for “Add” button as **“backFromModalAddSegue”**.




25. Run and see the final result! You should be able to:
    (I) prompt a modal view controller (AddNoteViewController)
    (II) add a new note from the modal view controller from unwind segue.

26. Finished. Submit the project to moodle in *.zip format!