Assignment 4

The manufacturing company was excited to get a prototype of the new inventory tracking tool. After using it there are several new features they are requesting. The company has multiple manufacturing plants that each have separate inventory systems. Add support for multiple manufacturing plants. Additionally, the company would like saving and loading support.

Required Tasks

1. Update your model objects with properties and methods as discussed in the Interface section.
    a. The *PreTest.h/m* unit test must build and run without errors.
2. Provide unit tests to verify all of the methods and properties in InventoryStore, Inventory, and Item.
    a. Create an isEqual for each object.
    b. Verify save/load to disk using NSFileManager.
    c. Reset the InventoryStore and clear the Documents folder for each unit test that tests save/load logic.
3. Add a root table view controller, CompanyInventoryController, to manage the display of multiple inventories.
    a. On app start, users should see a list of inventories.
    b. Create a custom UITableViewCell to display the inventory name, total cost, total items, and modified date.
    c. Tapping on a table view cell should show the Inventory's item list.
4. Save and load changes to disk on app start and close via multi-tasking.
    a. Track changes to inventories.
        i. If an item or inventory was created, the modified flag should be YES.
        ii. If the InventoryStore was saved, all modified flags should be NO.
        iii. If no inventory was modified, don’t re-save.
    b. Create a save directory “Inventory” in the Documents folder. Save all data files here.
5. Adding a new Inventory should show a modal InventoryDetailViewController to get input from the user.
    a. On the top bar two UIBarButtonItems should exist “Create” and “Cancel”
        i. Create - add the inventory if the name is not blank.
        ii. Cancel - do not add the new inventory.
    b. Display a name label and textfield to get a name for the inventory item.
    c. Display an “Add Inventory” button to create a new Inventory in the InventoryStore model object.
6. Adding a new Item should show a modal ItemDetailViewController to fill out all the fields.
    a. On the top bar two UIBarButtonItems should exist “Create” and “Cancel”
        i. Create - add the item if the name, current quantity, and unit cost is not blank.

    ii. Cancel - do not add the new item.

7. Do not add random Item or Inventory objects. Users should be able to edit an Item or Inventory object as they add it.
8. Add swipe to delete rows in the CompanyInventoryController and InventoryItemController

Property and Method Interface

Your code must build and run the PreTest.h unit test. This file checks that you conform to the interface. You must provide comprehensive unit tests for each method and class.

InventoryStore : NSObject

```
+ (InventoryStore *)shared;
+ (void)setShared:(InventoryStore *)inventoryStore;
+ (BOOL)saveShared;
+ (BOOL)loadShared;
+ (NSString *)inventoryStorePath;

- (id)init;
- (BOOL)saveInventoryStoreToFile:(NSString *)path;
- (BOOL)loadInventoryStoreFromFile:(NSString *)path;

- (void)addInventory:(Inventory *)inventory;
- (void)addInventory:(Inventory *)inventory atIndex:(NSUInteger)index;
- (Inventory *)inventoryAtIndex:(NSUInteger)index;
- (NSUInteger)count;

- (void)removeInventory:(Inventory *)inventory;
- (void)removeInventoryAtIndex:(NSUInteger)index;
- (void)removeAll;
```

Inventory : NSObject <NSCoding>

```
@property (nonatomic, copy) NSString *name;
@property (nonatomic, strong) NSDate *lastModified;
@property (nonatomic, assign) BOOL isModified;

- (CGFloat)totalCost;
- (NSInteger)totalItems;

- (void)addItem:(Item *)item;
- (void)addItem:(Item *)item atIndex:(NSUInteger)index;
- (Item *)itemAtIndex:(NSUInteger)index;
- (NSUInteger)count;

- (void)removeItem:(Item *)item;
- (void)removeItemAtIndex:(NSInteger)index;
- (void)removeAll;
```

Item : NSObject <NSCoding>

```
@property (nonatomic, copy) NSString *name;
@property (nonatomic, assign) NSInteger currentQuantity;
@property (nonatomic, assign) CGFloat unitCost;
@property (nonatomic, strong) NSDate *lastInventoryDate;
```

Hints

- Read about UINavigationController and NSFileManager
- Look at the UIViewController methods `viewDidAppear`: etc.

Evaluation (100 points)

Assignments will lose points for the following:

- Build errors.
- Build warnings.
- Missing files.
- NSLog() print statements.
- Lack of Javadoc-style method comments.
  - Assume the reader is familiar with Objective-C. Explain what methods do in your header files and complex logic in your implementation files.
- Incomplete or missing required specifications.
- Code style does not match “Coding Guidelines for Cocoa.”

Grading

- Style/Documentation 30
- Organization 20
- Functionality 50

Extra Credit

1. (3 points) Use a horizontal swipe gesture to switch back to the CompanyInventoryController from an InventoryViewController.
2. (2 points) Use a timer to autosave in the background every 2 minutes.