Core data is another way to store persistent data.

From the apple docs:

The Core Data framework stack provides a flexible alternative to directly managing data using SQLite or other data storage mechanisms. By providing an object oriented abstraction layer on top of the data the task of managing data storage is made significantly easier for the iOS iPhone application developer.

**\*Important- Only Master-Detail Application, Utility Application and Empty Application project templates offer the option to automatically include support for Core Data.**

**These will be our steps**

We will go through the finished version first.

**Part 1: Prep**

**Create an Empty View Application and make sure that the Devices menu is set to iPhone and that the check boxes next to Use Core Data and Use Automatic Reference Counting are selected. In the Product Name and Class Prefix fields enter CoreData**

**A file named CoreData.xcdatamodeld should have been created…select this file and look at the interface it loaded**

**Click Add Entity and name it Contacts.**

**Add three attributes: name, address and phone and make them all String datatypes**

**Now Create a new class named CoreDataViewController and make it a subclass of the UIViewController. Make sure you click create xib.**

**Open the CoreDataAppDelegate.h file and modify it so that the CoreDataViewController is the root controller**

**Here is the new code:**

#import <UIKit/UIKit.h>

#import "CoreDataViewController.h"

**//note the @class designation**

@class CoreDataViewController;

@interface CoreDataAppDelegate : UIResponder <UIApplicationDelegate>

@property (strong, nonatomic) CoreDataViewController \*viewController;

@property (strong, nonatomic) UIWindow \*window;

@property (readonly, strong, nonatomic) NSManagedObjectContext \*managedObjectContext;

@property (readonly, strong, nonatomic) NSManagedObjectModel \*managedObjectModel;

@property (readonly, strong, nonatomic) NSPersistentStoreCoordinator \*persistentStoreCoordinator;

- (void)saveContext;

- (NSURL \*)applicationDocumentsDirectory;

@end

**Now go to your CoreAppDelegate.m and alter the ViewDidLoadWithOptions method to the following:**

- (BOOL)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions

{

self.window = [[UIWindow alloc] initWithFrame:[[UIScreen mainScreen] bounds]];

// Override point for customization after application launch.

\_viewController = [[CoreDataViewController alloc]

initWithNibName:@"CoreDataViewController" bundle:nil];

self.window.backgroundColor = [UIColor whiteColor];

[self.window setRootViewController:\_viewController];

[self.window makeKeyAndVisible];

return YES;

}

**Go to the SQLLite Demo Xib file. Shift + Click all elements on the interface until they are selected.**

**Copy those elements with Command + C and paste them into your new interface in the Core Data project.**

**Now create the same outlets and connections as before.**

**Remember, the actions are applied to the Save and Find buttons and the outlets to the text fields and labels.**

**Remember to synthesize your properties**

**The new saveData method is as follows:**

- (IBAction)saveData:(id)sender {

CoreDataAppDelegate \*appDelegate =

[[UIApplication sharedApplication] delegate];

NSManagedObjectContext \*context =

[appDelegate managedObjectContext];

NSManagedObject \*newContact;

newContact = [NSEntityDescription

insertNewObjectForEntityForName:@"Contacts"

inManagedObjectContext:context];

[newContact setValue: \_name.text forKey:@"name"];

[newContact setValue: \_address.text forKey:@"address"];

[newContact setValue: \_phone.text forKey:@"phone"];

\_name.text = @"";

\_address.text = @"";

\_phone.text = @"";

NSError \*error;

[context save:&error];

\_status.text = @"Contact saved";

}

**The new findData method is as follows:**- (IBAction)findContact:(id)sender {

CoreDataAppDelegate \*appDelegate =

[[UIApplication sharedApplication] delegate];

NSManagedObjectContext \*context =

[appDelegate managedObjectContext];

NSEntityDescription \*entityDesc =

[NSEntityDescription entityForName:@"Contacts"

inManagedObjectContext:context];

NSFetchRequest \*request = [[NSFetchRequest alloc] init];

[request setEntity:entityDesc];

NSPredicate \*pred =

[NSPredicate predicateWithFormat:@"(name = %@)",

\_name.text];

[request setPredicate:pred];

NSManagedObject \*matches = nil;

NSError \*error;

NSArray \*objects = [context executeFetchRequest:request

error:&error];

if ([objects count] == 0) {

\_status.text = @"No matches";

} else {

matches = objects[0];

\_address.text = [matches valueForKey:@"address"];

\_phone.text = [matches valueForKey:@"phone"];

\_status.text = [NSString stringWithFormat:

@"%d matches found", [objects count]];

}

}