**This will be a single view application that stores persistent data. This may accommodate some of your storage needs or be used in unison with other storage techniques that will be presented**

**These will be our steps**

1. Prepare as always
2. Design the Interface
3. Create the database in the ViewDidLoad method
4. Save data to the database
5. Find data and populate our fields with data from the databases

We will go through the finished version first.

**Part 1: Prep**

**Create a new iOS iPhone Single View Application project.**

**Give it a name like sqlite demo**

**Check Storyboard and Automatic Reference Counting**

**Add the SQLite dynamic library (libsqlite3.dylib) by selecting the project name in the left side of the project navigator.   
  
Then select “Build Phases”**

**Then look for the Link Binary with Library section.**

**Click on the ‘+’ button to display the full list. From this list search for, and then select libsqlite3.dylib and click Add.**

**Now import the sqlite header file in the ViewController.h file:**

**Also be sure to add an NSString object that can be used to store the path to the database. Your .h file should look like the following:**

#import <UIKit/UIKit.h>

#import <sqlite3.h>

@interface ViewController : UIViewController

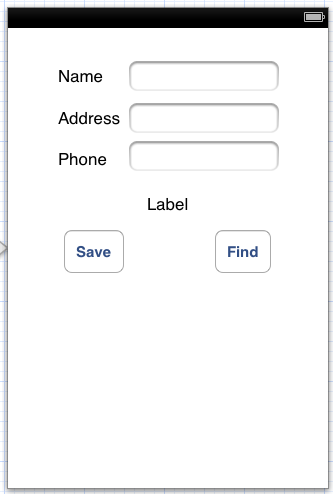
@property (strong, nonatomic) NSString \*databasePath;

@property (nonatomic) sqlite3 \*contactDB;

@end

**Part 2: Design**

**Create the following:**

****

**Create your quick connections for the three text fields, the two buttons and the label.**

**The three text fields will be outlets…you can name them name, address and phone**

**The label will also be an outlet, you can name that status**

**The Save Button should create an action named saveData**

**The find button should create an action named findContact**

**At this point, you ViewController.h should look like the following:**

#import <UIKit/UIKit.h>

#import <sqlite3.h>

@interface ViewController : UIViewController

@property (strong, nonatomic) IBOutlet UITextField \*name;

@property (strong, nonatomic) IBOutlet UITextField \*address;

@property (strong, nonatomic) IBOutlet UITextField \*phone;

@property (strong, nonatomic) IBOutlet UILabel \*status;

- (IBAction)saveData:(id)sender;

- (IBAction)findContact:(id)sender;

@property (strong, nonatomic) NSString \*databasePath;

@property (nonatomic) sqlite3 \*contactDB;

@end

**Part 3. The Database**

**When the application launches, it will need to check for the database. Remember the databasePath? If that database, does not exist, we need to create it in our code, so in ViewController.m, look for the ViewDidLoad method…**

- (void)viewDidLoad {

[super viewDidLoad];

NSString \*docsDir;

NSArray \*dirPaths;

**// Get the documents directory**

dirPaths = NSSearchPathForDirectoriesInDomains(

NSDocumentDirectory, NSUserDomainMask, YES);

docsDir = dirPaths[0];

**//Identifies the application’s Documents directory and constructs a path to the contacts.db database file**

\_databasePath = [[NSString alloc] initWithString: [docsDir stringByAppendingPathComponent: @"contacts.db"]];

**//Creates an NSFileManager instance and subsequently uses it to detect if the database file already exists.**

NSFileManager \*filemgr = [NSFileManager defaultManager];

if ([filemgr fileExistsAtPath: \_databasePath ] == NO)

{

const char \*dbpath = [\_databasePath UTF8String];

**//If the file does not yet exist the code converts the path to a UTF-8 string and creates the database via a call to the SQLite sqlite3\_open() function, passing through a reference to the contactDB variable declared previously in the interface file (ViewController.h).**

if (sqlite3\_open(dbpath, &\_contactDB) == SQLITE\_OK)

{

char \*errMsg;

**//Prepares a SQL statement to create the contacts table in the database**

const char \*sql\_stmt =

"CREATE TABLE IF NOT EXISTS CONTACTS (ID INTEGER PRIMARY KEY AUTOINCREMENT, NAME TEXT, ADDRESS TEXT, PHONE TEXT)";

**//Reports the success or otherwise of the operation via the status label.**

if (sqlite3\_exec(\_contactDB, sql\_stmt, NULL, NULL, &errMsg) != SQLITE\_OK)

{

\_status.text = @"Failed to create table";

}

**//Closes the db**

sqlite3\_close(\_contactDB);

} else {

\_status.text = @"Failed to open/create database";

}

}

}

**Part 4. Saving our contact…**

**We’ll need open our Contacts.db and access the values of the text fields and put them in an sql insert statement. Then we will close the db.**

**This will be our saveData method in the ViewController.m**

- (void) saveData:(id)sender

{

**//Initialize a statement variable…note the type**

sqlite3\_stmt \*statement;

**//redeclare the dbpath**

const char \*dbpath = [\_databasePath UTF8String];

if (sqlite3\_open(dbpath, &\_contactDB) == SQLITE\_OK)

{

**//if all is well, prepare the statement…they have backslashes for escapes…**

NSString \*insertSQL = [NSString stringWithFormat:

@"INSERT INTO CONTACTS (name, address, phone) VALUES (\"%@\", \"%@\", \"%@\")",self.name.text, self.address.text, self.phone.text];

const char \*insert\_stmt = [insertSQL UTF8String];

**//this will run the query**

sqlite3\_prepare\_v2(\_contactDB, insert\_stmt,

-1, &statement, NULL);

**//this will test the query’s success**

if (sqlite3\_step(statement) == SQLITE\_DONE)

{

**//added confirmation to the label**

self.status.text = @"Contact added";

self.name.text = @"";

self.address.text = @"";

self.phone.text = @"";

} else {

self.status.text = @"Failed to add contact";

}

sqlite3\_finalize(statement);

sqlite3\_close(\_contactDB);

}

}

**Part 6: Accessing Stored Content**

**Remember, the findContact method is triggered by clicking the Find button. We’ll to prepare a query from the text that was entered in the name field**

- (void) findContact:(id)sender

{

**//same as in the saveData method**

const char \*dbpath = [\_databasePath UTF8String];

sqlite3\_stmt \*statement;

if (sqlite3\_open(dbpath, &\_contactDB) == SQLITE\_OK)

{

**//here’s out prepared statement**

NSString \*querySQL = [NSString stringWithFormat:

@"SELECT address, phone FROM contacts WHERE name=\"%@\"",

\_name.text];

const char \*query\_stmt = [querySQL UTF8String];

if (sqlite3\_prepare\_v2(\_contactDB,

query\_stmt, -1, &statement, NULL) == SQLITE\_OK)

{

if (sqlite3\_step(statement) == SQLITE\_ROW)

{

**//if there is a row with that name match**

NSString \*addressField = [[NSString alloc]

initWithUTF8String:

(const char \*) sqlite3\_column\_text(

statement, 0)];

**//make the address text equal to the addressField in the DB for that one record**

\_address.text = addressField;

NSString \*phoneField = [[NSString alloc]

initWithUTF8String:(const char \*)

sqlite3\_column\_text(statement, 1)];

**//make the phone text equal to the phoneField in the DB for that one record**

\_phone.text = phoneField;

\_status.text = @"Match found";

} else {

\_status.text = @"Match not found";

\_address.text = @"";

\_phone.text = @"";

}

sqlite3\_finalize(statement);

}

sqlite3\_close(\_contactDB);

}

}

**Run that sucker and you should be done…**