

<http://github.com/GroundSpeed/ParseSwift>

Locking Out the Abominable Snowman in iOS

A CodeMash 2015 Introduction to parse.com using Swift



Don Miller

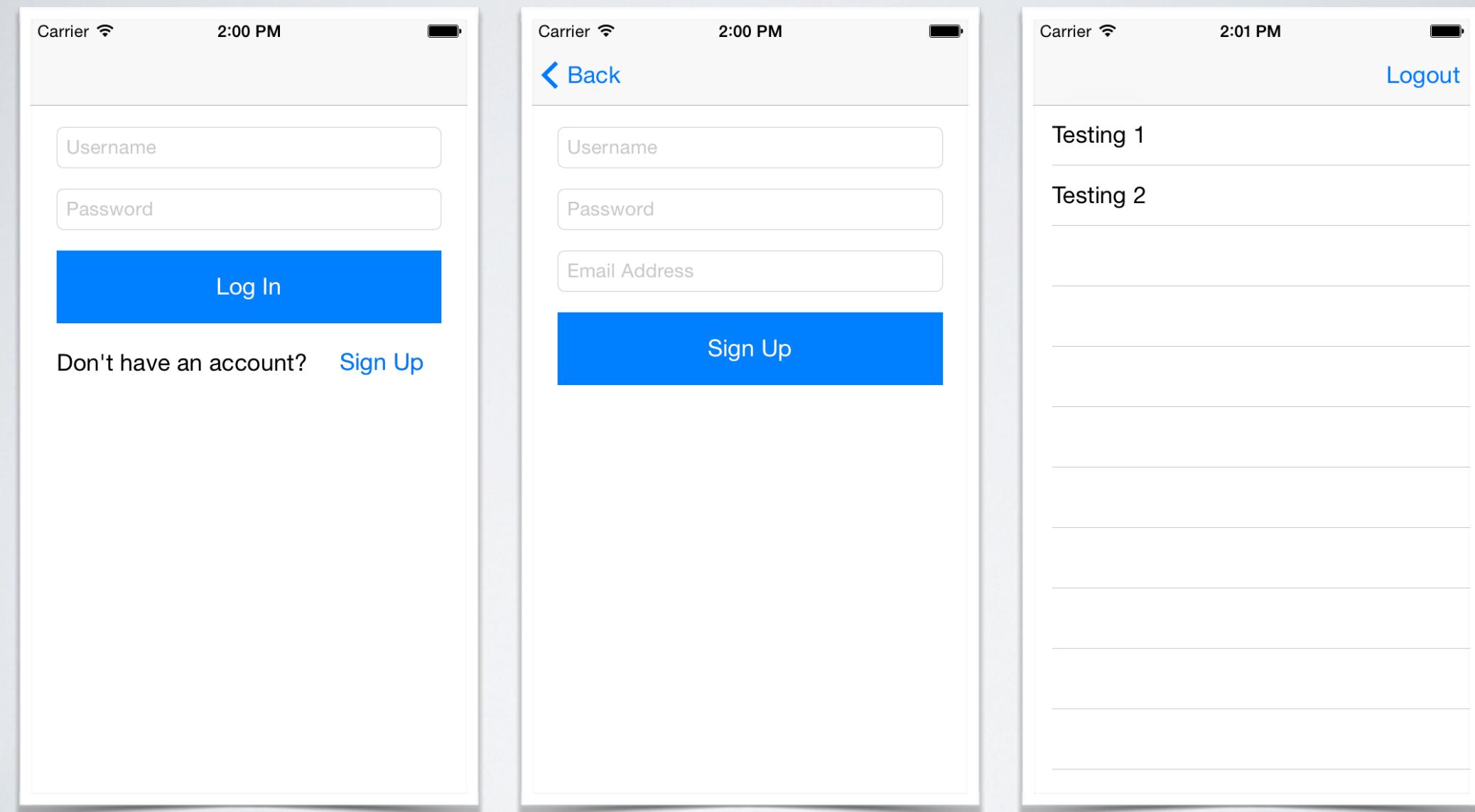
GroundSpeed™
rapid web + mobile software

A bit about Parse

- “Cloud” service to store various pieces of data and notifications
 - Users for Security
 - Classes for data tables (Core)
 - We can pull, push, update, or delete records
 - Analytics
 - Push Notifications



Let's Create a Simple App to Authenticate to Parse



Create Parse App

R

Add a Column ×

String ▼

name

Must only contain alphanumeric or underscore characters, and must begin with a letter or number.

Create Column Cancel

ay



GroundSpeed™
rapid web + mobile software

Create A New Single View Application

Choose a template for your new project:

iOS	 Master-Detail Application	 Page-Based Application	 Single View Application	 Tabbed Application
OS X	 Game			
	<p>Single View Application</p> <p>This template provides a starting point for an application that uses a single view. It provides a view controller to manage the view, and a storyboard or nib file that contains the view.</p>			

Cancel **Previous** **Next**



GroundSpeed™
rapid web + mobile software

Enter the Product Name

Choose options for your new project:

Your company's name

Product Name: ParseMash

Organization Name: GroundSpeed

Organization Identifier: com.groundspeedhq

Bundle Identifier: com.groundspeedhq.ParseMash

Language: Swift

Devices: iPhone

Use Core Data

Cancel

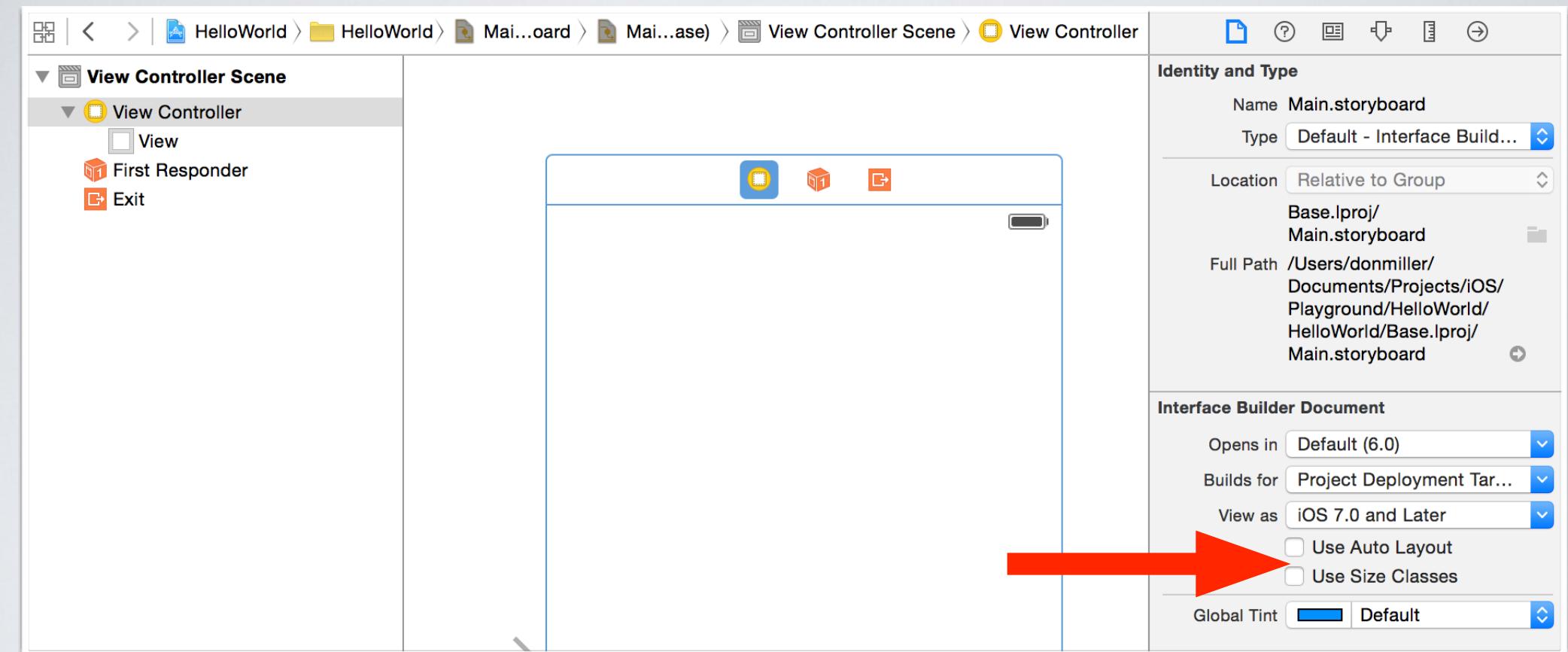
Previous

Next

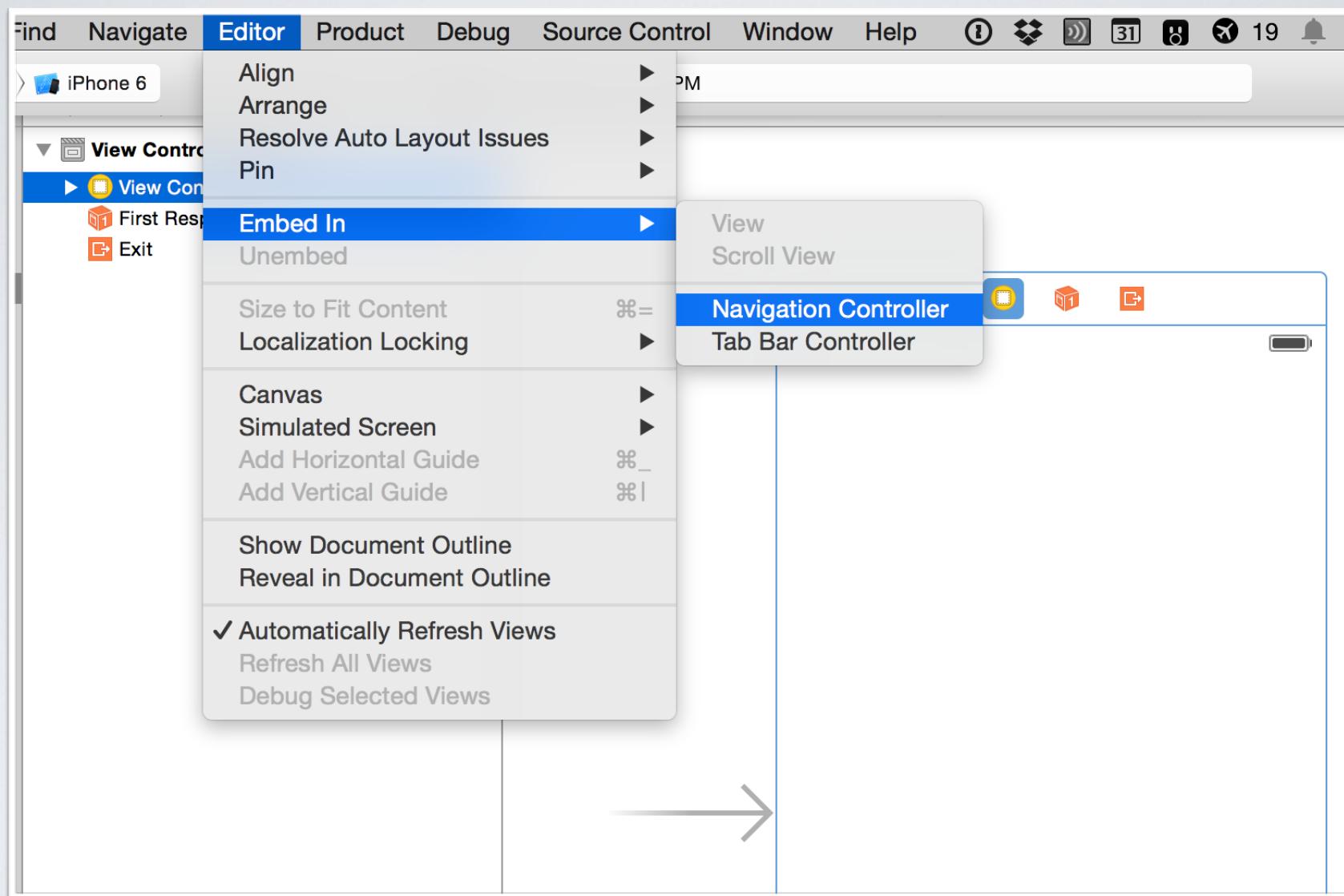


GroundSpeed™
rapid web + mobile software

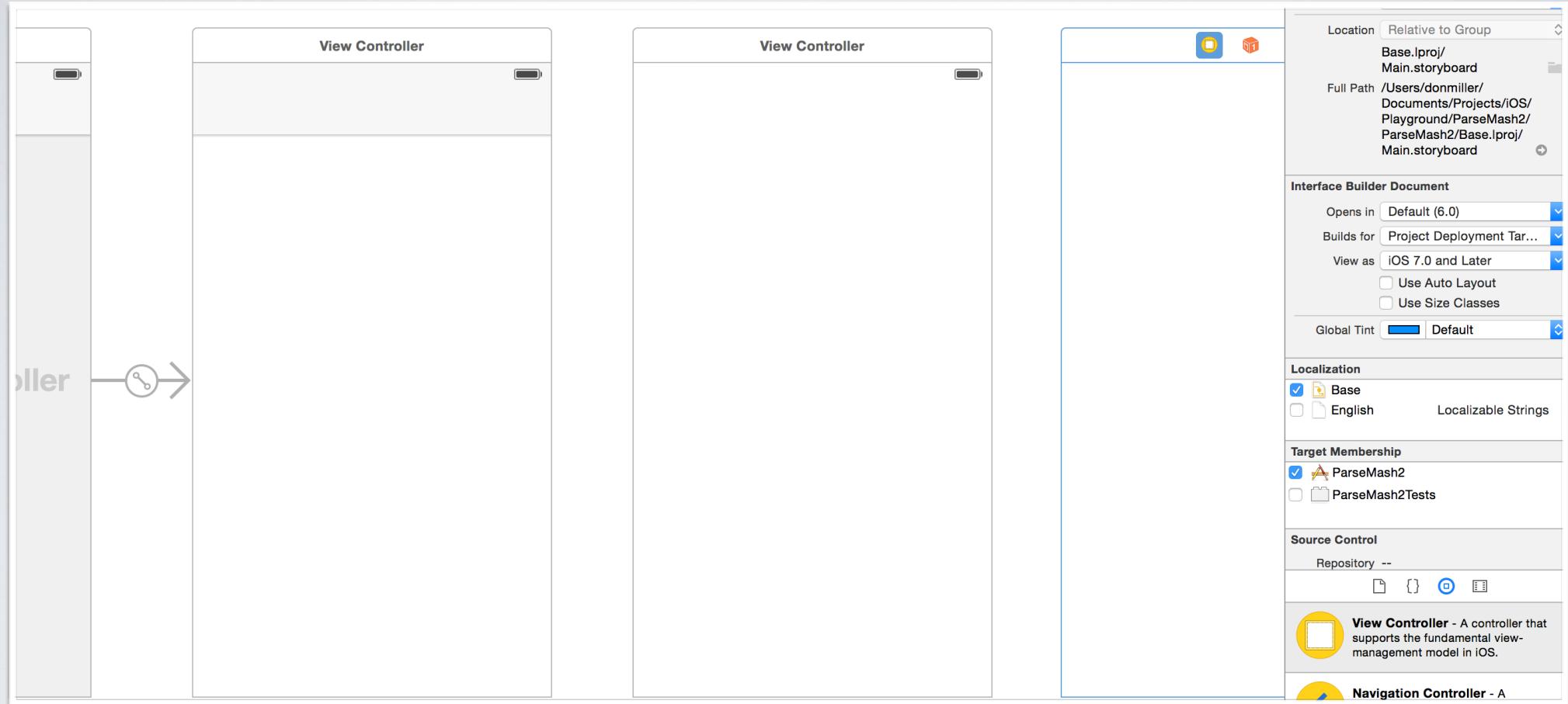
Turn Off Size Classes and Auto Layout



Embed Main View Inside Navigation



Create 2 Additional View Controllers



GroundSpeed™
rapid web + mobile software

Add TextFields and Buttons to the Login and Signup

Login



Username

Password

Login

Don't have an account? [Sign Up](#)

SignUp



Username

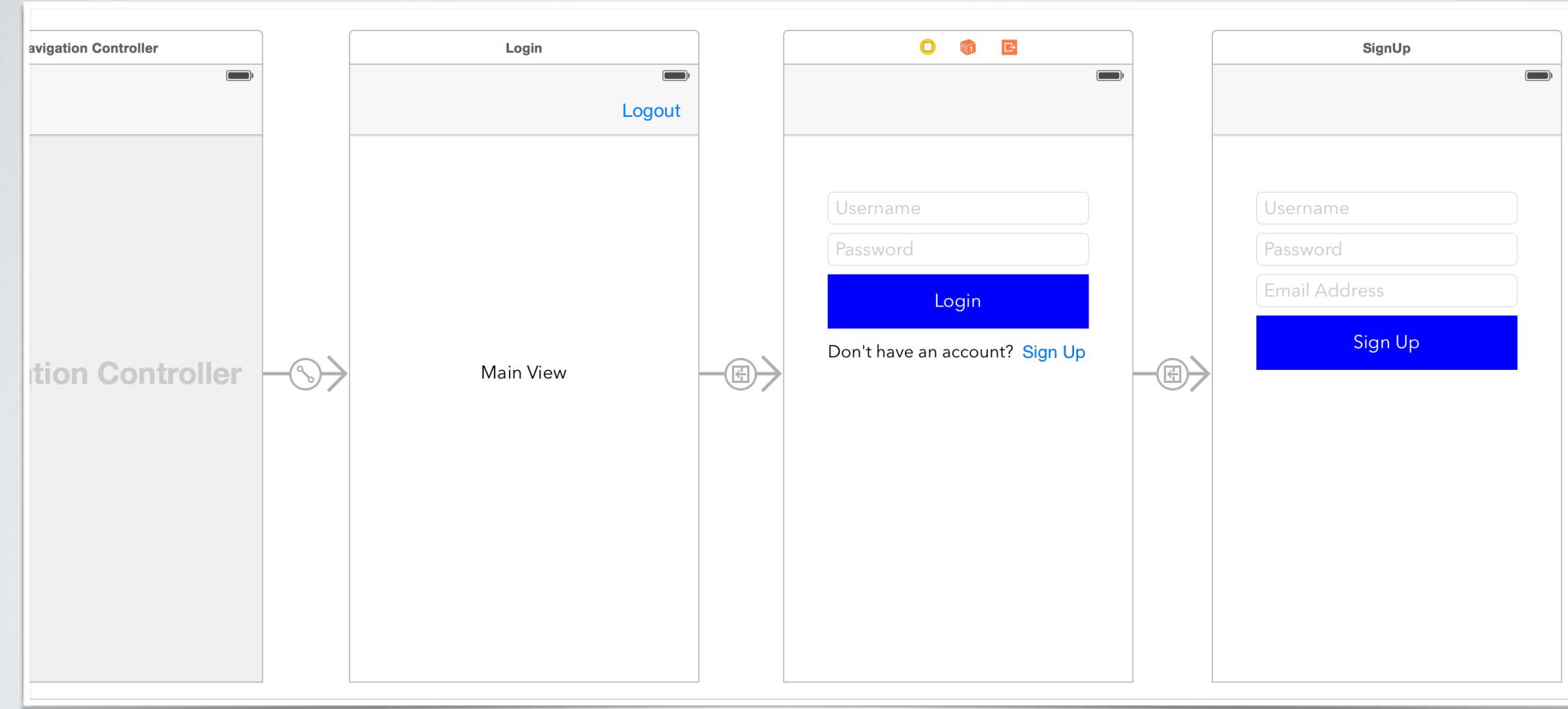
Password

Email Address

Sign Up

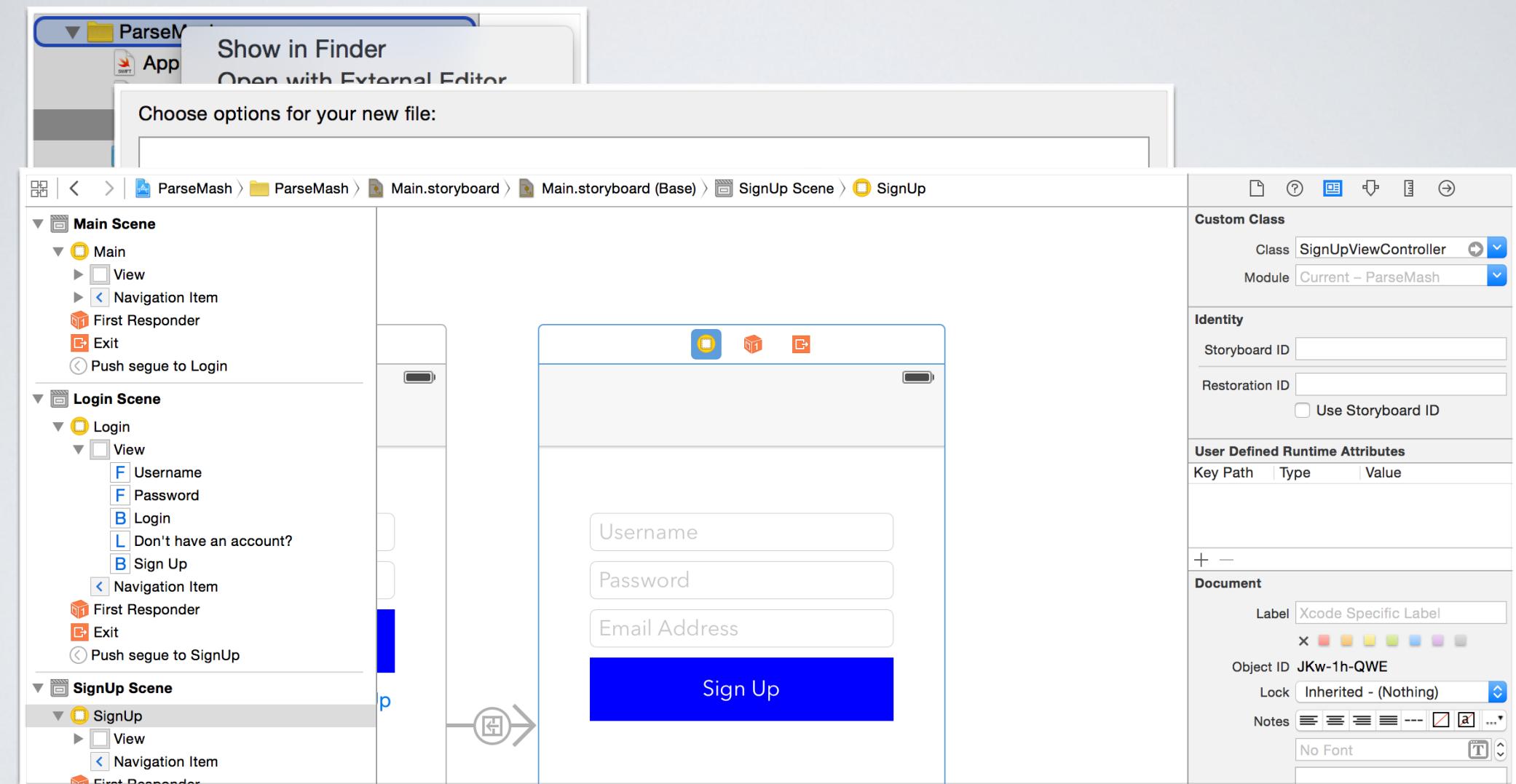


Add Logout Bar Button and Push Segues



GroundSpeed™
rapid web + mobile software

Create Login and Signup View Controllers



GroundSpeed™
rapid web + mobile software

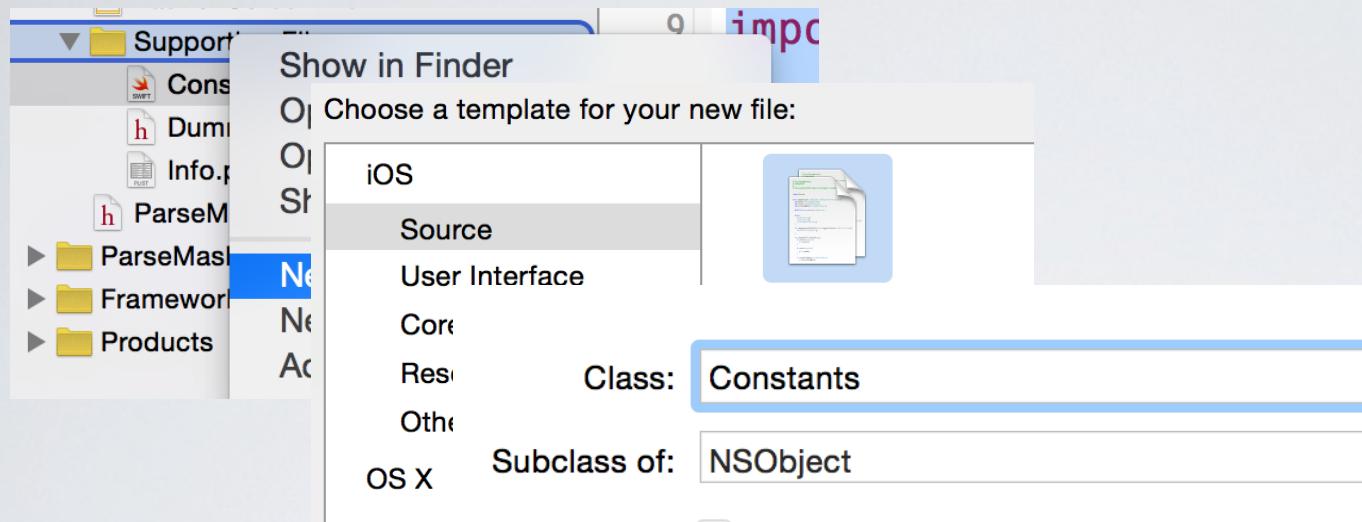
Add the Needed Parse Framework and Prerequisites

The red are required by Parse, but the screenshot is required for our needs

- `AudioToolbox.framework`
- `CFNetwork.framework`
- `CoreGraphics.framework`
- `CoreLocation.framework`
- `libz.dylib`
- `sqlite3.dylib`
- `MobileCoreServices.framework`
- `QuartzCore.framework`
- `Security.framework`
- `StoreKit.framework`
- `SystemConfiguration.framework`
- `Bolts.framework`
- `Parse.framework` (obviously)



Create a Constants Class



The screenshot shows the Xcode interface with a modal dialog for creating a new file. The dialog is titled "Choose a template for your new file:" and lists "iOS" and "OS X" categories. Under iOS, "Source" is selected, and the "Class" field contains "Constants". The "Subclass of" field is set to "NSObject". The background shows a project structure with files like "Support", "Constants.h", "Constants.m", "Info.plist", and "ParseM...".

CodeMash 2015 ▾

General Keys Push Hosting Authentication Email

Application Keys

Application ID: 7gHOA37vdCLhuMDBTLxFGv1DG6vD27MncHwQQAle

Client Key: Zgi6DNP7Ufas7biZIJqPA3hFtHE31XT2931HPL9a



Add Parse to the App Delegate

```
import Parse

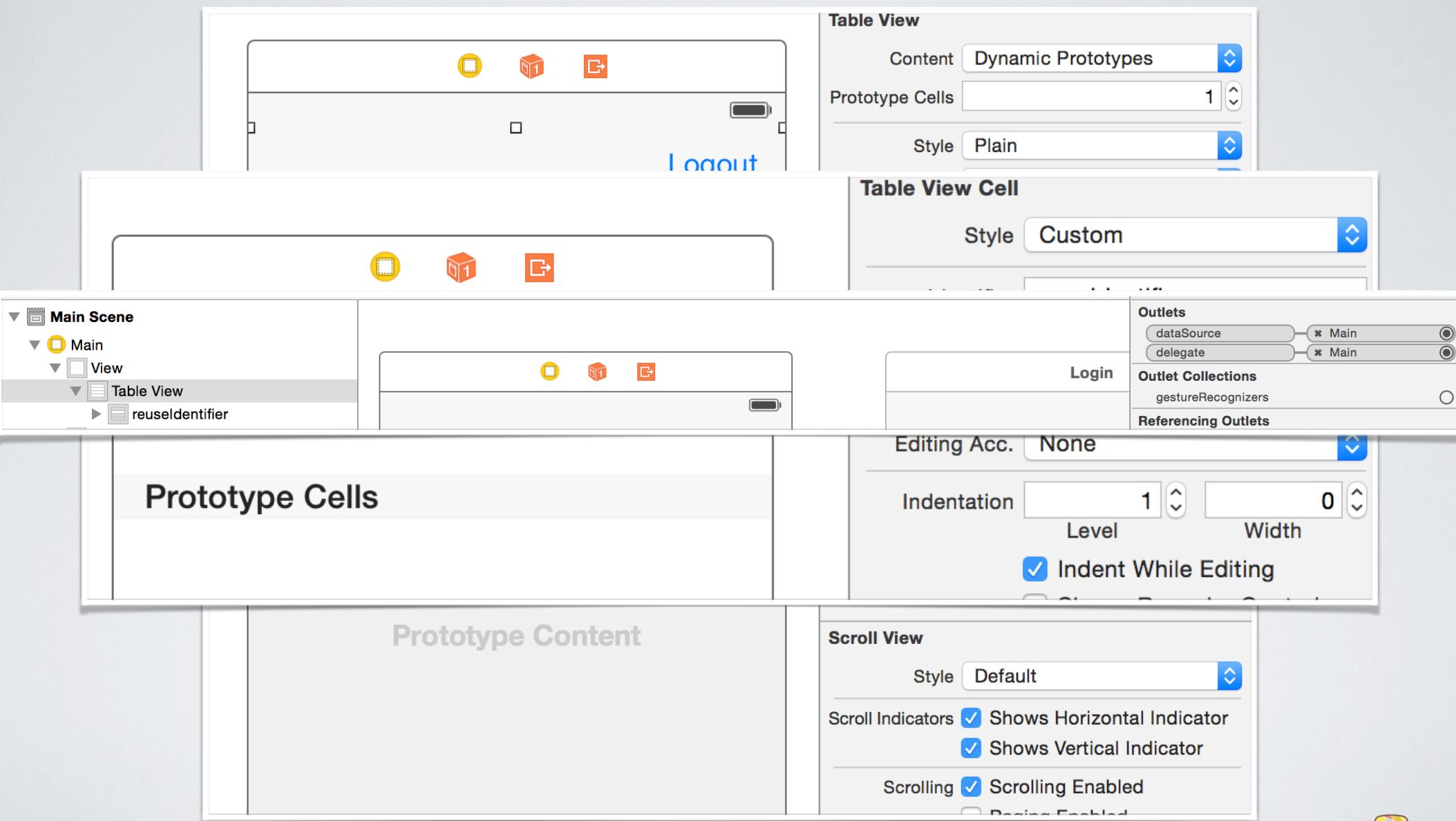
@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

    var window: UIWindow?
    let c = Constants()

    func application(application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [NSObject: AnyObject]?) -> Bool {
        // Override point for customization after application launch.
        Parse.setApplicationId(c.kApplicationId, clientKey: c.kClientKey)
        return true
    }
}
```



Add a TableView to Your Main Controller



Program Your Main ViewController

```
override func viewDidAppear(animated: Bool) {
    tblWhiskey.reloadData()
}

func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    return self.whiskeys.count
}

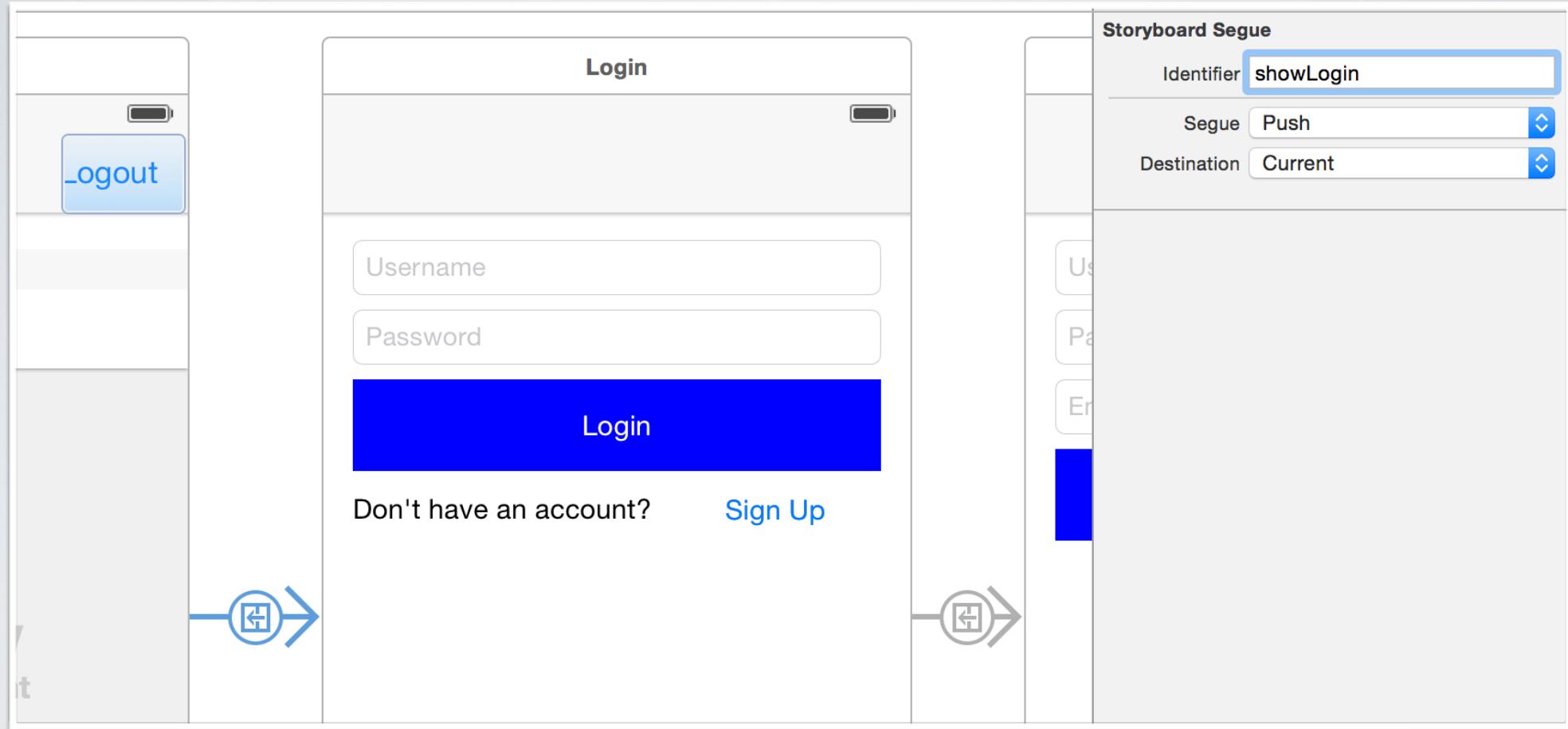
func tableView(tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    let cell = tableView.dequeueReusableCell(withIdentifier: "reuseIdentifier",
forIndexPath: indexPath) as UITableViewCell

    // Configure the cell...
    cell.textLabel?.text = self.whiskeys[indexPath.row]

    return cell
}

@IBAction func btnLogout(sender: AnyObject) {
    PFUser.logOut()
    PFUser.currentUser().save()
    self.performSegueWithIdentifier("showLogin", sender: self)
}
```

Update Identifier in Login Segue



Add your IBAction and IBOulet Connections to the Main View

The screenshot shows the Xcode storyboard editor. On the left, a main view contains three icons (blue square with white square, orange cube, red square with white square) and a 'Logout' button. Below the main view is a section labeled 'Prototype Cells'. On the right, a sidebar displays the connection editor with the following sections:

- Triggered Segues**: manual
- Outlets**:
 - searchDisplayController
 - tblWhiskey → * Table View
 - view → * View
- Presenting Segues**:
 - relationship → * Navigation Co... root view contr...
- embed
- push
- modal
- custom
- Referencing Outlets**: New Referencing Outlet
- Referencing Outlet Collections**: New Referencing Outlet Collection
- Received Actions**: btnLogout: → * Logout



Program Your LoginViewController

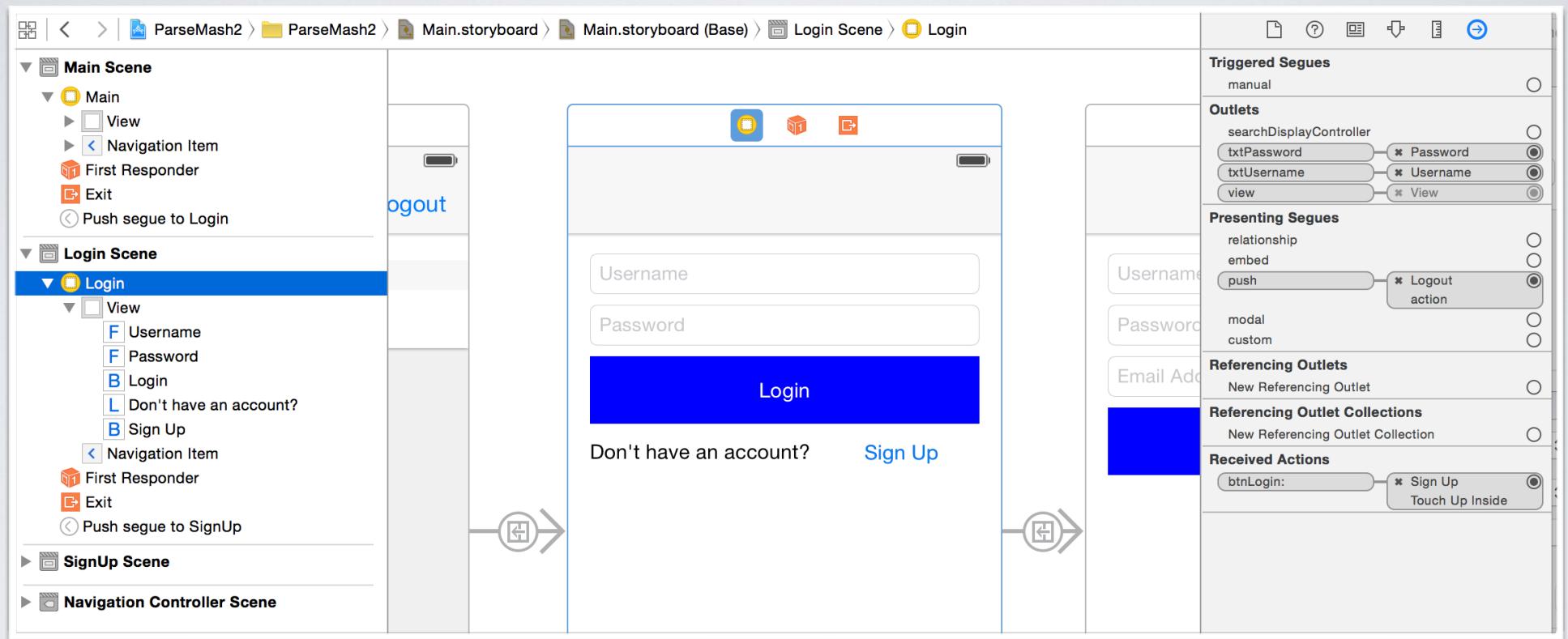
```
@IBOutlet var txtUsername: UITextField!
@IBOutlet var txtPassword: UITextField!

override func viewDidLoad() {
    super.viewDidLoad()
    self.navigationItem.hidesBackButton = true
}

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

@IBAction func btnLogin(sender: AnyObject) {
    PFUser.logInWithUsernameInBackground(txtUsername.text, password:txtPassword.text)
{
    (user: PFUser!, error: NSError!) -> Void in
    if user != nil {
        self.navigationController?.popToRootViewControllerAnimated(true)
    } else {
        let alert = UIAlertView()
        alert.title = "Sorry!"
        alert.message = error.userInfo![ "error" ] as NSString
        alert.addButtonWithTitle("OK")
        alert.show()
    }
}
}
```

Add your IBAction and IBOulet Connections to the Login View

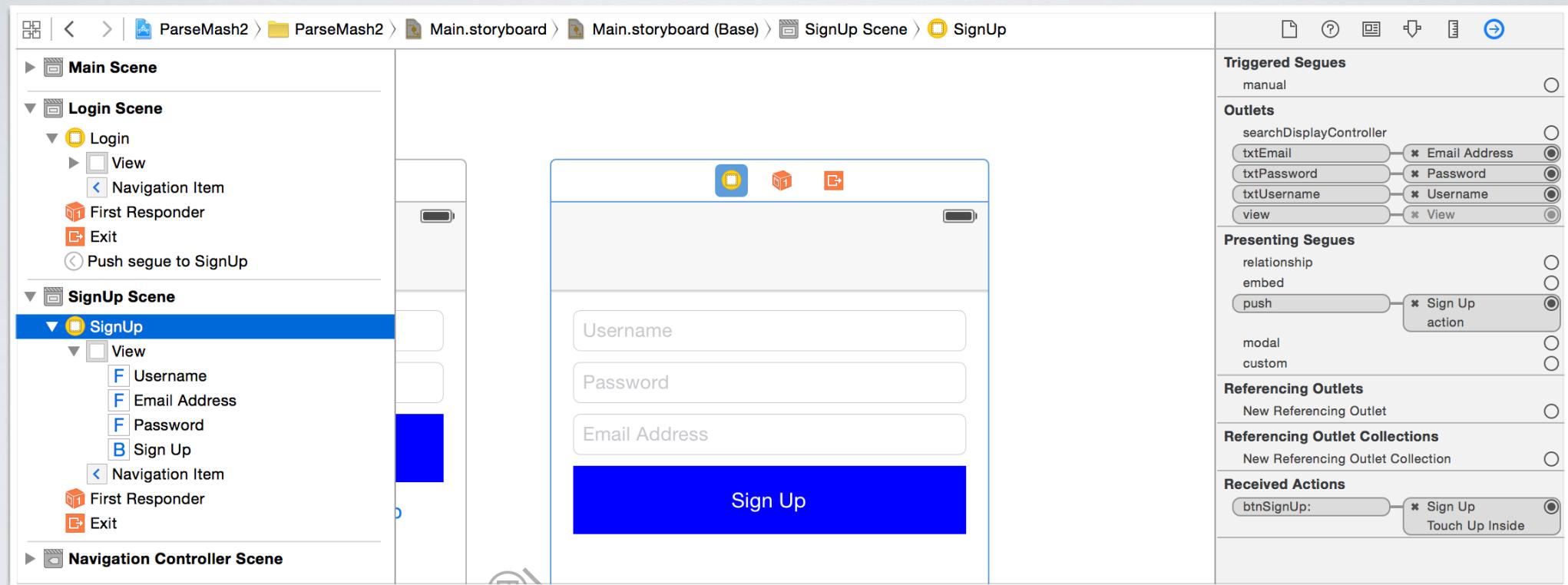


Program Your SignUpViewController

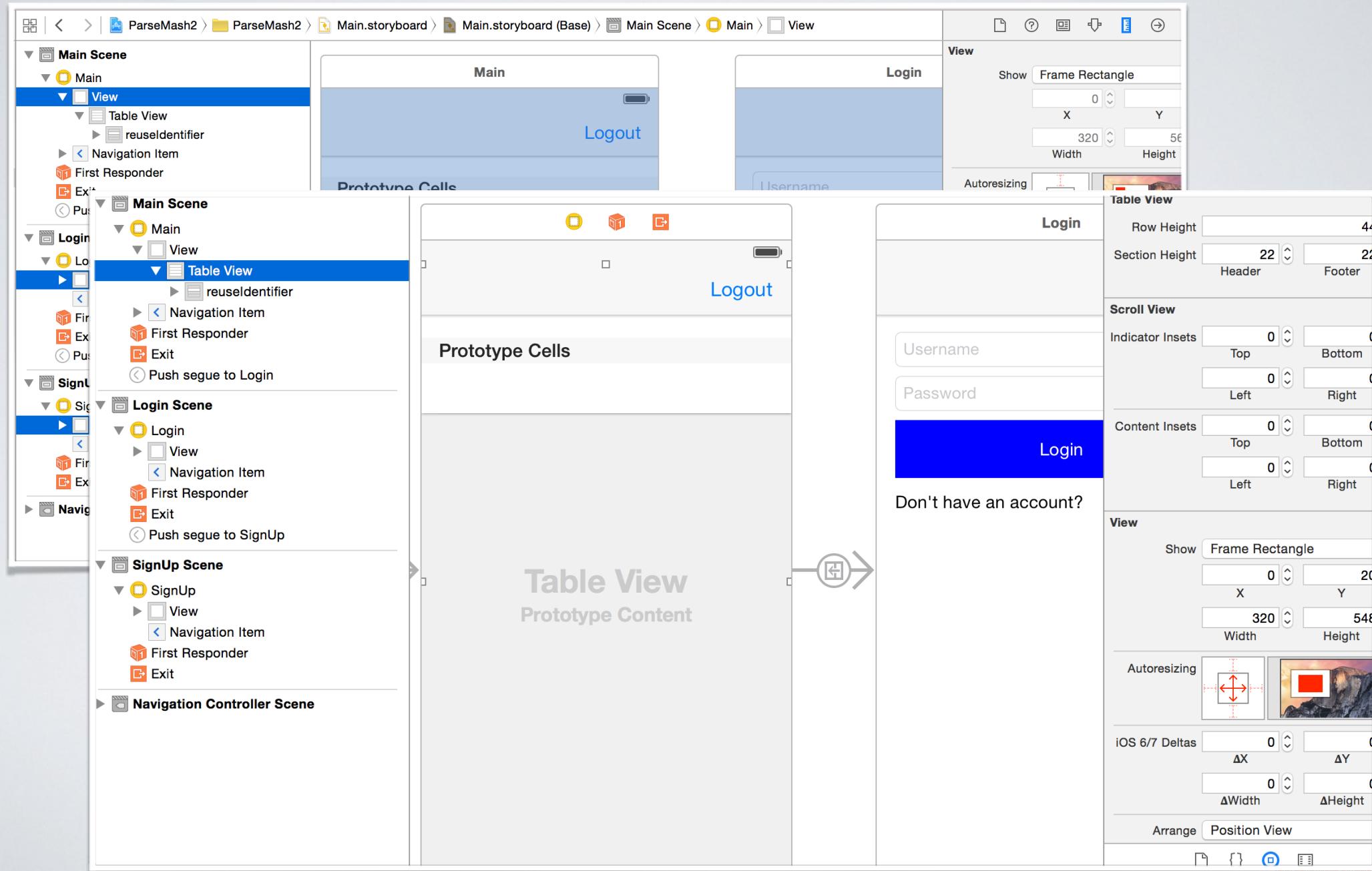
```
if (countElements(username) == 0 || countElements(password) == 0 || countElements(email) == 0)
{
    let alert = UIAlertView()
    alert.title = "Oops!"
    alert.message = "Make sure you enter an username, password, and email address!"
    alert.addButtonWithTitle("OK")
    alert.show()
}
else
{
    var newUser = PFUser()
    newUser.username = username
    newUser.password = password
    newUser.email = email

    newUser.signUpInBackgroundWithBlock {
        (succeeded: Bool!, error: NSError!) -> Void in
        if error == nil {
            self.navigationController?.popToRootViewControllerAnimated(true)
        } else {
            let alert = UIAlertView()
            alert.title = "Sorry!"
            alert.message = error.userInfo![ "error"] as NSString
            alert.addButtonWithTitle("OK")
            alert.show()
        }
    }
}
```

Add your IBAction and IBOulet Connections to the SignUp View



Clean Up - Because We Aren't Using Auto Layout



Don Miller
don@groundspeedhq.com
[@donmiller](https://twitter.com/donmiller)

Or follow [@groundspeedhq](https://twitter.com/groundspeedhq)
[groundspeedhq.com](http://www.groundspeedhq.com)

