Tab Bar Hands on

1. Create new project using Single view template. Name it tabBar. Leave prefix blanks.

2. Notice that class files have no prefix and have general viewController as a name.
3. Double click ViewController.h file and highlight ViewController in .h file description next to @interface. Right click mouse with “ViewController” highlighted and click on Refactor->Rename. Pick Tab1ViewController as new name, you can click preview and see what changes would look like once happy with change you can save it. Files will be renamed to Tab1ViewController.h/.m

4. Next Add view Controller class for the second tab content view. Right click on group and add New File, pick UIViewController subclass, on the next screen give name for new class as Tab2ViewController and ensure that subclass is set to UIViewController, ensure that xib is not clicked. Click next to generate the class files.

5. Next step is to add Tab Bar controller to main. Select MainStoryBoard.storyboard file - which is the original view controller instance and hooked to NSapp. Highlight Tab1ViewController, click Editor on top Xcode bar menu line (apple line) -> Embed In -> Tab Bar Controller menu option. The Tab bar will appear in the story board connected to that tab Bar and first view.

6. Add second view controller to the story board by choosing View Controller object from the object library and dropping it anywhere on canvass. High light VC and view identity inspector. Change custom class to Tab2ViewController

7. Control Click on the tab bar controller window and drag it to Tab2ViewController. When you release mouse three options will be provided - pick Relationship - view Controllers. Now we should have Tab Bar Controller and two connected views: Tab1ViewController and Tab2ViewController

8. Change back ground colors of Tab1ViewController so we can visually see that view has changed when user selects a tab.

9. Some basic things to clean up - double click icon and change text. you can add picture for the tab change background etc.

Build and Run - this is our basic skeleton tabBar app. Next we will add image to the second tab.

10. Drag imac.jpg image from previous project and put in Resources group.

11. Change Tab2ViewController.h file as follows:

```
@interface Tab2ViewController : UIViewController {
    IBOutlet UIImageView *imageView2;

}
```

```
@property (nonatomic, retain) UIImageView *imageView2;
```

12. Change Tab2ViewController.m file - viewDidLoad method as follows:

```
    //---initialize the first imageview to display an image---
    [imageView2 setImage:[UIImage imageNamed:@"iMac.jpeg"]];


    //---make the first imageview visible and hide the second---
    [imageView2 setHidden:NO];
```

13. Select MainStoryBoard and goto object library, pick UIImageView and drop in Tab2ViewController view. Size it appropriately, you can overlap on the buttons and battery bar but generally a good idea not to do it. The bottom black bar labeled as “Tab2 View Controller” has first responder and controller click the bar. Control click from View Controller to UIImageView and select imageView2 as outlet.

14. B & R

15. Notice that your image is outside of bounds that you thought you had put. It’s wonders of “Auto Layout”. Highlight view and visit File Inspector. Un-check “Use Auto Layout”.

16. Build and Run and image should be within bounds that you had set.

17. Change Tab Label: Highlight tab bar in first tab view window and visit attributes inspector. You will see Title for Bar Item – Change it to My Colors. Highlight second tab in second view and visit attributes inspector. Notice that Badge/Identifier/ Title in Tab Bar Item section. Change identifier from Custom to Favorites – note that you cannot change text for non-custom bar items.

18. Build and Run and notice app.

Add text to Tab1ViewController.

19. Add UITextView on to Tab1ViewController view. Size up for tab bar and top. Change Tab1ViewController as follows:

```
@interface Tab1ViewController : UIViewController {
```

```
    IBOutlet UITextView *textView;
}
```

Update Tab1ViewController.m as follows:

```
- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view, typically
from a nib.

    NSString *filePath = [[ NSBundle mainBundle]
pathForResource:@"myText" ofType:@"rtf"];

    textView.text = [NSString stringWithContentsOfFile:filePath
encoding:NSUTF8StringEncoding error:nil];
}
```

20. Make connections in storyboard to connect outlet.

21. Build and Run

You can edit properties at run time for example

you can change the following:

```
[textView setEditable:NO];
[textView setScrollEnabled:NO];
[textView setFont:[UIFont systemFontOfSize:16]];
```

textAlignment, textColor, etc.