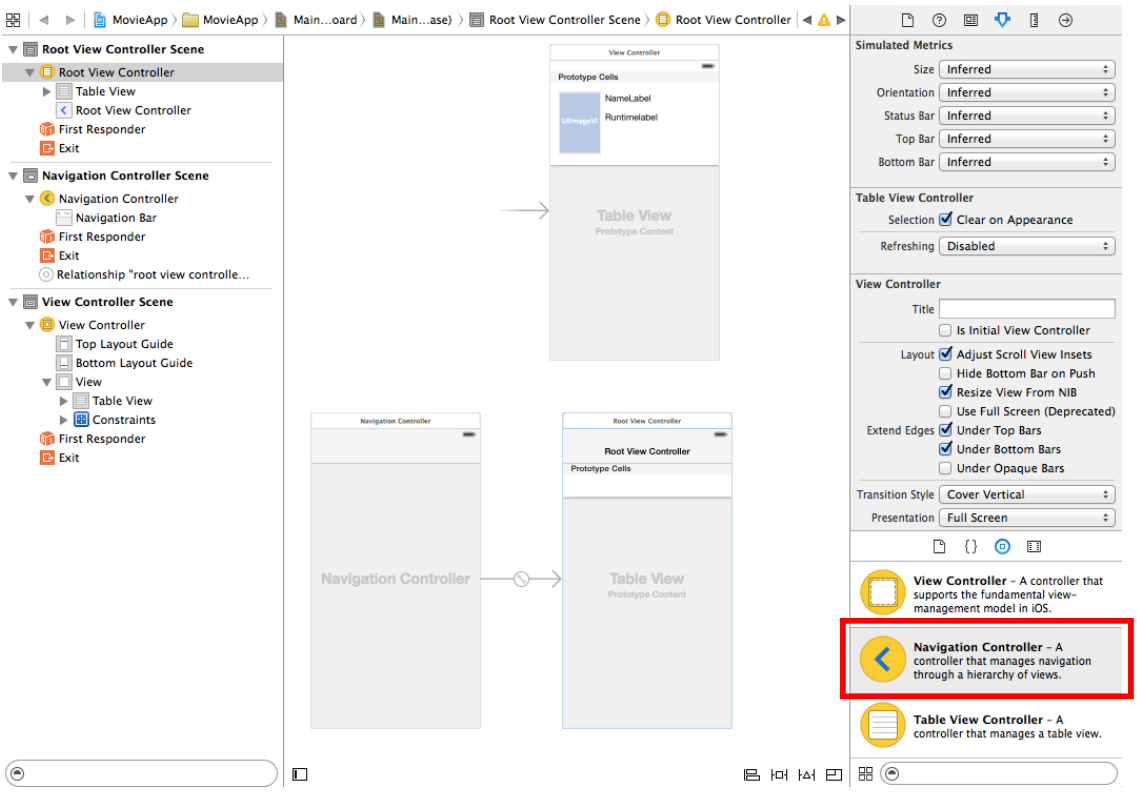
Practical 05: Working with UINavigationController

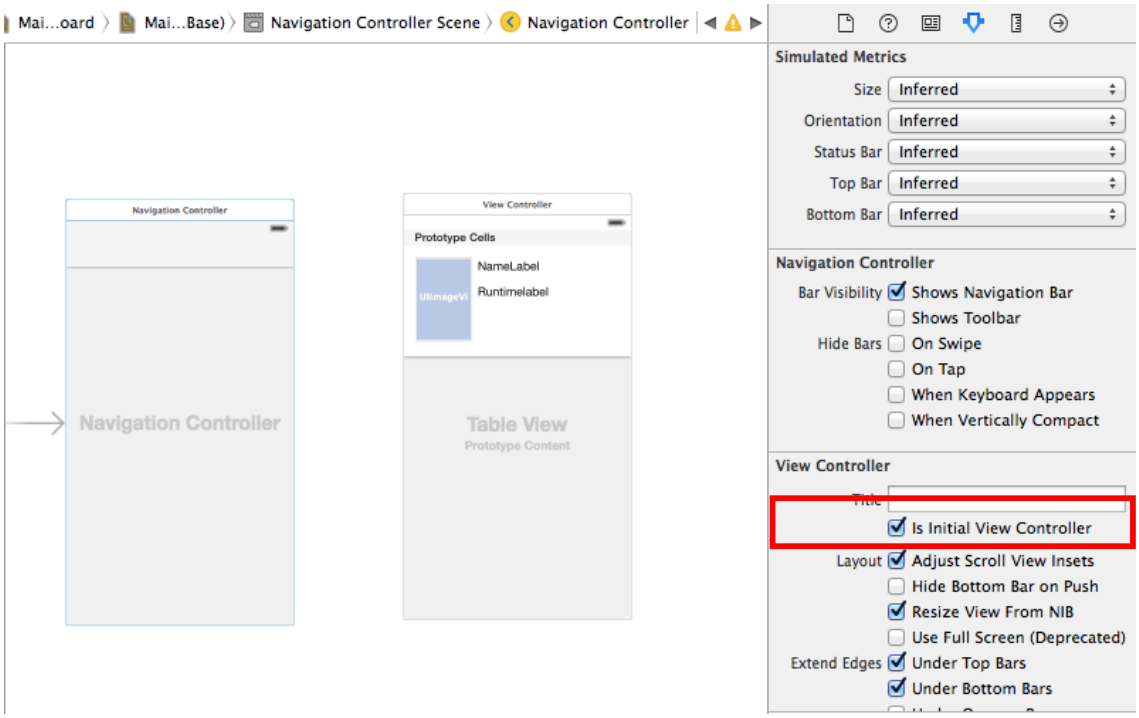
In this chapter, we will learn how to combine the table View with a navigation controller. We will make use of the application we created in the previous practical and add a navigation controller and allow user to view full details of the Movie selected.

Section 1: Add UINavigationController

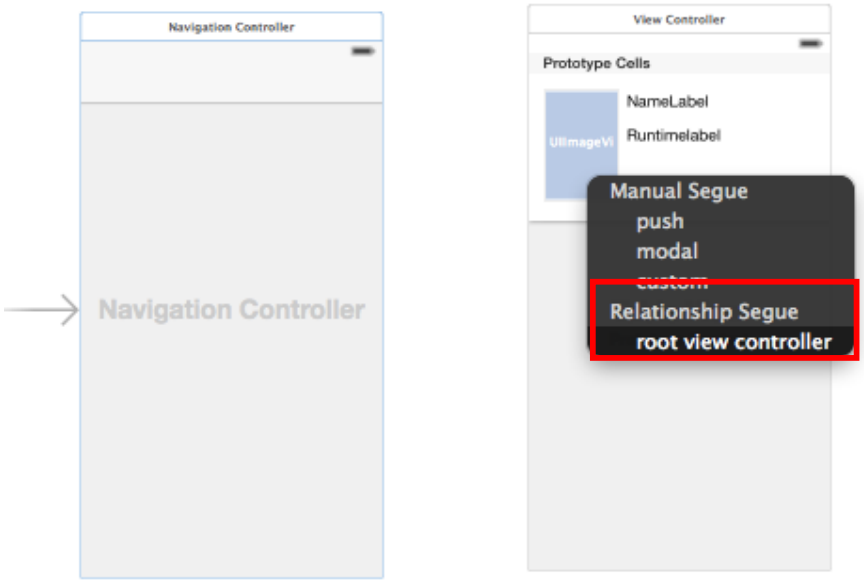
1. Open the **MovieApp** project created during your previous practical lesson. Click on the **Main.Storyboard.** In the Object Library, select the **Navigation Controller** object and drag it into the interface builder.



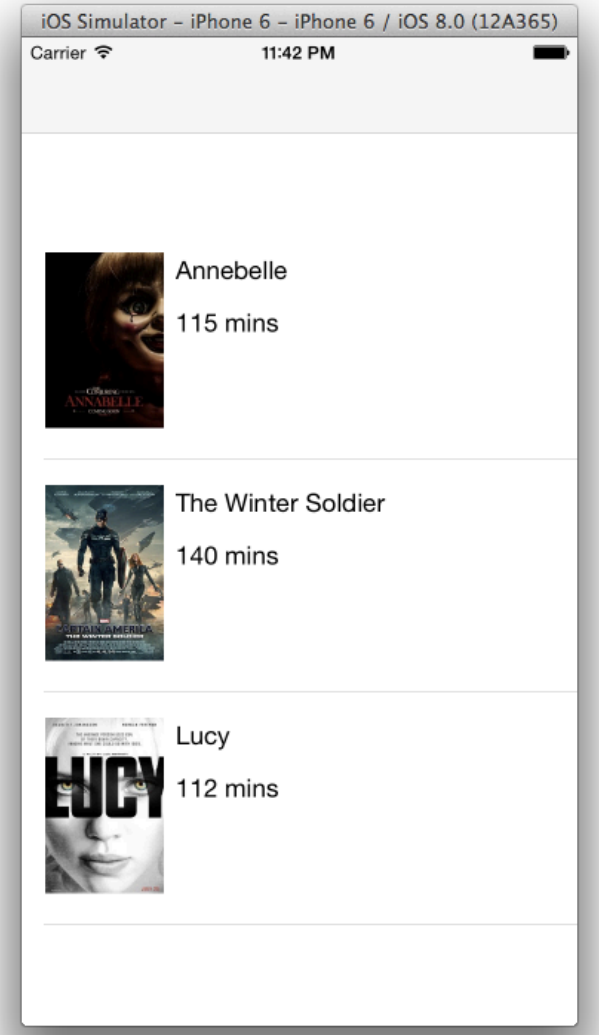
2. At the interface builder, select and delete the **Root View Controller**.
3. With the **Navigation Controller** selected, under the **Attribute inspector**, click on the **Is Initial View Controller** to set it as the first screen when the application is launched.



4. Ctrl+click and drag the **Navigation Controller** to connect to the **View Controller** that we did previously and select the Relationship Segue -> root view controller.

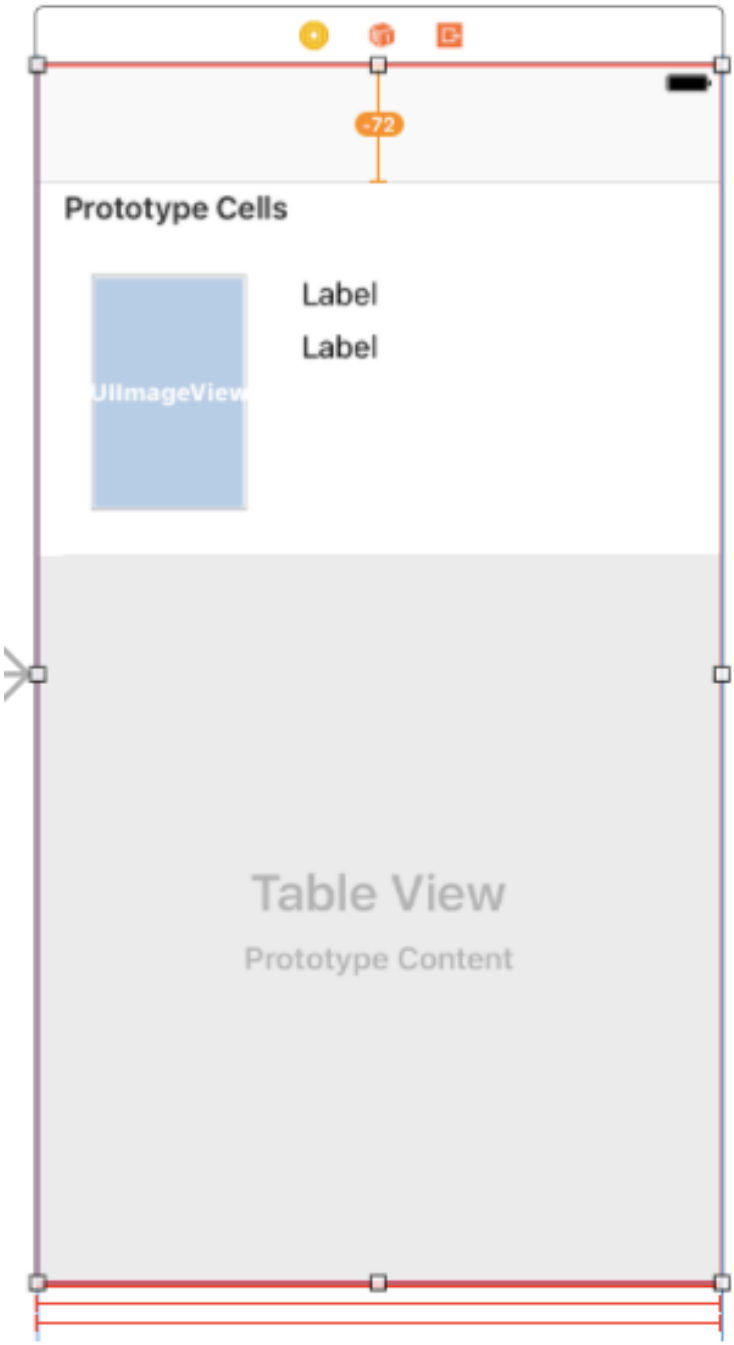


5. Build and run the application. Notice that the MovieApp has a navigation bar at the top of the screen.

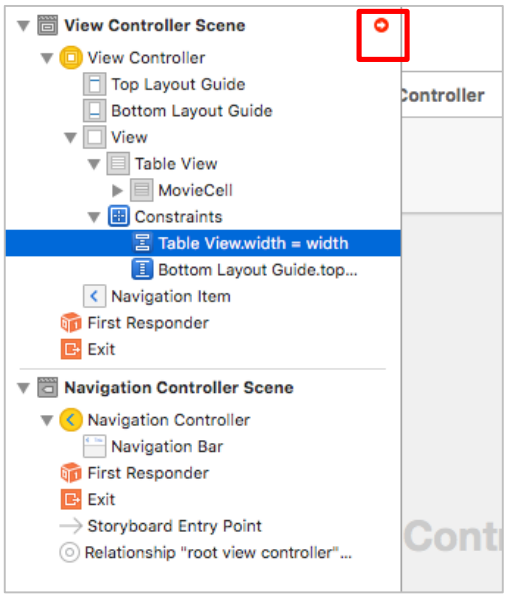


6. You will noticed that the table view has the extra white space on top of the table view. Due to the additional navigation item in the view, we need to adjust the height of the table view and change the constraints.

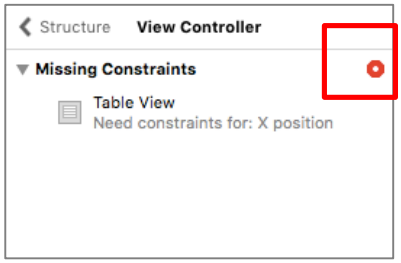
7. Click on the **Table View** and adjust the top edge of the Table View to the top of the View.



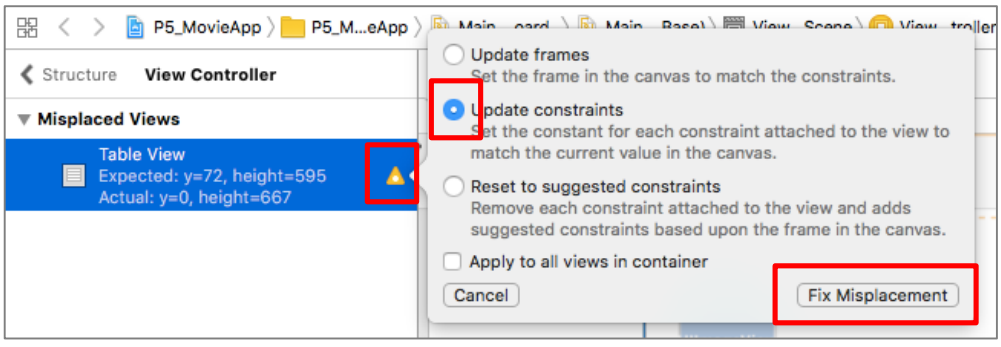
8. At the outline pane, you will notice a red icon. Click on the red icon.



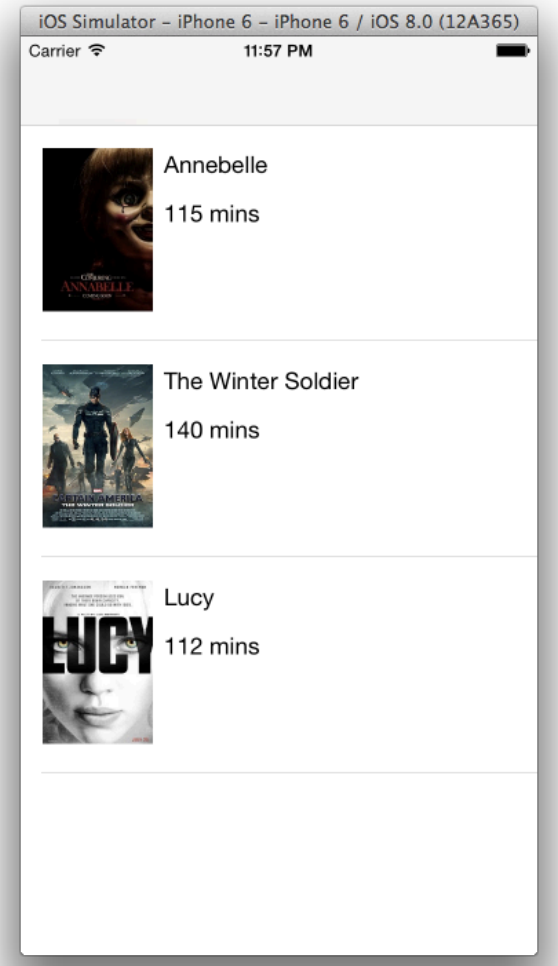
9. The outline pane will refresh to show some missing constraints errors. Click on the red icon and **Add Missing Constraints**.



10. After adding the missing constraints, you may still encounter a warning on the constraints. Click on the warning icon, select **Update Constraints**, then click **Fix Misplacement**.



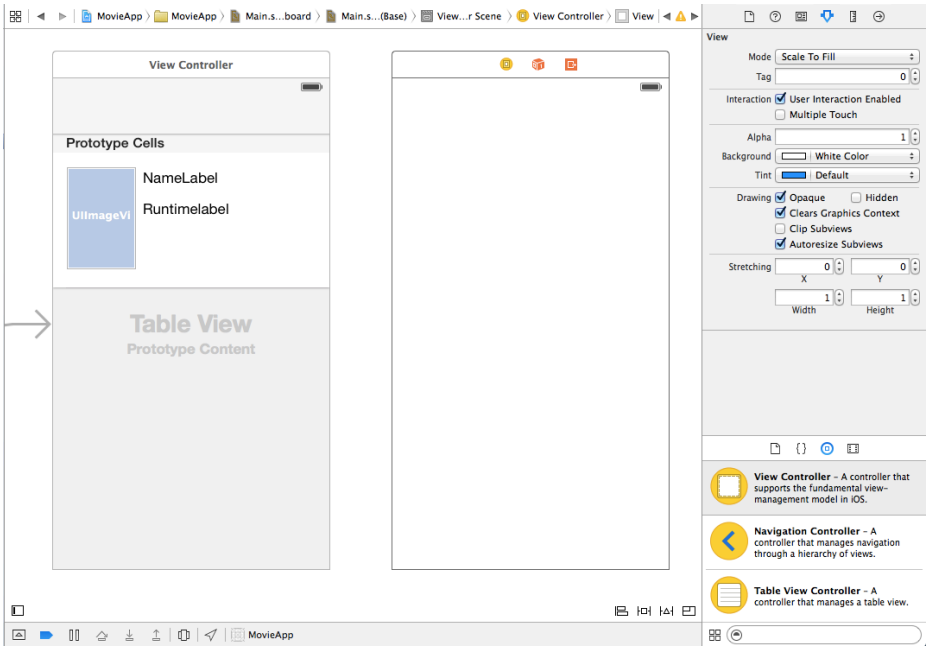
11. Build and run the application. Notice that the white space is removed.



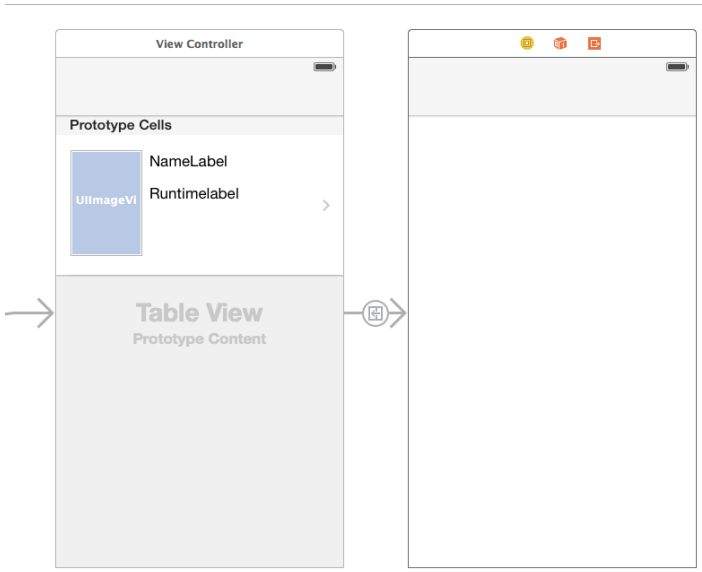
Section 2: Passing selected object value to another ViewController

In the section we will build a new screen that will show up when the user clicks on one of the movies in the Table View. This new screen will show a detailed description about the movie. In order to do that, we will have to detect which movie the user clicked, and pass that information to our new screen.

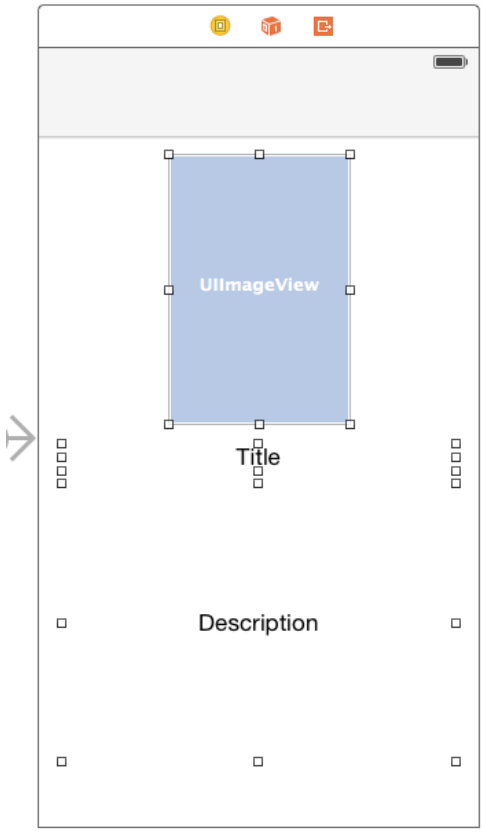
12. Click on the **Main.storyboard**. Under the Object Library, drag and drop the **View Controller** object onto the interface builder.



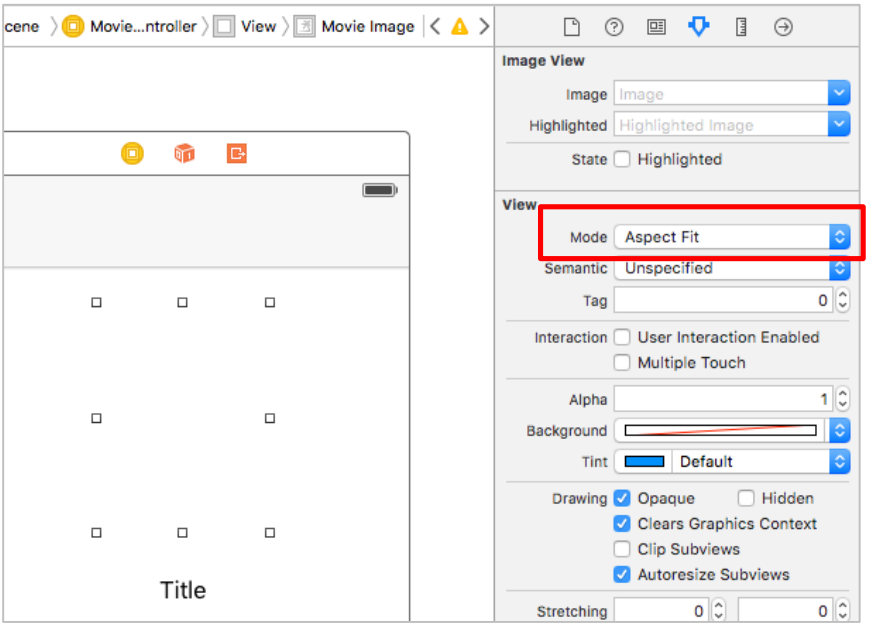
13. Ctrl+click and drag the **Movie Cell** to the newly created View Controller and select the **Selection segue -> push**. Now we are telling iOS that whenever the user taps on one of the Movies, show the new view controller.



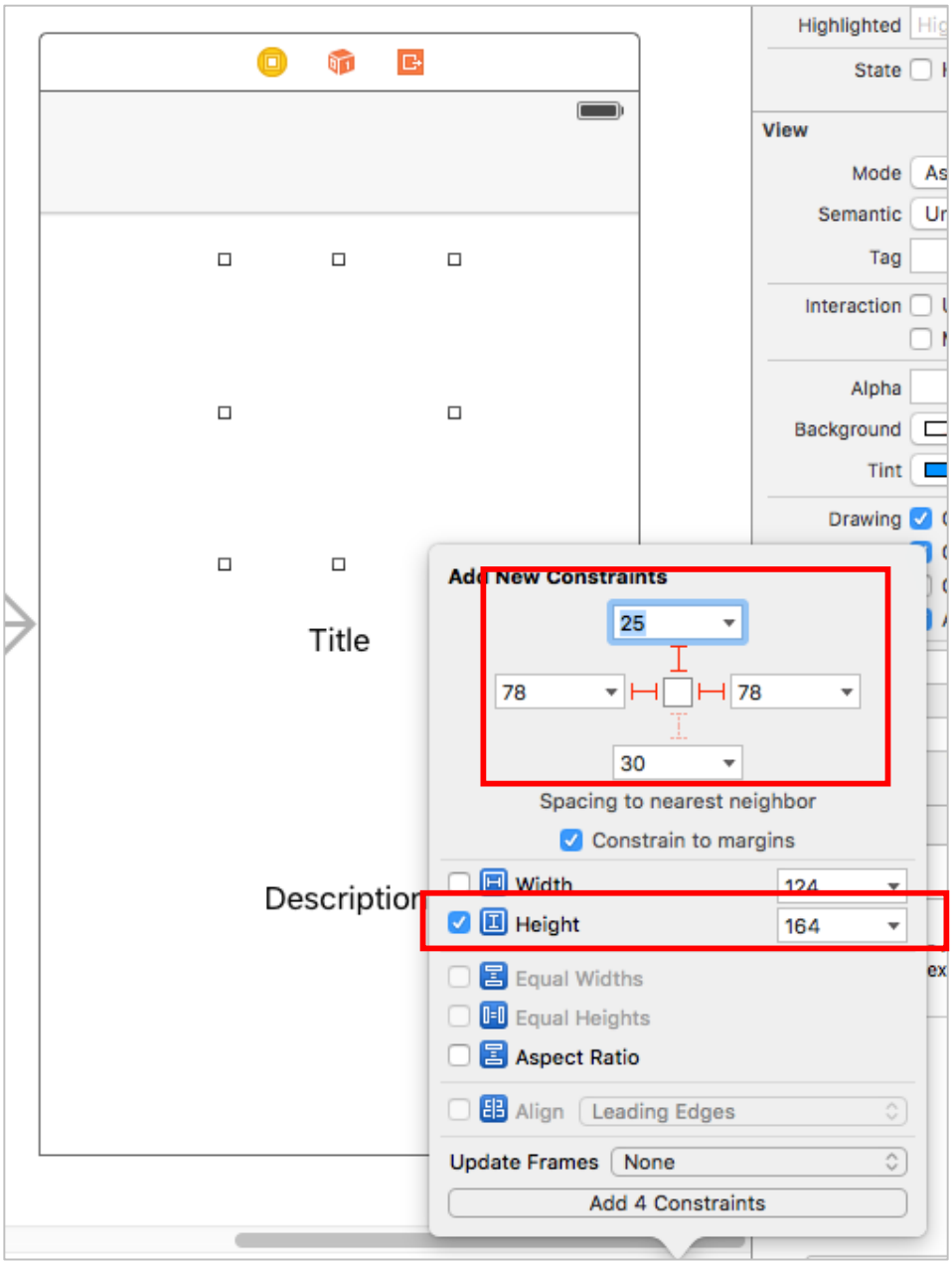
14. Next, design the newly added **ViewController** view as shown below. Note that the **label** for the description of the movie, under the **attribute inspector**, make sure that the **lines** are set to **10**.



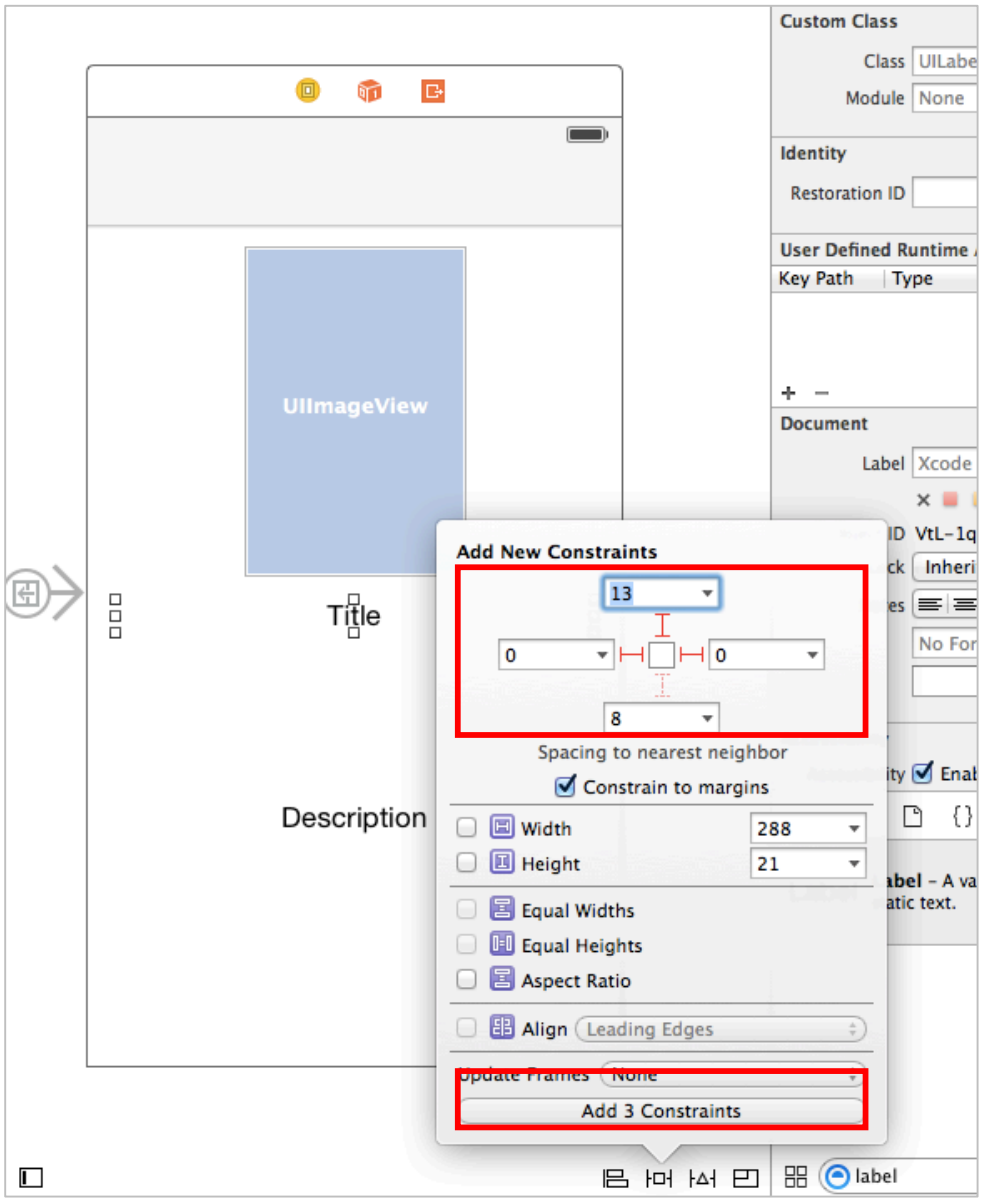
15. Select the Image View and set the Mode to Aspect Fit.



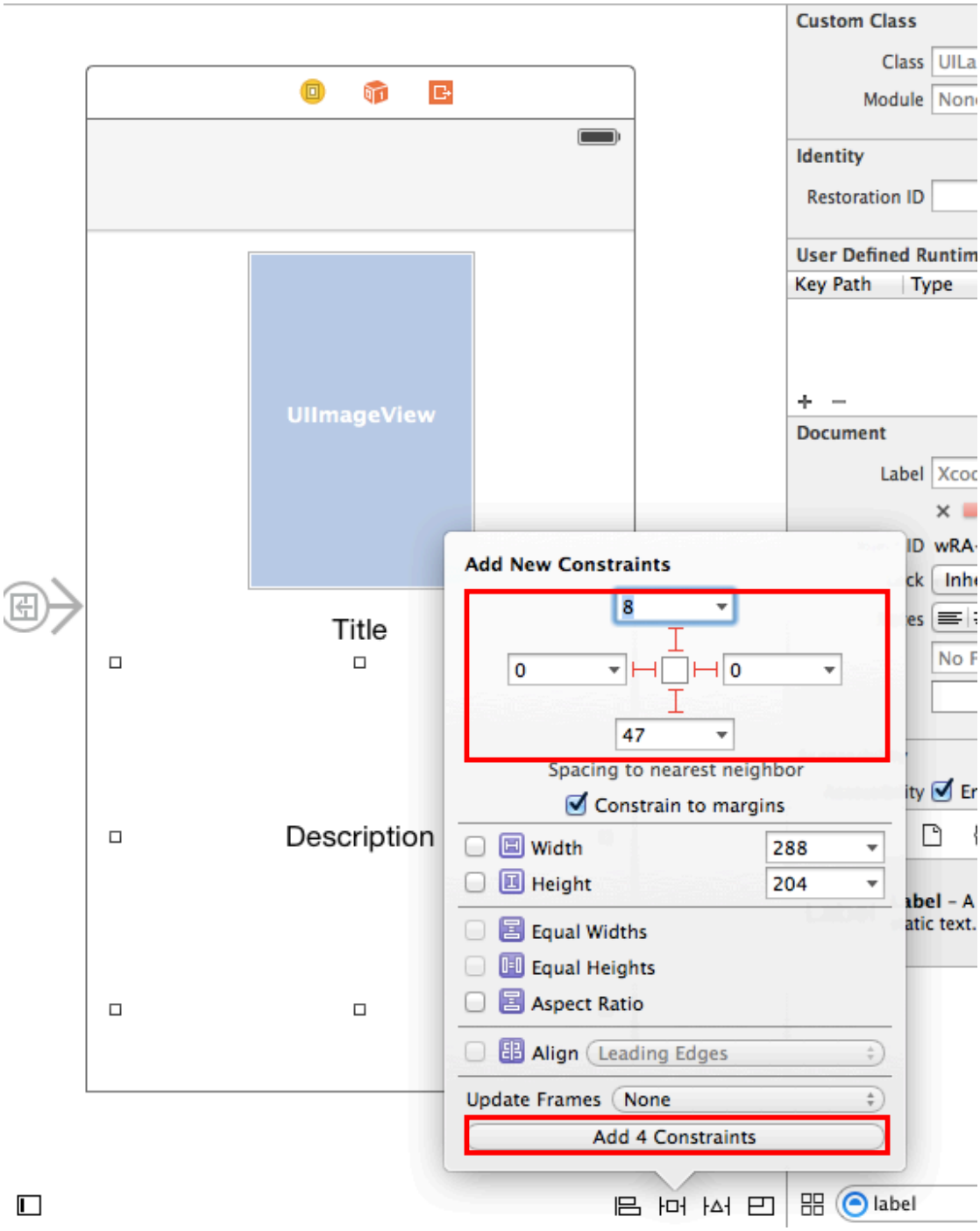
16. Let's add in the constraints so that the items will be placed nicely in all views.
17. Select the **Image View**, from the autolayout menu, select the Pin icon and click on the constraints as shown below:



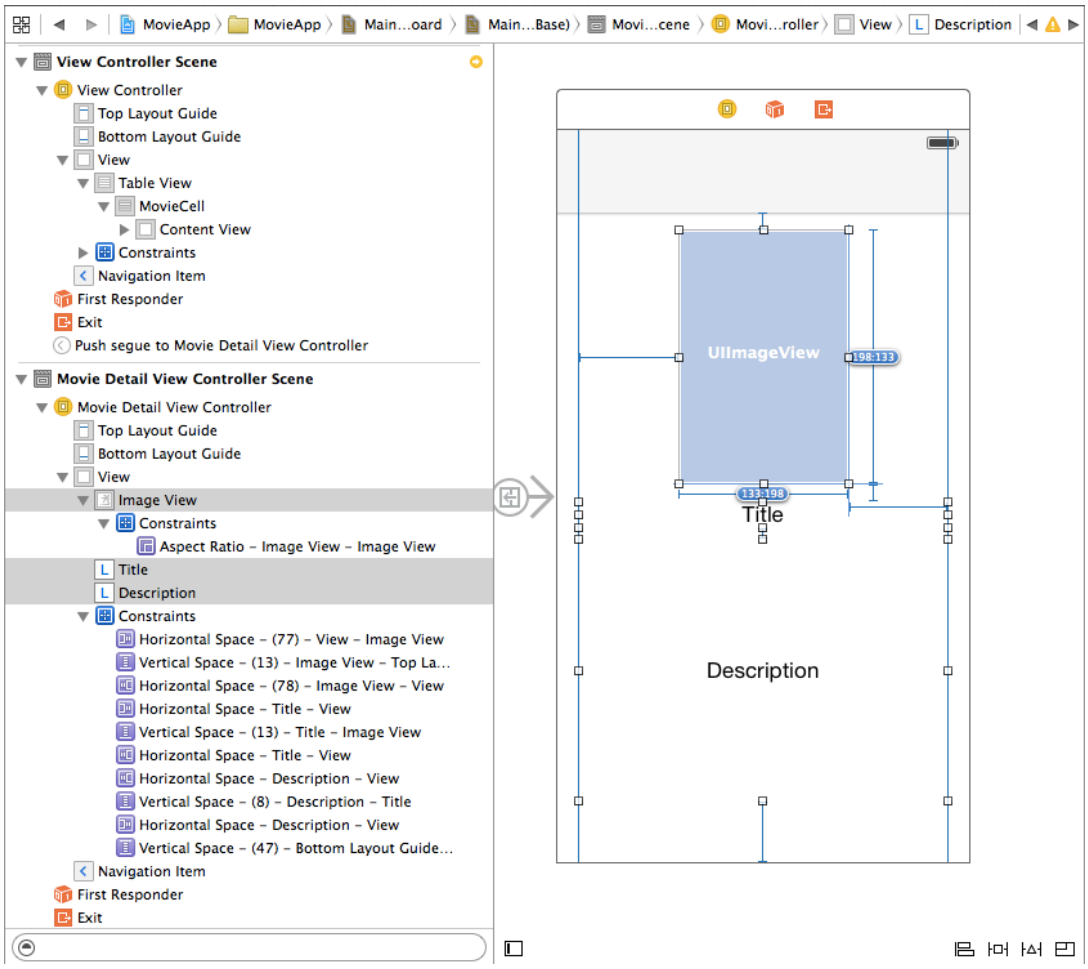
18. Select the **Title Label**, from the autolayout menu, select the Pin icon and click on the constraints as shown below:



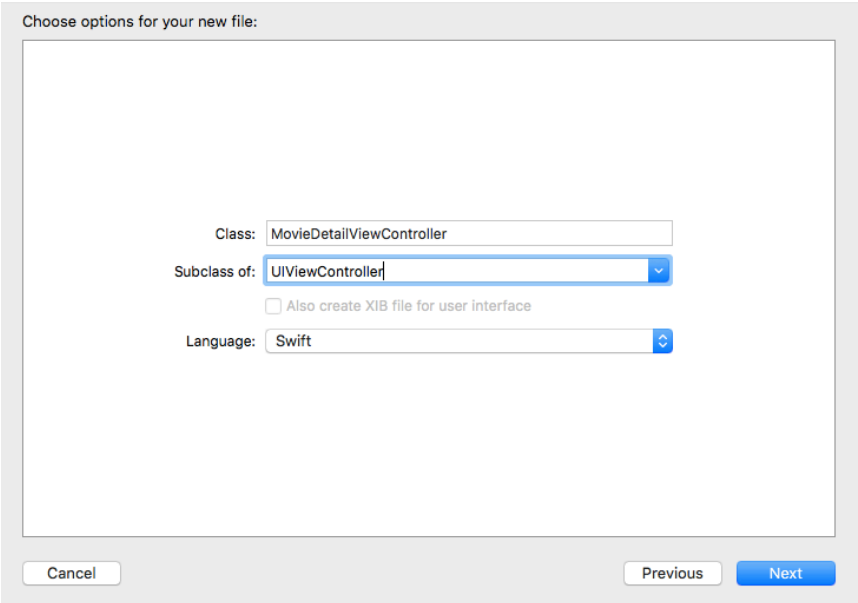
19. Select the **Description Label**, from the autolayout menu, select the Pin icon and click on the constraints as shown below:



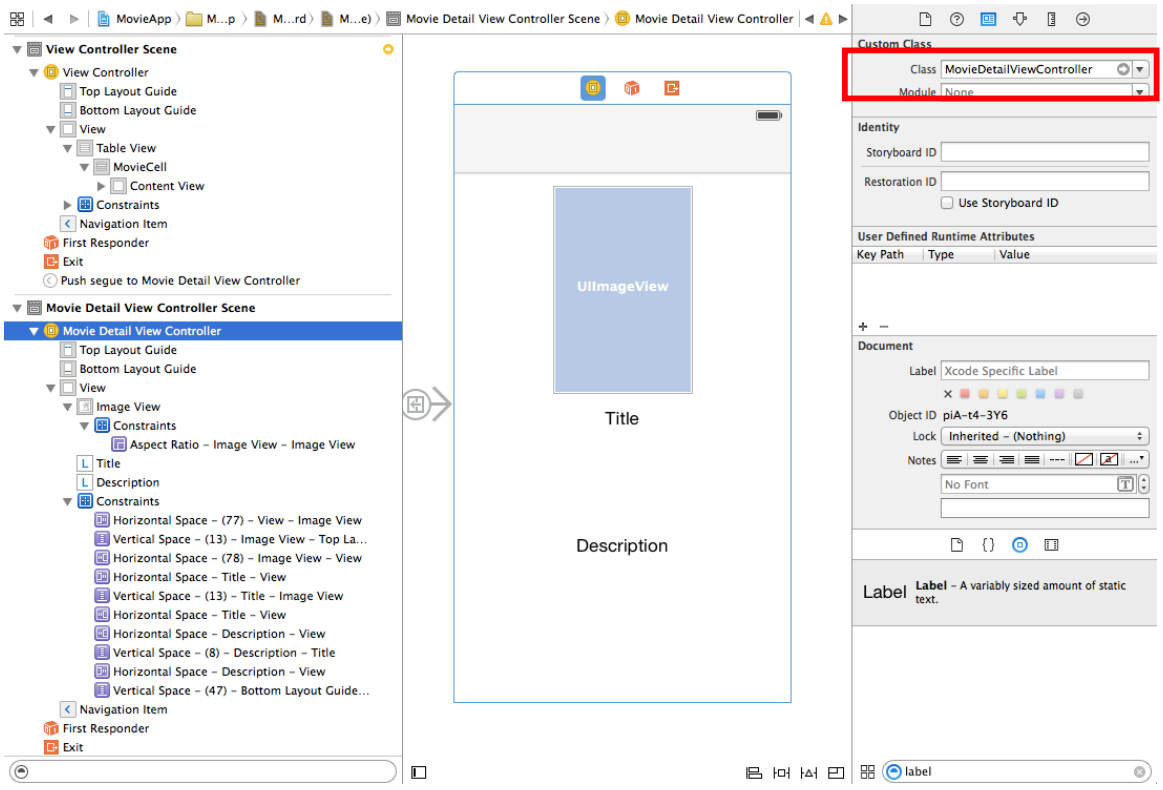
20. You may notice that a red arrow appears in the Interface Builder outline. This tells us that there are some constraint problems with our image view. Xcode can help us fix them. Simply click on that red arrow, followed the red indicator and finally click “Add missing constraints”.



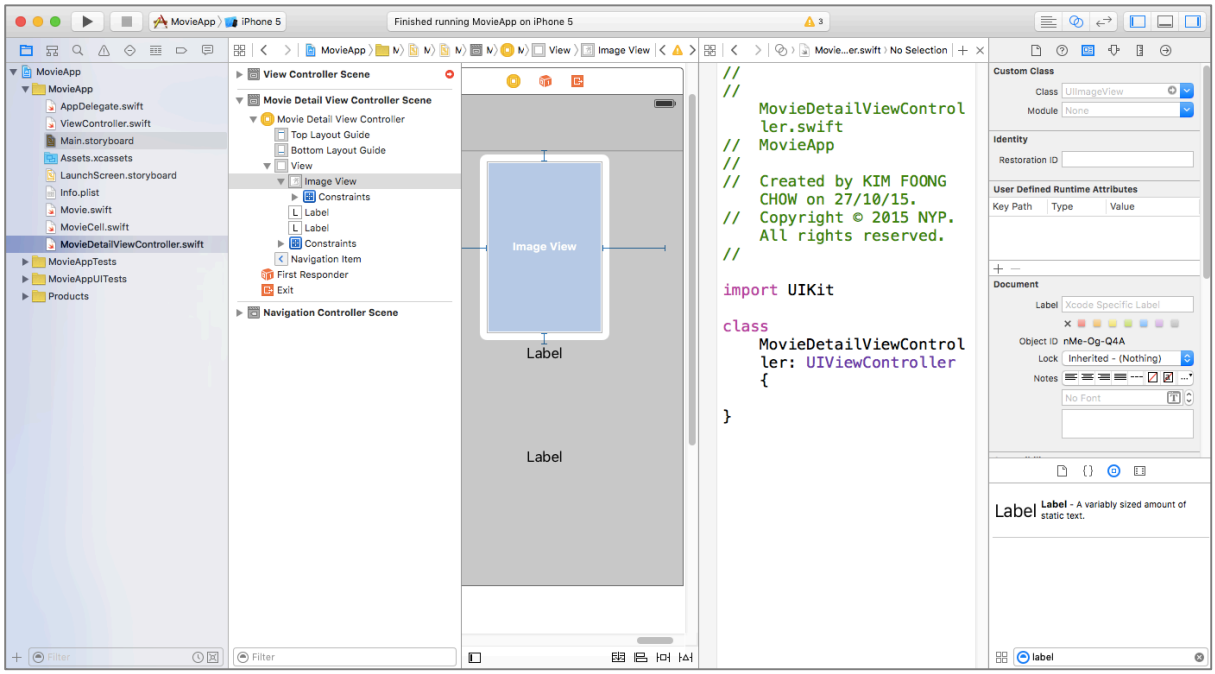
21. Right-click on the project and click on **New File…**. Add **Cocoa Touch** class with **UIViewController** subclass and name it as **MovieDetailViewController**.



22. Click on the **Main.storyboard**. With the newly created View Controller selected, under the **identity inspector**, change the class to **MovieDetailViewController**.

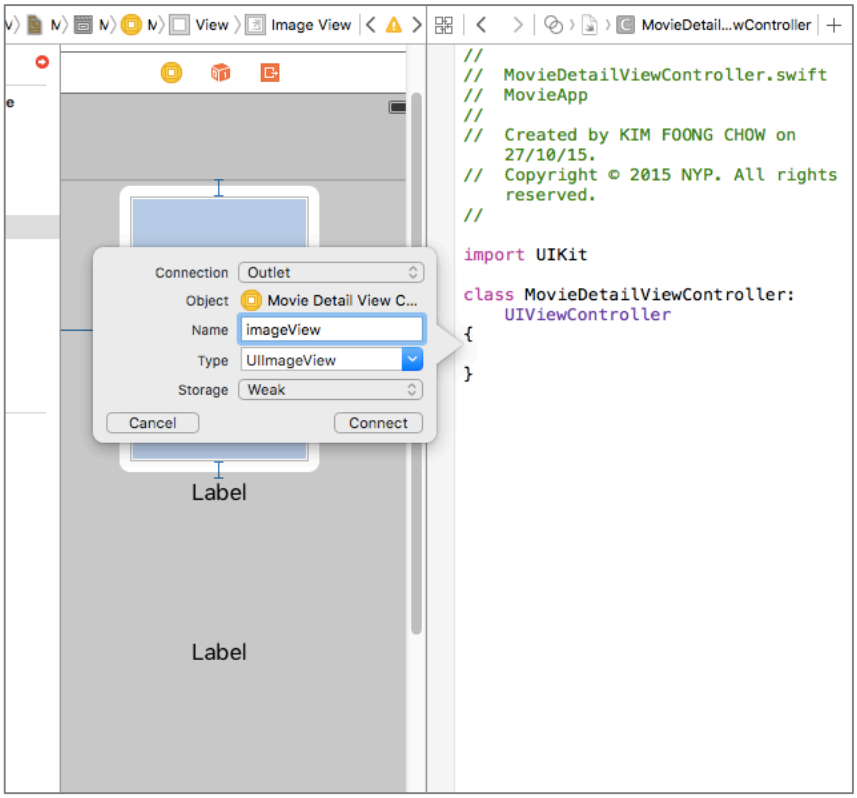


23. In the next few steps, we will demonstrate how we can auto generate an outlet and making a connection without typing any codes. First, click on the **Main.storyboard** in the project navigator and select the View controller that we have created earlier on. Next, **Option-click MovieDetailViewController.swift** in the project navigator. This shortcut will open the file in the **assistant editor**, right next to MovieDetailViewController.

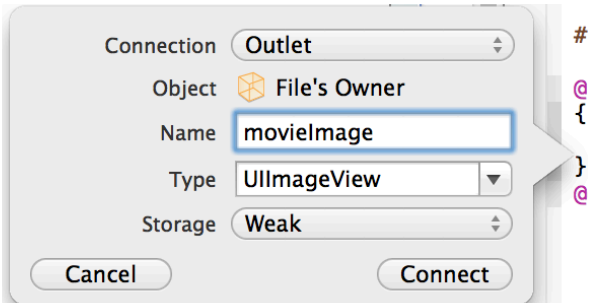


Note: If your file is not open right next to the xib file selected, select **View → Assistant Editor**, make sure your **Assistant Editors on Right** is selected.

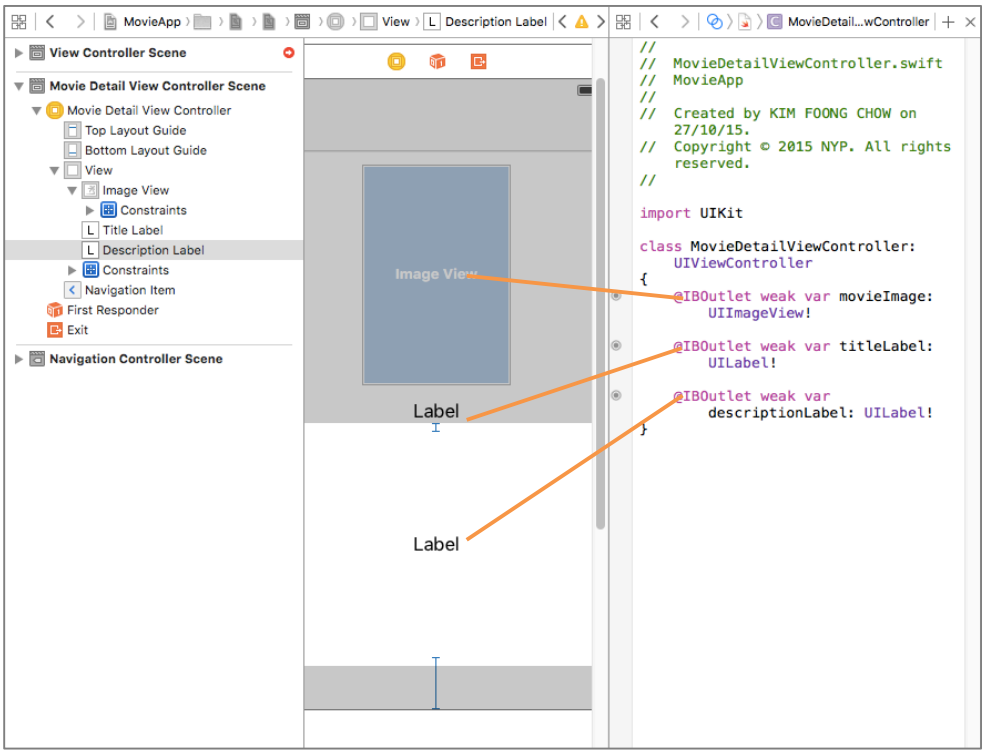
24. Now**, Control-drag** from the **UIImageView** to the instance variable area (inside the curly brackets) in the **MovieDetailViewController.swift** as shown. A pop-up window will be displayed.



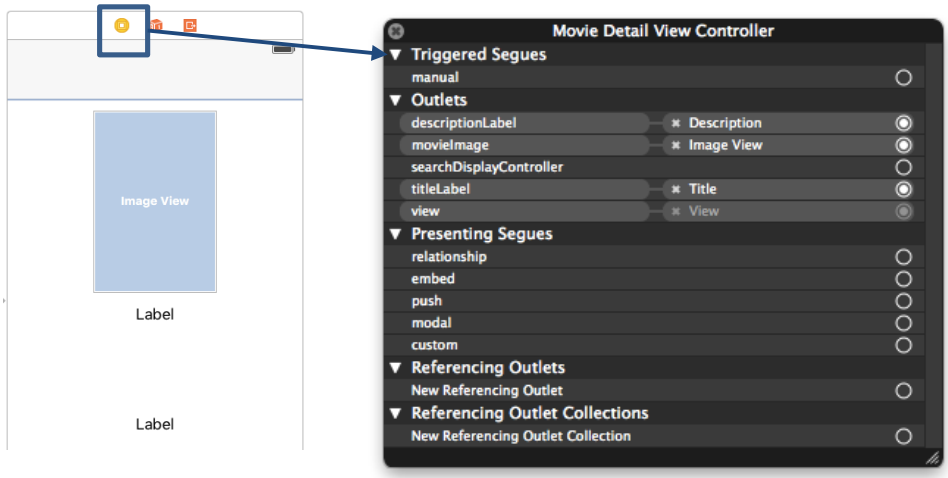
25. In the pop-up window, enter **movieImage** and select **Weak** from the **Storage** drop down list and click connect. This will create an IBOutlet instance variable of type **UIImageView** named **movieImage** in the MovieDetailViewController.



26. Repeat the same for the **2 UILabel** and name it as shown.



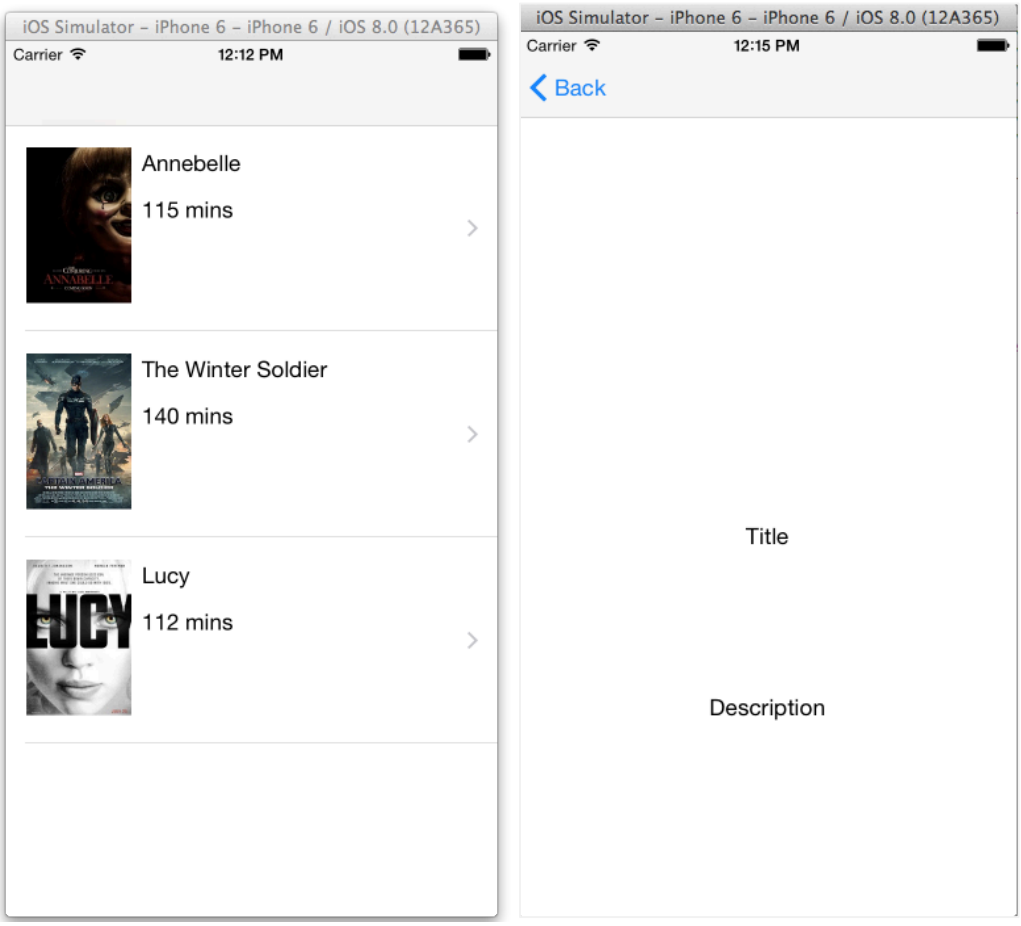
27. To ensure you have connected the outlets correctly, you can **right-click** on the **View Controller icon** in the Storyboard. Ensure that the Connections Inspector window looks as shown:



Note: If there are yellow warning signs next to any connection, click on the **x icon** next to those connections to disconnect them.

It's important to ensure there are no bad connections in the storyboard. A bad connection typically happens when you change the name of the instance variable but do not update the connection in the XIB file. A bad connection will cause your application to crash when the storyboard is loaded.

28. Build and run the application. Notice that when you select one of the movies, the MovieDetailViewController is pushed into the stack of the navigation controller. However, no content is displayed.



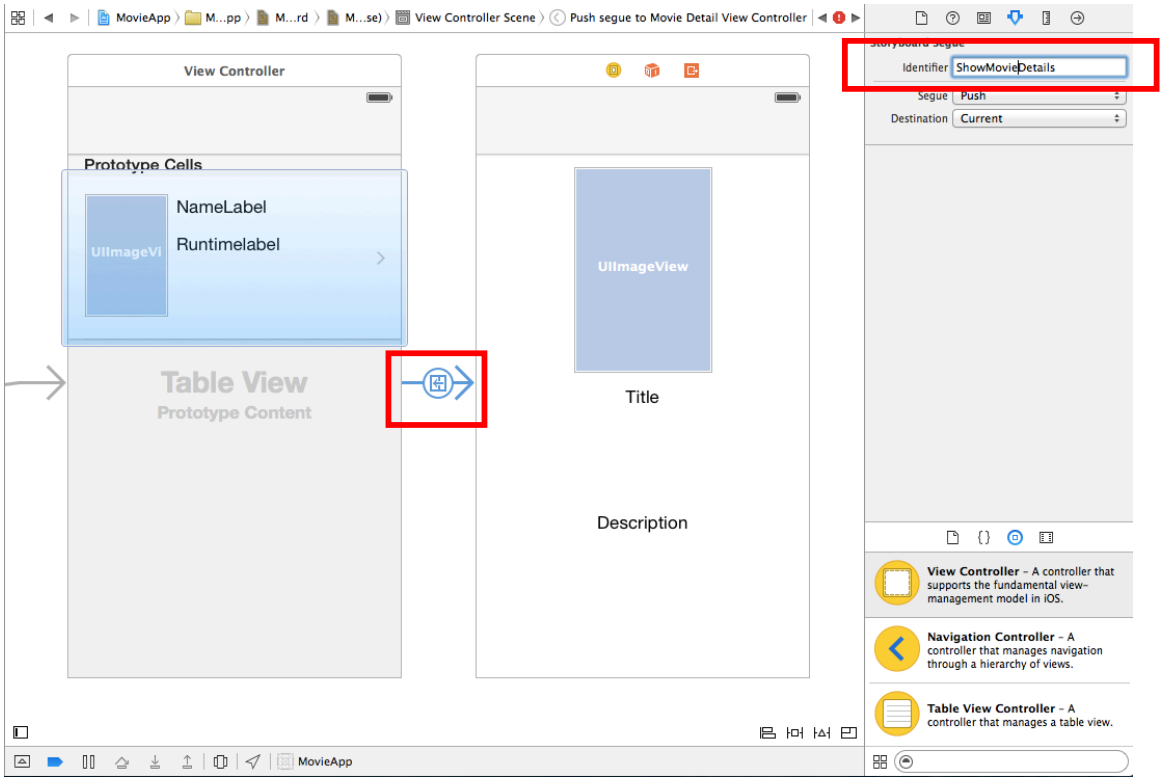
29. The next step will be to pass the Movie Instance from the ItemsViewController to the MovieDetailViewController. In **MovieDetailViewController.swift**, declare Movie and add this property.

```
import UIKit

class MovieDetailViewController: UIViewController
{
    @IBOutlet weak var movieImage: UIImageView!

    @IBOutlet weak var titleLabel: UILabel!

    @IBOutlet weak var descriptionLabel: UILabel!

    var movieItem : Movie?


}
```

30. Besides synthesizing the movieItem, we need to display the contents using the movieItem that was set. In the **MovieDetailViewController.swift**, override **viewWillAppear** method to set the various controls.

```
class MovieDetailViewController: UIViewController
{
    @IBOutlet weak var movieImage: UIImageView!

    @IBOutlet weak var titleLabel: UILabel!

    @IBOutlet weak var descriptionLabel: UILabel!

    var movieItem : Movie?

    // This function is triggered when the view is about to appear.
    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)

        titleLabel.text = movieItem?.movieName
        descriptionLabel.text = movieItem?.movieDesc
        movieImage.image = UIImage(named: (movieItem?.imageName)!)

    }
}
```

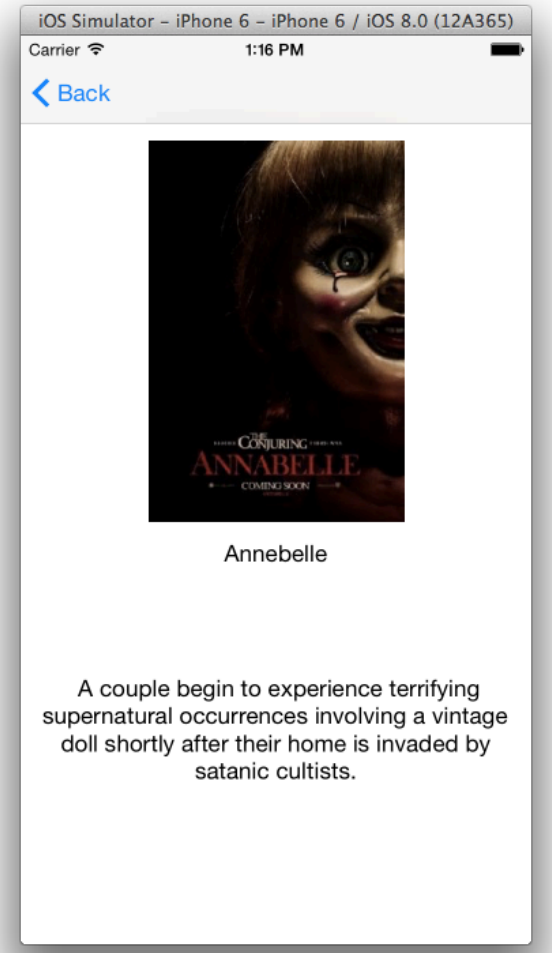
31. Click on the **Main.storyboard.** To assign the identifier, select the segue and set it in the identity inspector. Let's name the segue as "**ShowMovieDetails**".



32. Next, we'll implement the **prepare** method in ViewController, which is the source view controller of the segue. Select the **ViewController.swift** and add the following code:

```
...
    // This function is triggered when a segue will be triggered.
    override func prepare(for segue: UIStoryboardSegue,
        sender: Any?)
    {
        if(segue.identifier == "ShowMovieDetails")
        {
            let detailViewController =
                segue.destination as!
                MovieDetailViewController
            let myIndexPath = self.tableView.indexPathForSelectedRow

            if(myIndexPath != nil)
            {
                // Set the movieItem field with the movie
                // object selected by the user.
                //
                let movie = movieList[myIndexPath!.row]
                detailViewController.movieItem = movie
            }

        }
    }
```

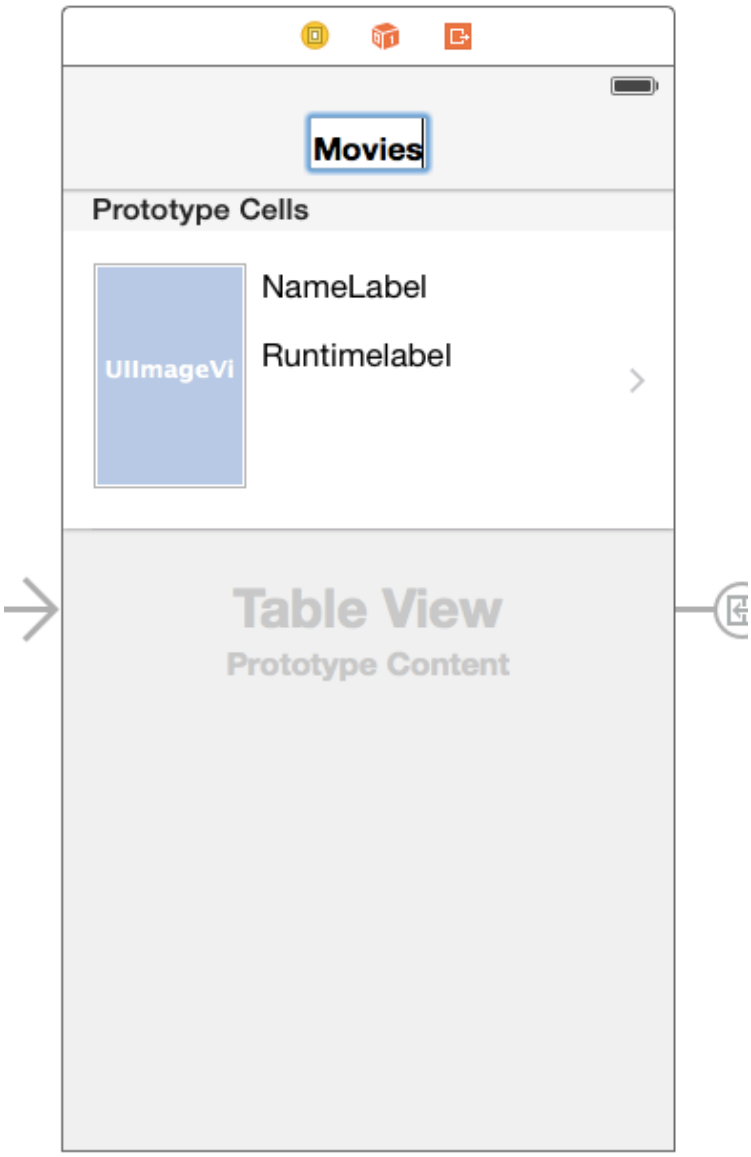
33. We are done. Build and run the application.



Section 3: Setting title to UINavigationBar

Every UIViewController has a navigationItem property of type UINavigationItem. By default, the UINavigationItem is empty. In the last section of the chapter, let's make our application complete with a navigation title.

34. Click on the **Main.storyboard** and select the Navigation Item on the View Controller. Double click on the Navigation item and type in Movies.



35. Last step, let's set the title of the MovieDetailViewController to be the name of the selected movie. In **MovieDetailViewController.swift**, modify the viewWillAppear method to update the navigation bar title.

```
import UIKit

class MovieDetailViewController: UIViewController
{
    @IBOutlet weak var movieImage: UIImageView!

    @IBOutlet weak var titleLabel: UILabel!

    @IBOutlet weak var descriptionLabel: UILabel!


    // This function is triggered when the view is about to appear.
    override func viewWillAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
```

```
        titleLabel.text = movieItem?.movieName
        descriptionLabel.text = movieItem?.movieDesc
        movieImage.image = UIImage(named: (movieItem?.imageName)!)

        self.navigationItem.title = movieItem?.movieName
    }

}
```

36. Build and run the application.