**Mobile Computing – iOS Spring’23**

**Assignment05**

**30 Points**

**Please follow the following instructions to complete this assignment.**

1. Open Xcode from the launchpad of your Mac.
2. Click on create a new Xcode project. Select the iOS template and click on the App application.
3. Click on next which will prompt you to choose options for the project.
4. Provide product name as **LastnameAlbumApp**, “**edu.nwmissouri**” for organization identifier, “**storyboard**” as interface and “**Swift**” as the language.
5. Click on next and select an appropriate location to save your app and click on create. A project directory will be loaded.
6. From the project navigator click on “Main.storyboard” file, a blank mobile screen will be loaded, where the required fields for an app need to be added.

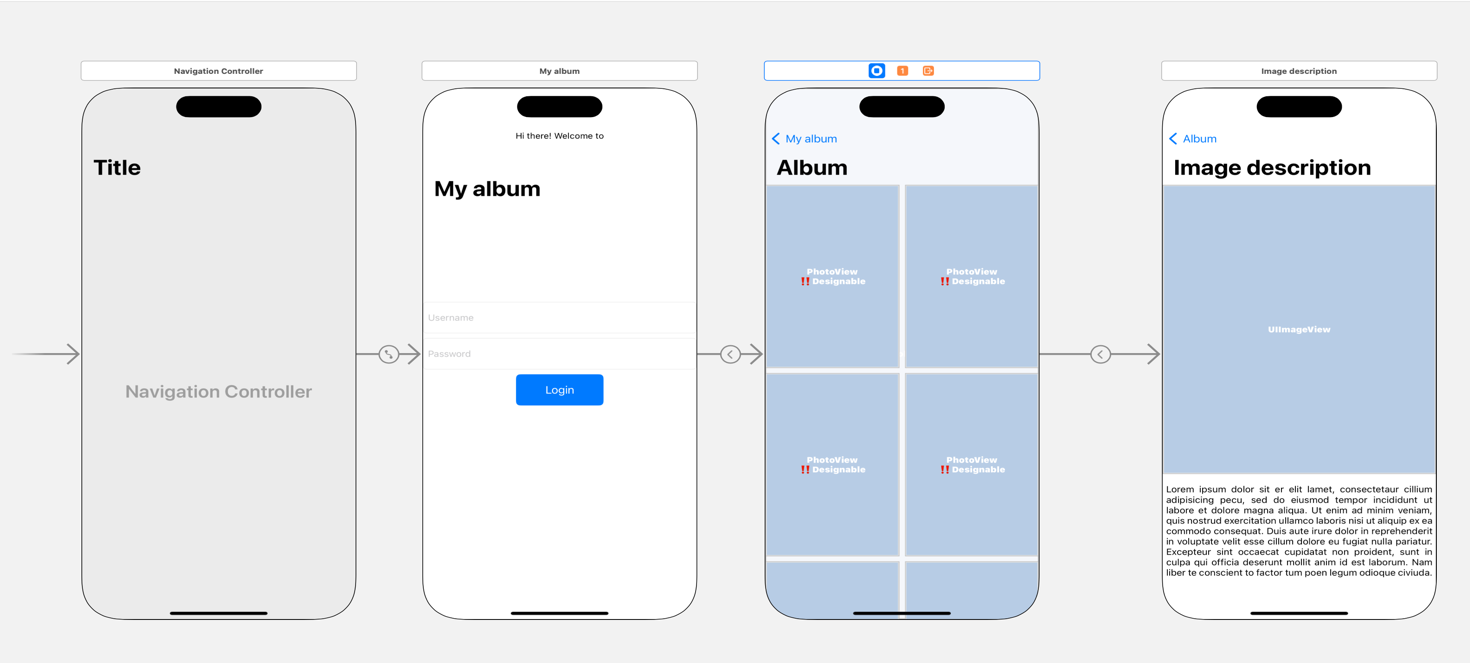
**The View**

1. At first, you will see a single view (first view) in the Main.storyboard file. Select that view and embed it in a navigation controller.
   1. Editor > Embed In > Navigation Controller
2. Check Prefers Large Title property for the navigation bar.
3. Create a Cocoa Touch Class “RootVC” that is a sub class of UIViewController and assign it as class to the first view that is present in the Main.storyboard file. It will be the root view controller of the navigation controller.
4. Add two more view controllers to the storyboard for gallery view and image description view, respectively.

Table 1 The View Controllers

|  |  |
| --- | --- |
| **Screen** | **View Controller** |
| 1 | RootVC |
| 2 | AlbumVC |
| 3 | ImageDescVC |

1. Refer to Table 1 to create required classes and assign them to proper view controllers in the storyboard.
   1. To do it, repeat Step 2 for AlbumVC and ImageDescVC.
2. Design the views as per the Figures 1, 2, 3, and 4.



3

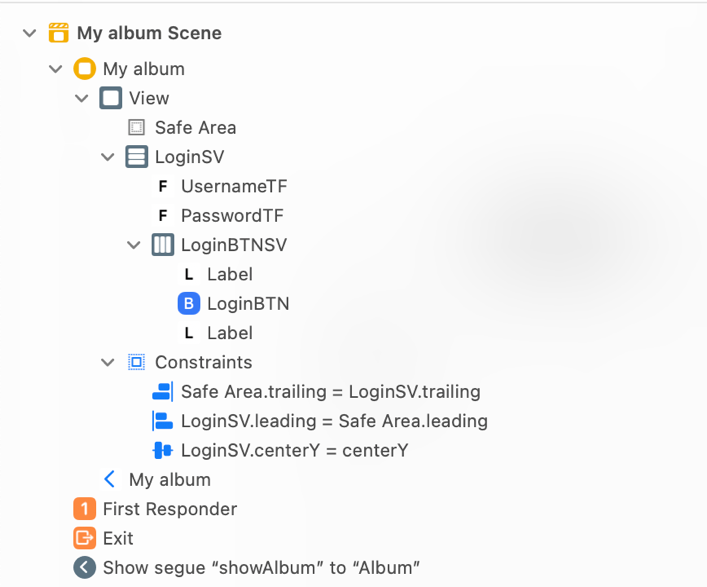
2

1

Segue2

Segue1

Figure 1 The View



The two labels in Login stack view are for padding.

Figure 2 Auto Layout for RootVC

Table 2 RootVC's properties for stack views

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stack View** | **Axis** | **Alignment** | **Distribution** | **Spacing** |
| LoginSV | Vertical | Fill | Fill | Standard |
| LoginBTNSV | Horizontal | Fill | Fill Equally | Standard |

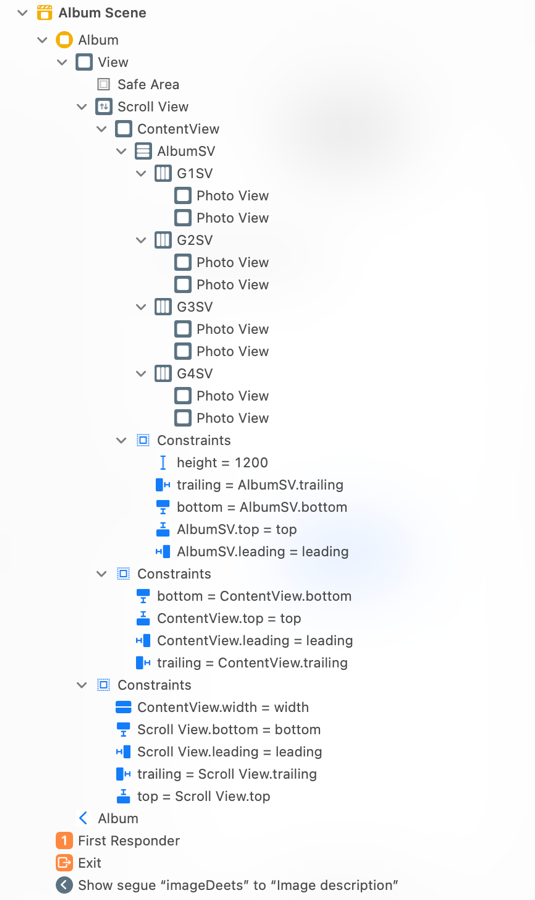


Figure 3 Auto Layout for AlbumVC

Table 3 AlbumVC's properties for stack views

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stack View** | **Axis** | **Alignment** | **Distribution** | **Spacing** |
| AlbumSV | Vertical | Fill | Fill Equally | Standard |
| G1SV, G2SV, G3SV, G4SV | Horizontal | Fill | Fill Equally | Standard |

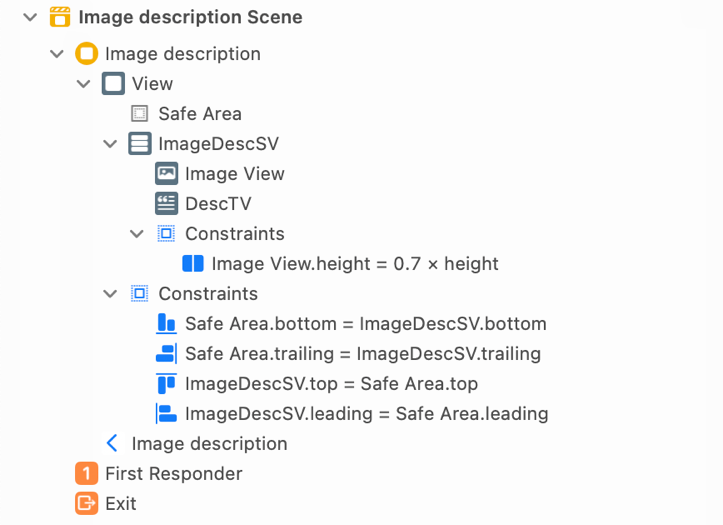


Figure 4 Auto Layout for ImageDescVC

Table 4 ImageDesctVC's properties for stack views

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Stack View** | **Axis** | **Alignment** | **Distribution** | **Spacing** |
| ImageDescSV | Vertical | Fill | Fill | Standard |

Table 5: UI elements configuration for RootVC

|  |  |  |
| --- | --- | --- |
| **UI element** | **Purpose** | **Outlet/action name** |
| 2 text fields | To enter username | userNameTF |
| To enter password | passwordTF |
| 1 UI Button | To interact with it | loginBTN |

Table 6 UI elements configuration for AlbumVC

|  |  |  |
| --- | --- | --- |
| **UI element** | **Purpose** | **Outlet/action name** |
| 1 scroll view | To configure it | scrollView |
| 1 normal view | For zooming in and out | contentView |
| 1 outlet collection for UIImage | To group all the image views | vehicles |

Table 7 UI elements configuration for ImageDescVC

|  |  |  |
| --- | --- | --- |
| **UI element** | **Purpose** | **Outlet/action name** |
| 1 image view | To display an image | imageView |
| 1 text view | To display text related to the image | descTV |

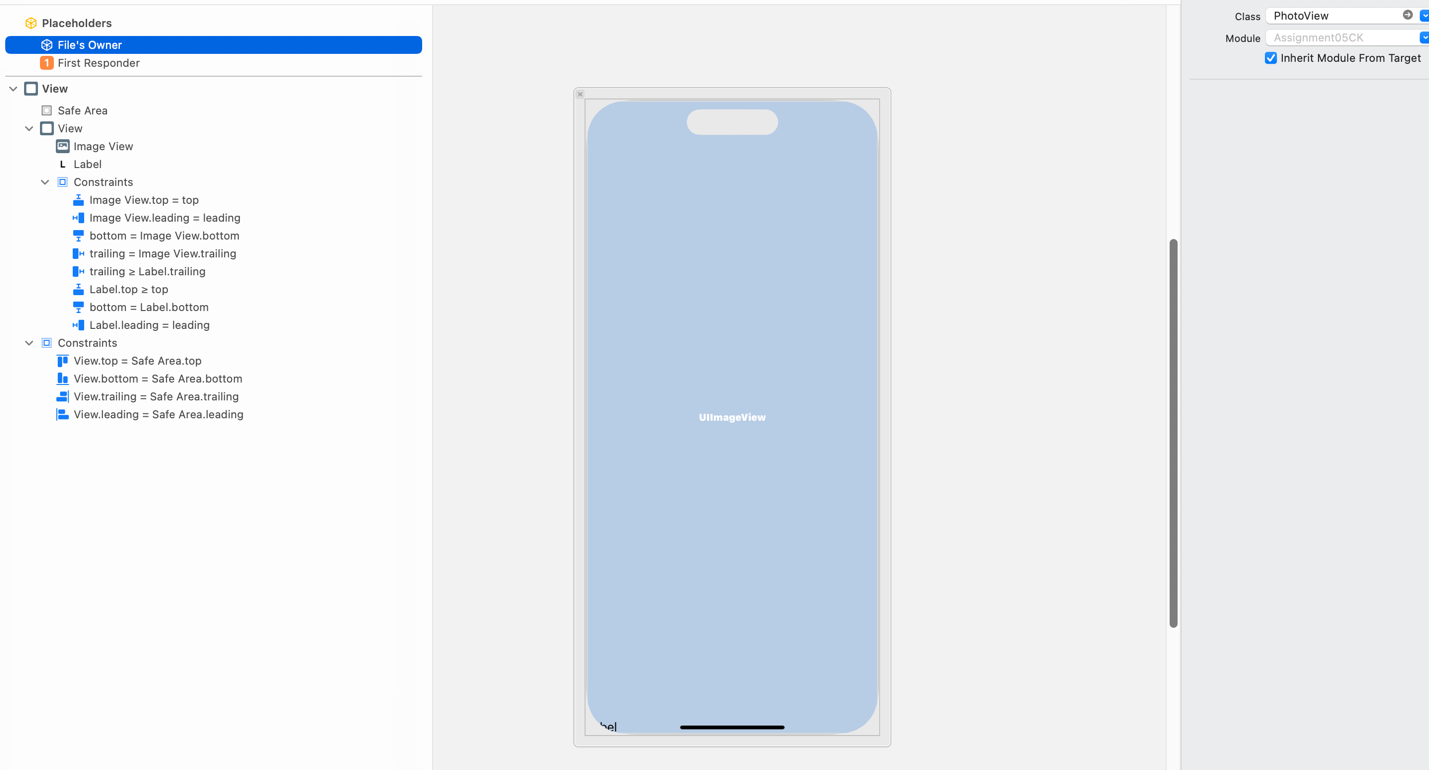
Table 8 UI elements configuration for PhotoView

|  |  |  |
| --- | --- | --- |
| **UI element** | **Purpose** | **Outlet/action name** |
| 1 image view | To display an image | imageView |
| 1 label | To display the like symbol | likeLBL |
| 1 normal var property | To render the brief history of the image | imageDescription |

*Note: Define appropriate methods wherever required to achieve the required functionality via @IBActions and @Objc functions.*

**A Custom UIView**

1. Create a custom view to configure the look of each vehicle in the album.
2. First create a class PhotoView that is a subclass of UIView. Next, create a user interface view file and name it as PhotoView as well, but the file type will be a .xib.



Set this to PhotoView.

Figure The PhotoView

1. Design the UI and apply auto layout constraints as shown in the Figure 5.

**The Model**

1. Create a Swift file called Model that has the following struct.



1. Property vehicles is an array of tuples. Each tuple is a combination of image (that you provide in your Assets folder), and its brief history description.
   1. Note: For each image in the Assets folder, provide @2x and @3x versions as well.

**The Controller**

1. ***PhotoView***
   1. Define the required initializers for the PhotoView.
   2. Set background view corner radius to 10.0, border color to dark gray, and border width to 2.0.
   3. Add a double tap gesture to the image view that shows or hides the like symbol ❤️.
   4. By default, hide the likeLBL.
   5. Create a variable property called imageDescription to save the short notes about the image.
2. ***RootVC***
   1. The passwordTF should only be enabled if username is “admin”.
   2. The loginBTN should only be enabled if username is “admin” and password in “password”.
   3. On tapping the loginBTN, perform a segue (i.e., Segue1 with identifier “showAlbum”) to AlbumVC as shown in Figure 1.
3. ***AlbumVC***
   1. For each image in the album, add a long press gesture that performs a segue (i.e., Segue2 with identifier “imageDeets”) to ImageDescVC as shown in Figure 1. Moreover, play an appropriate sound when a tap is recognized.
      1. Make sure to enable user interaction.
      2. Set content mode to “Scale to Fill.”
   2. While performing the segue, send the long pressed PhotoView’s tag value to the ImageDescVC to display the details of the image using the AppConstants struct.
   3. Enable zooming feature for the scroll view.
      1. Set scrollView minimum zoom scale to 1/25 and maximum zoom scale to 1.0
4. ***ImageDescVC***
   1. When the view is loaded, show the image and its related description.
   2. Ass swipe gestures up and down to the image such that swipe up will zoom in the image by 200%, whereas swipe down on the image will bring it back to its original size.
   3. Apply a corner radius of 20.0 to the image view.

**Sample Output (Video Link):**

<https://app.vidgrid.com/view/kkkb00YUe6qn>