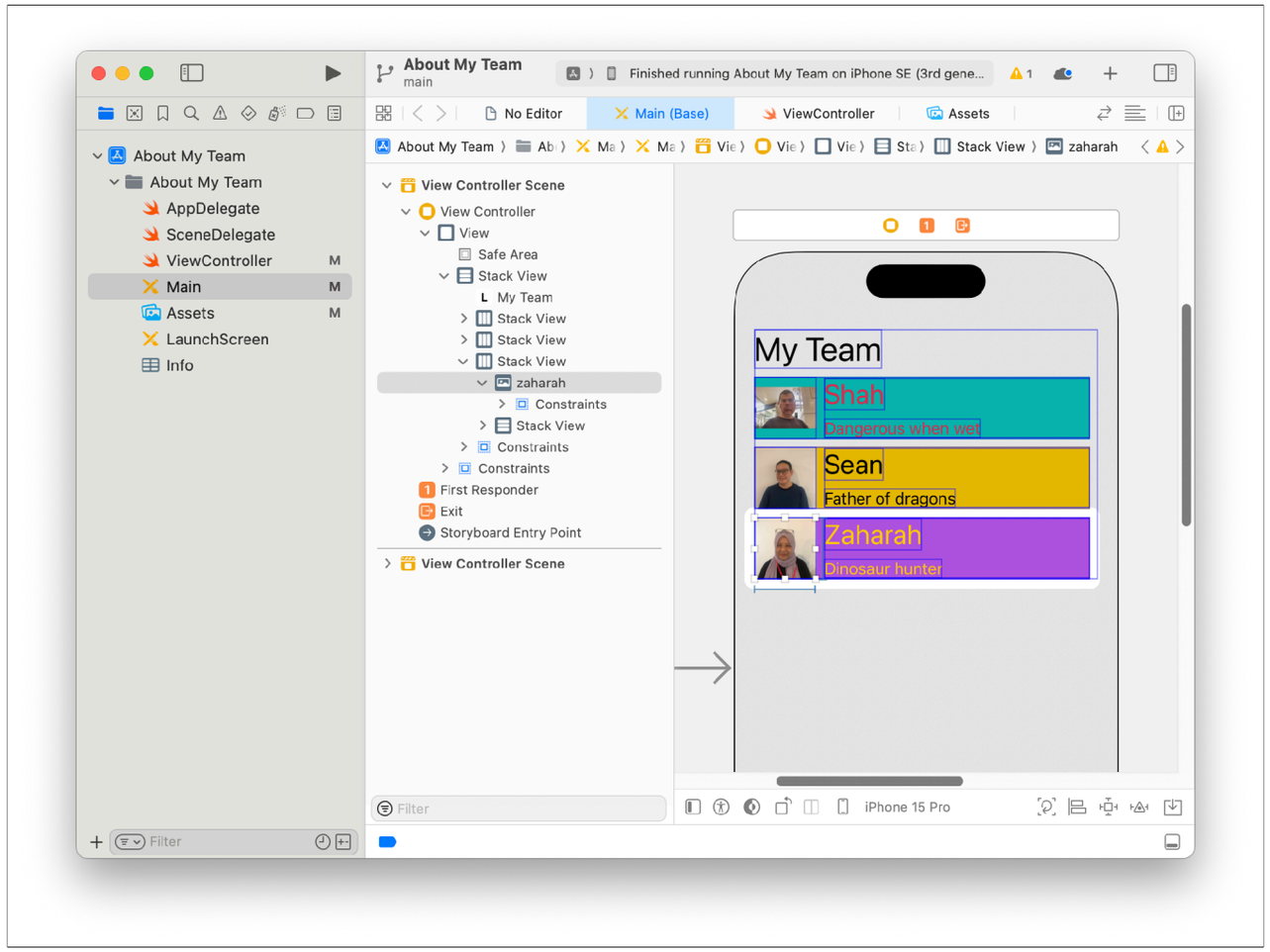
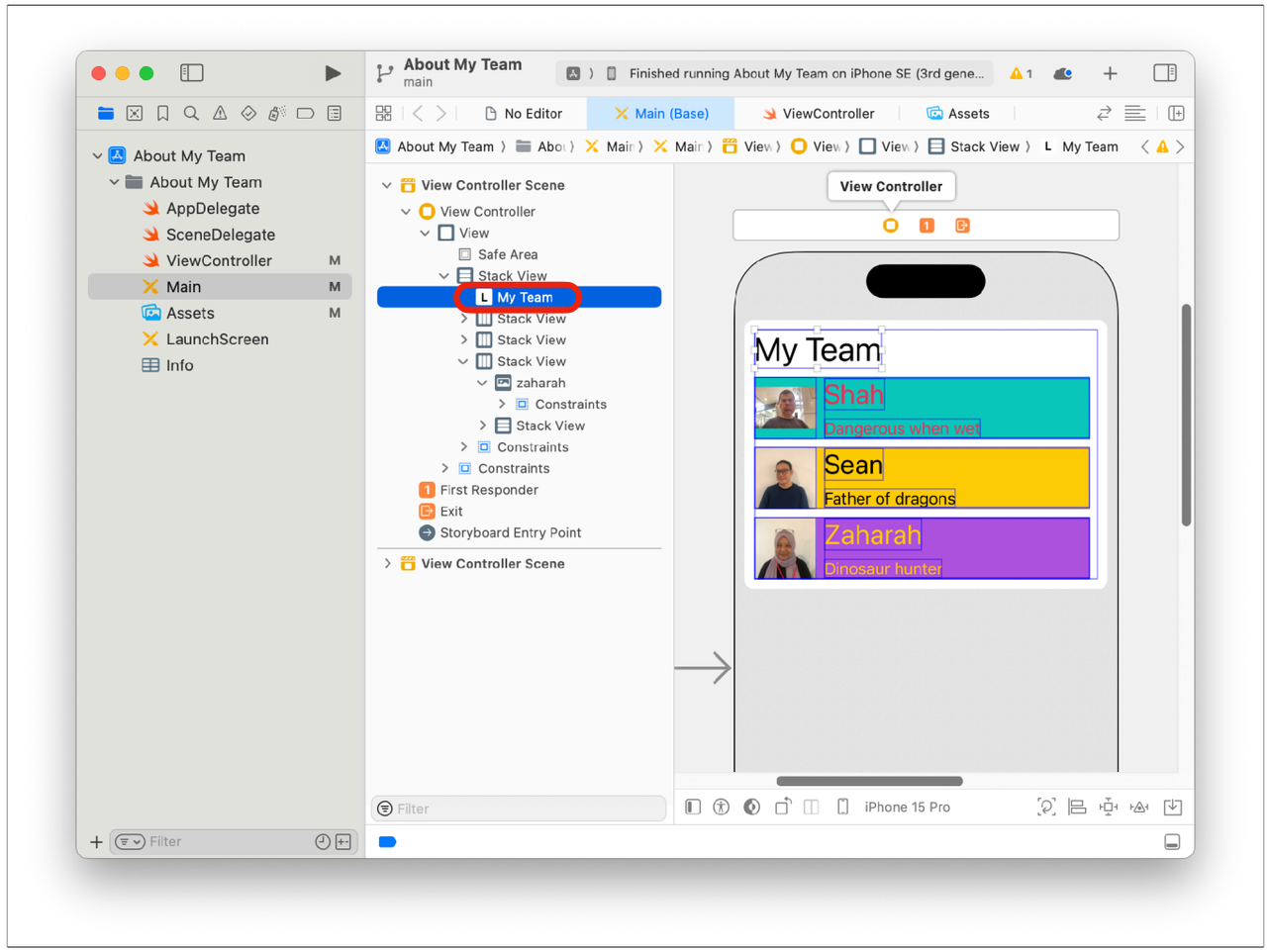
3. Navigation

UIKit Academy

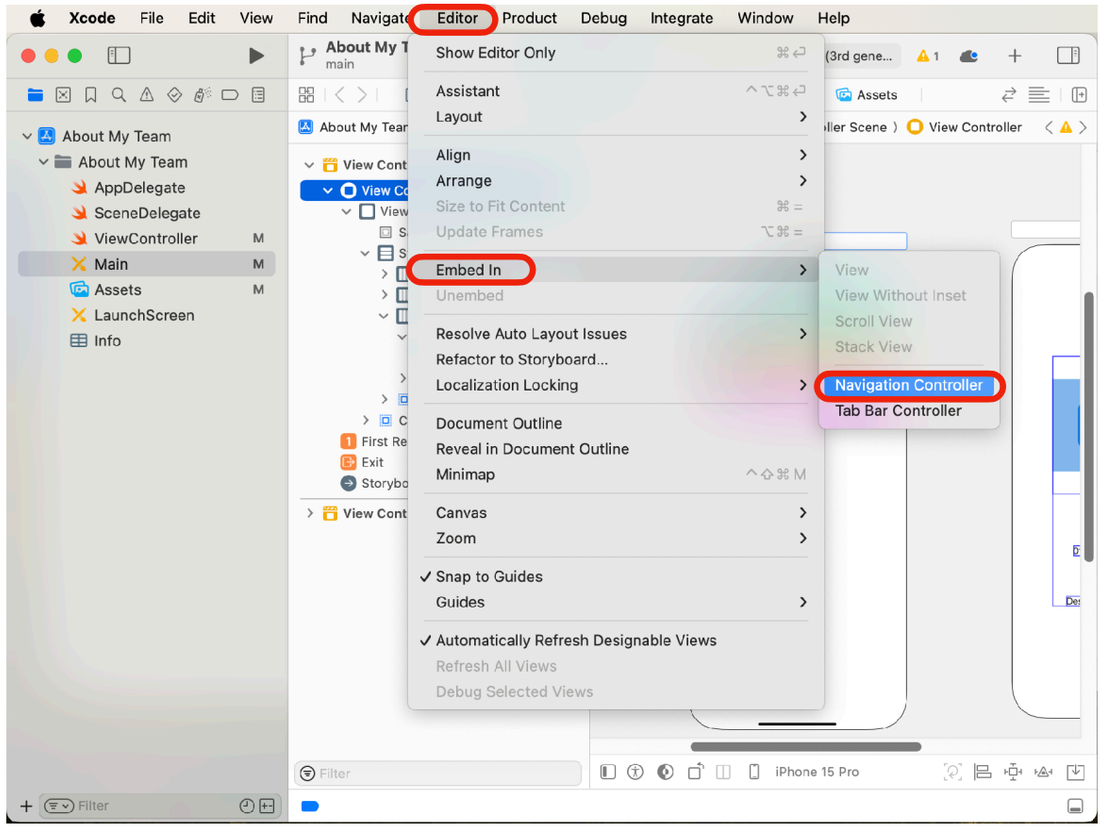
shah@mac.com



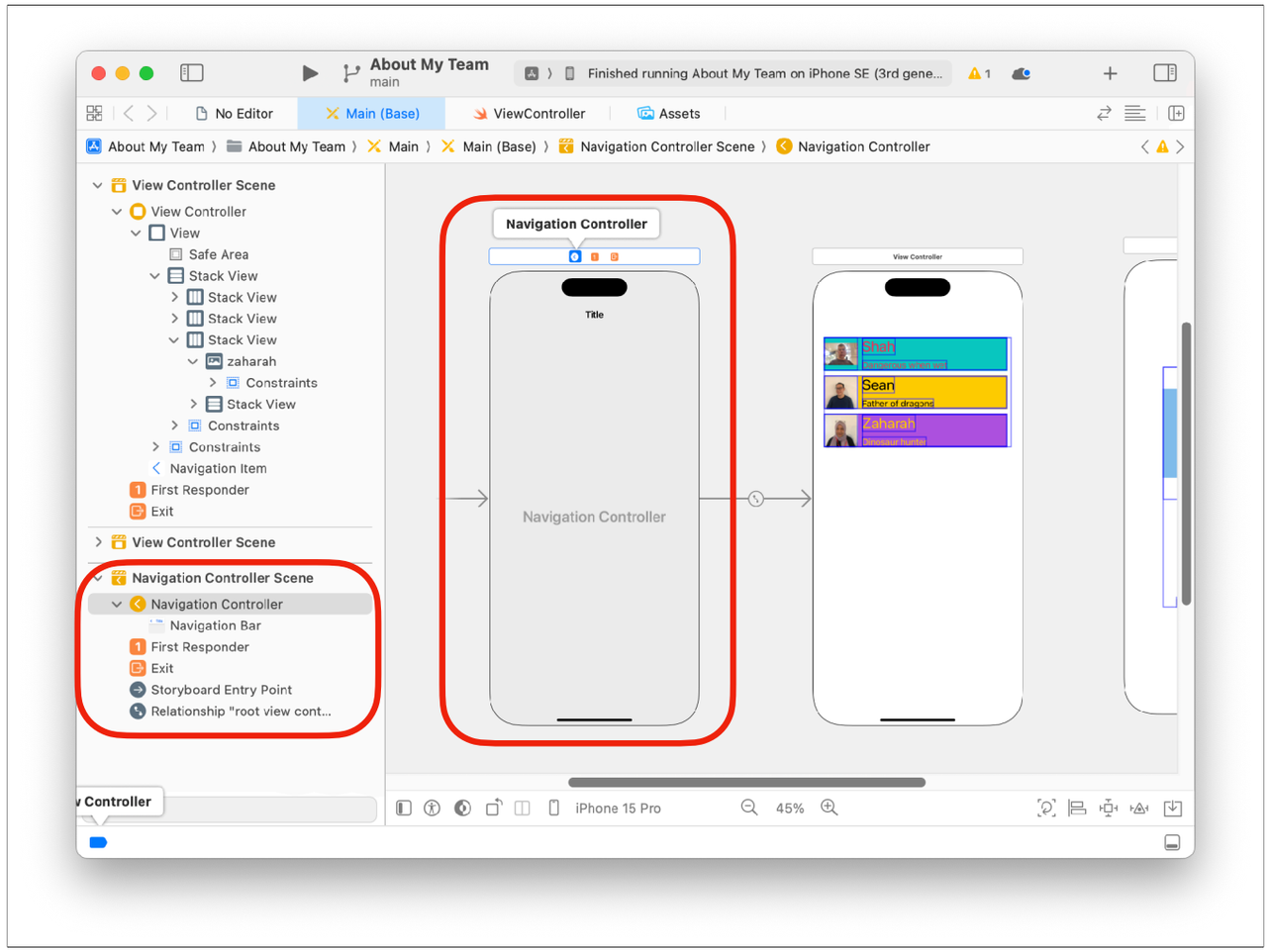
Open the About My Team app that you created previously.



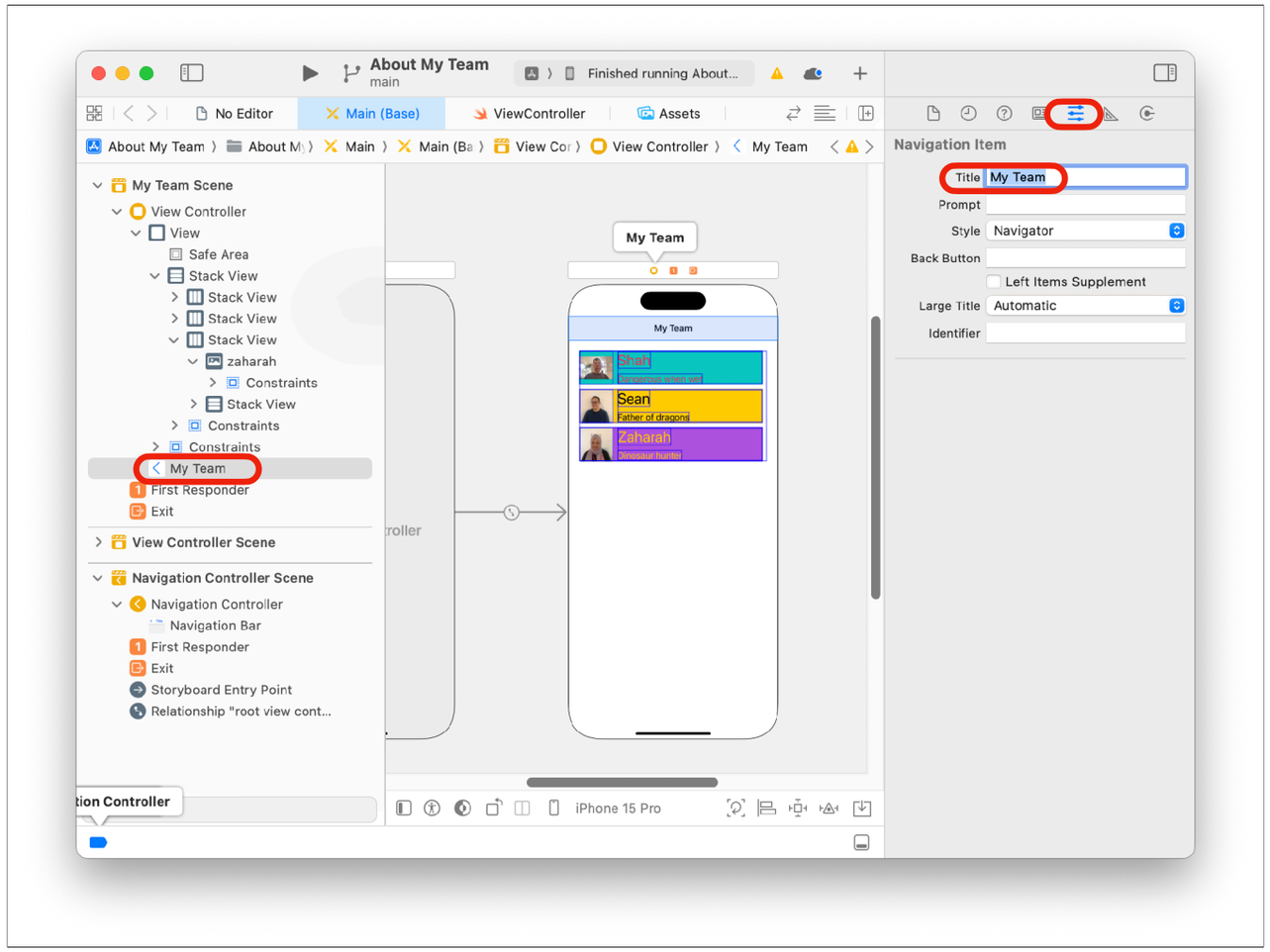
Delete the label in the Stack View.



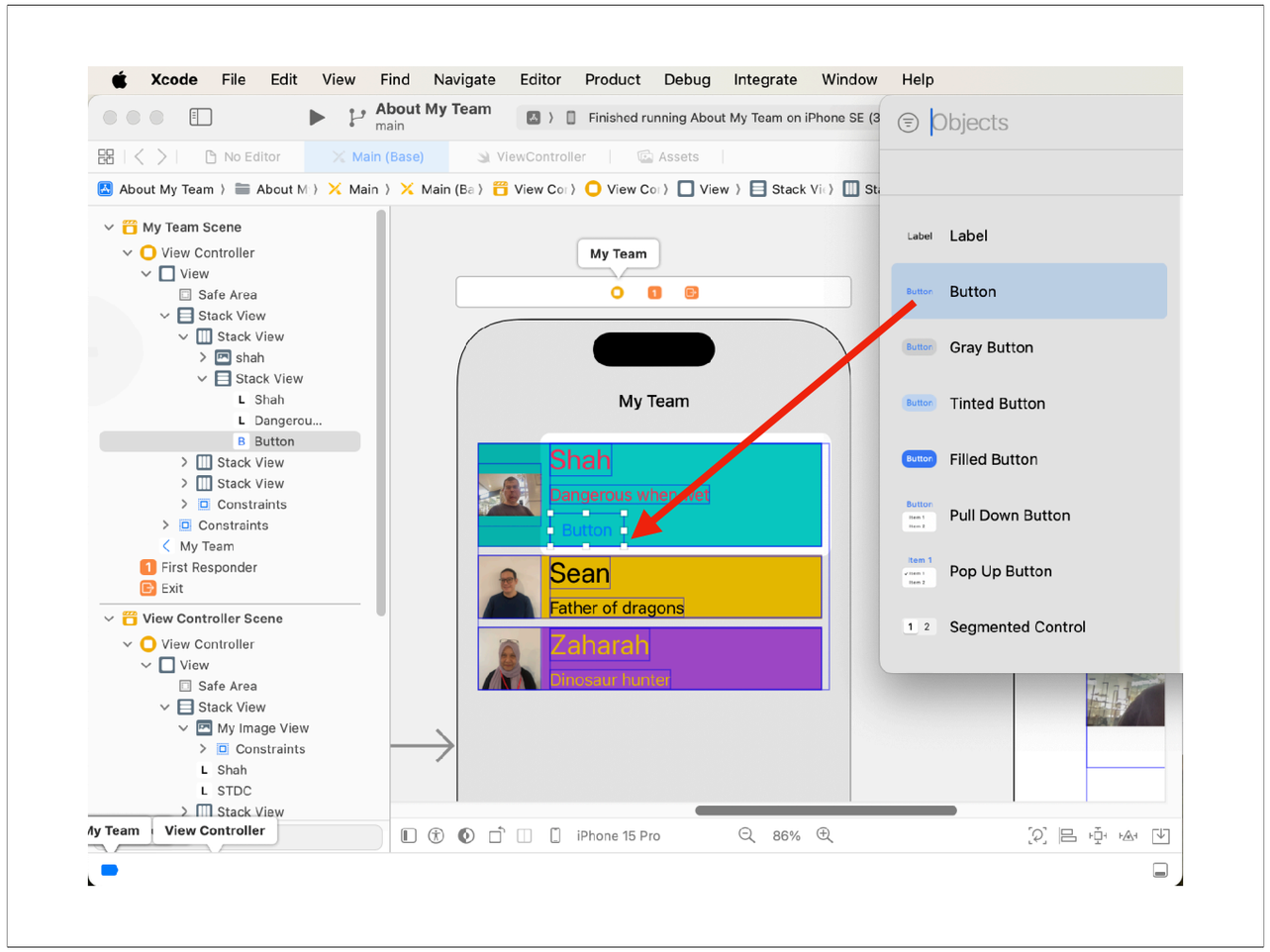
With the View Controller selected, choose Editor > Embed In > Navigation Controller.



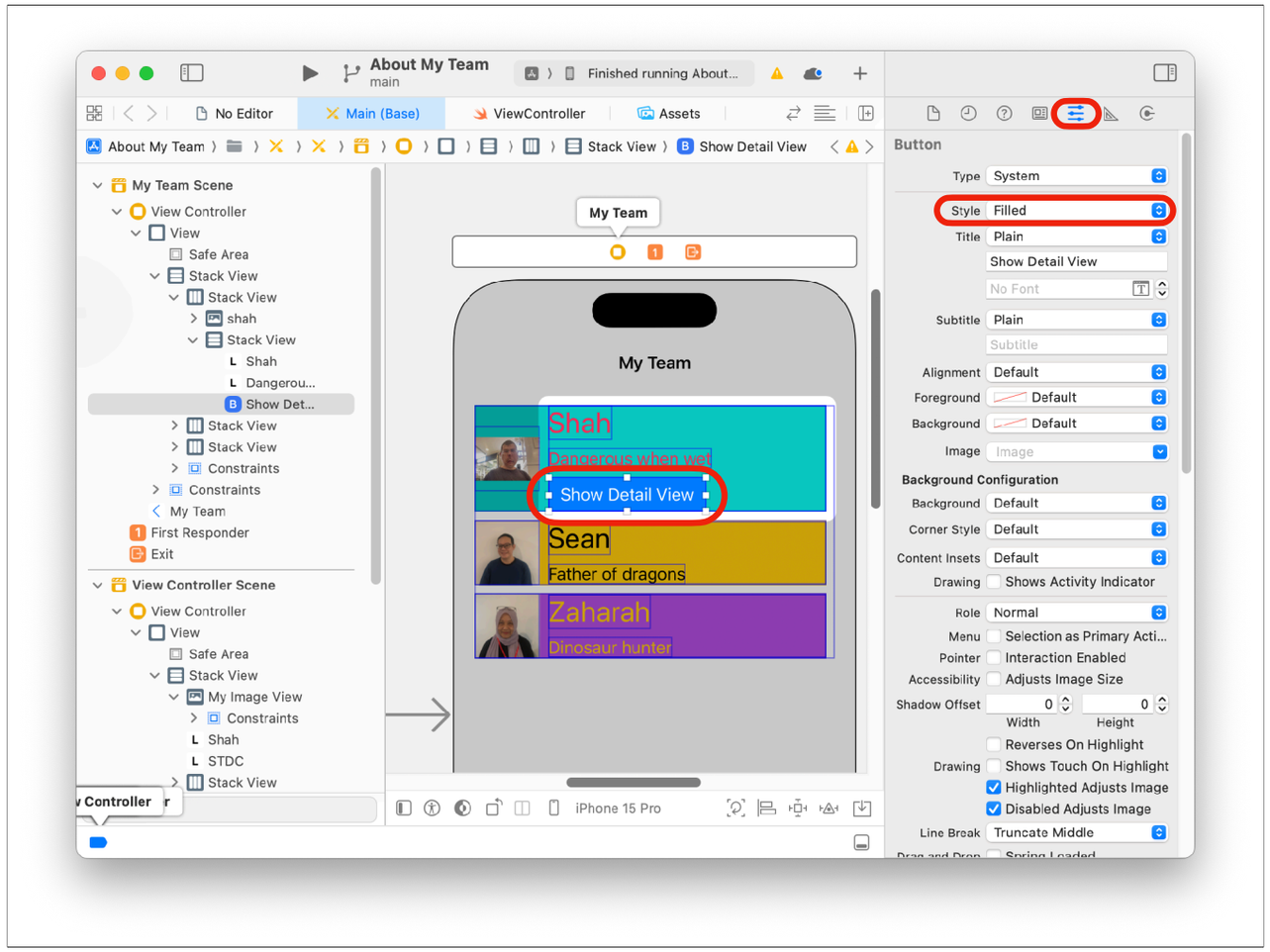
Note the Navigation Controller object is now in Main.storyboard.



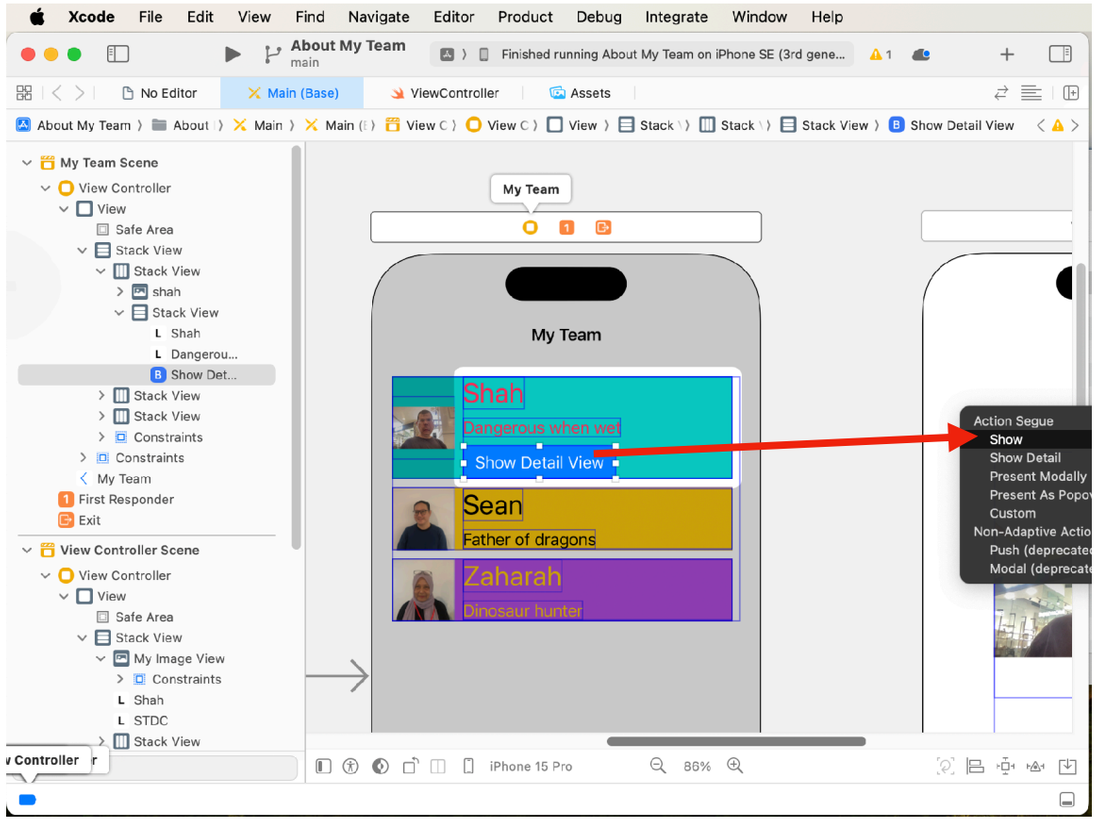
Change the title of the Navigation Item to My Team.



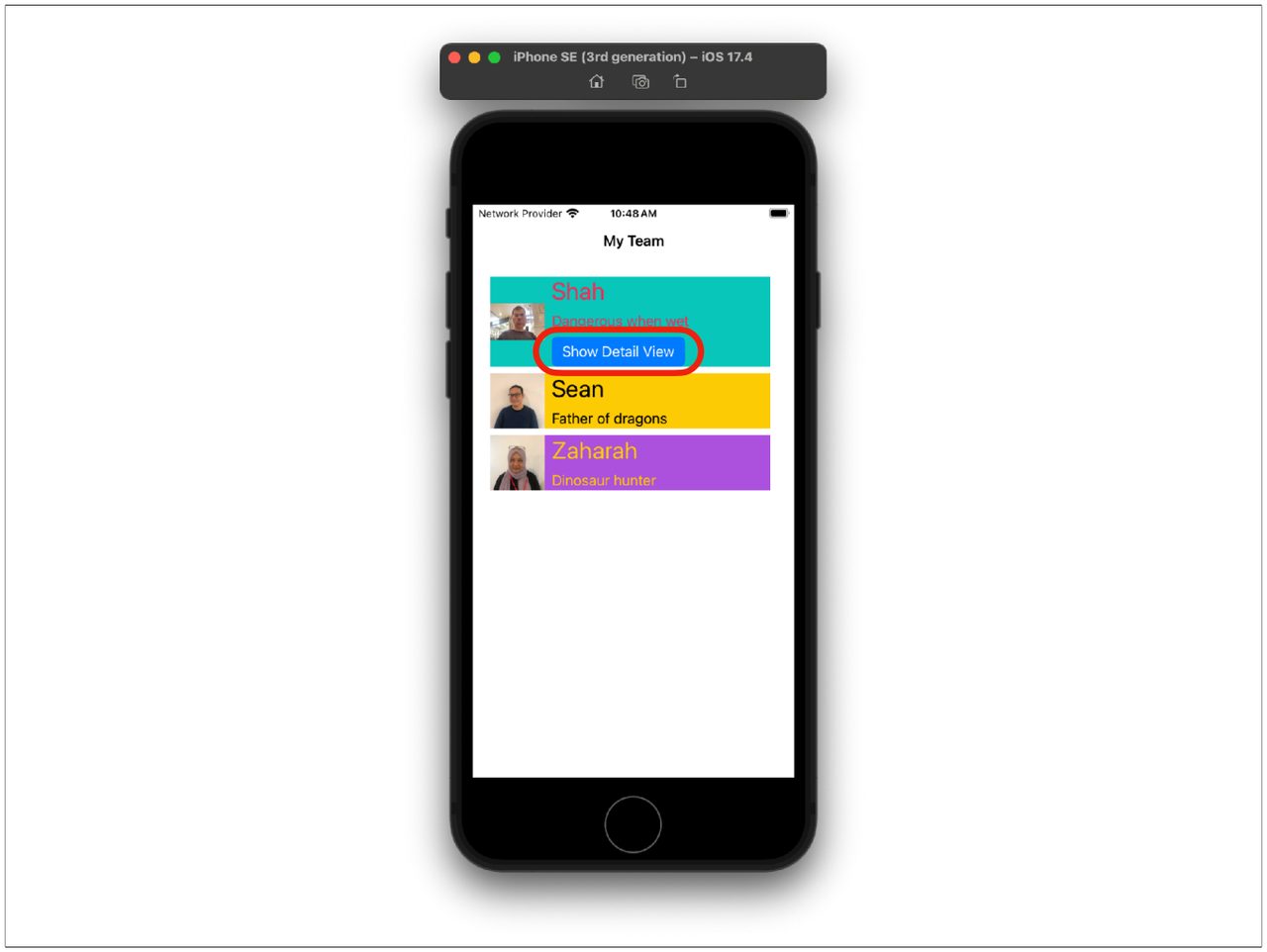
Add a button to the first child stack view.



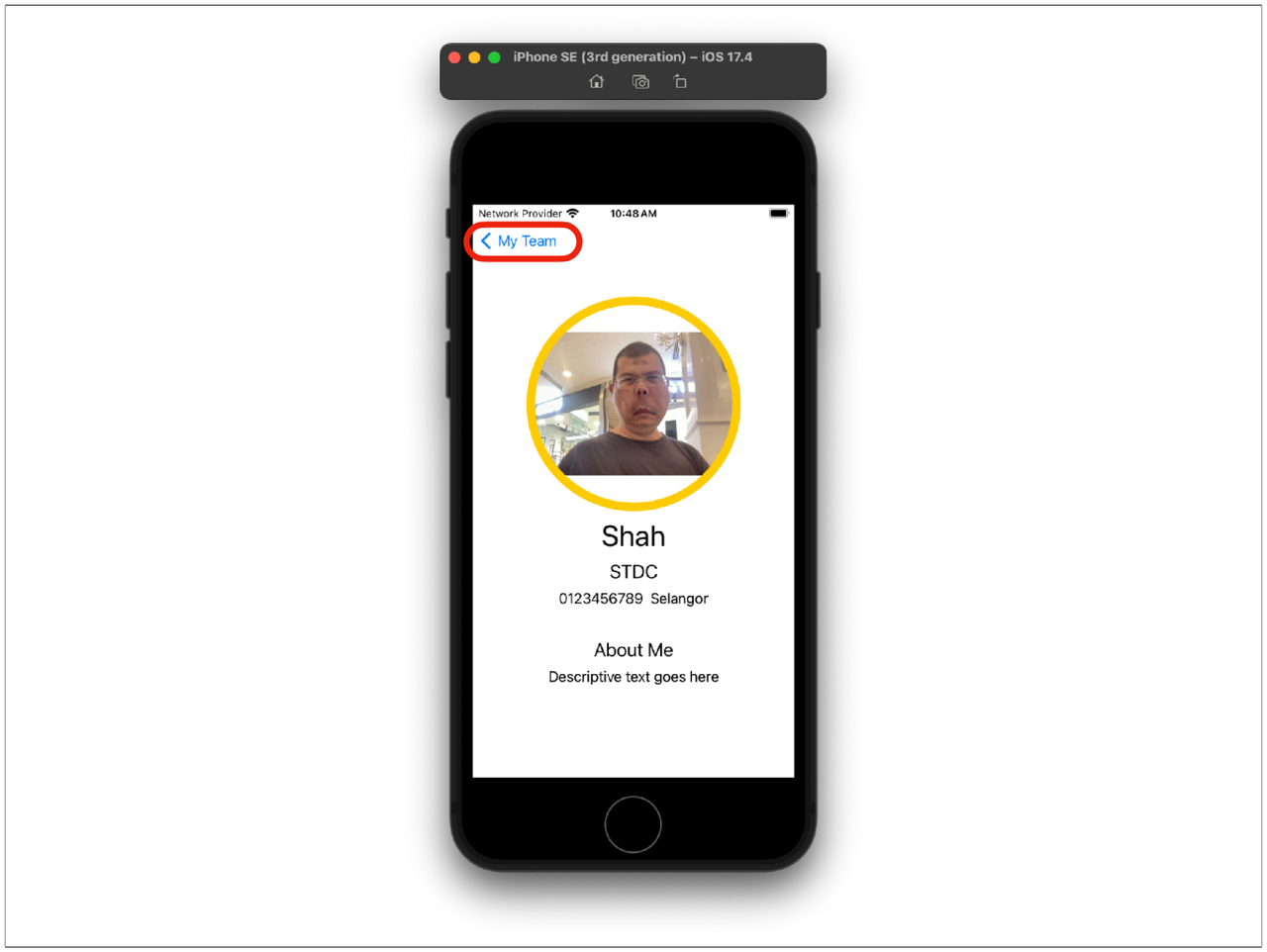
In the Attributes inspector, change the button style to Filled.



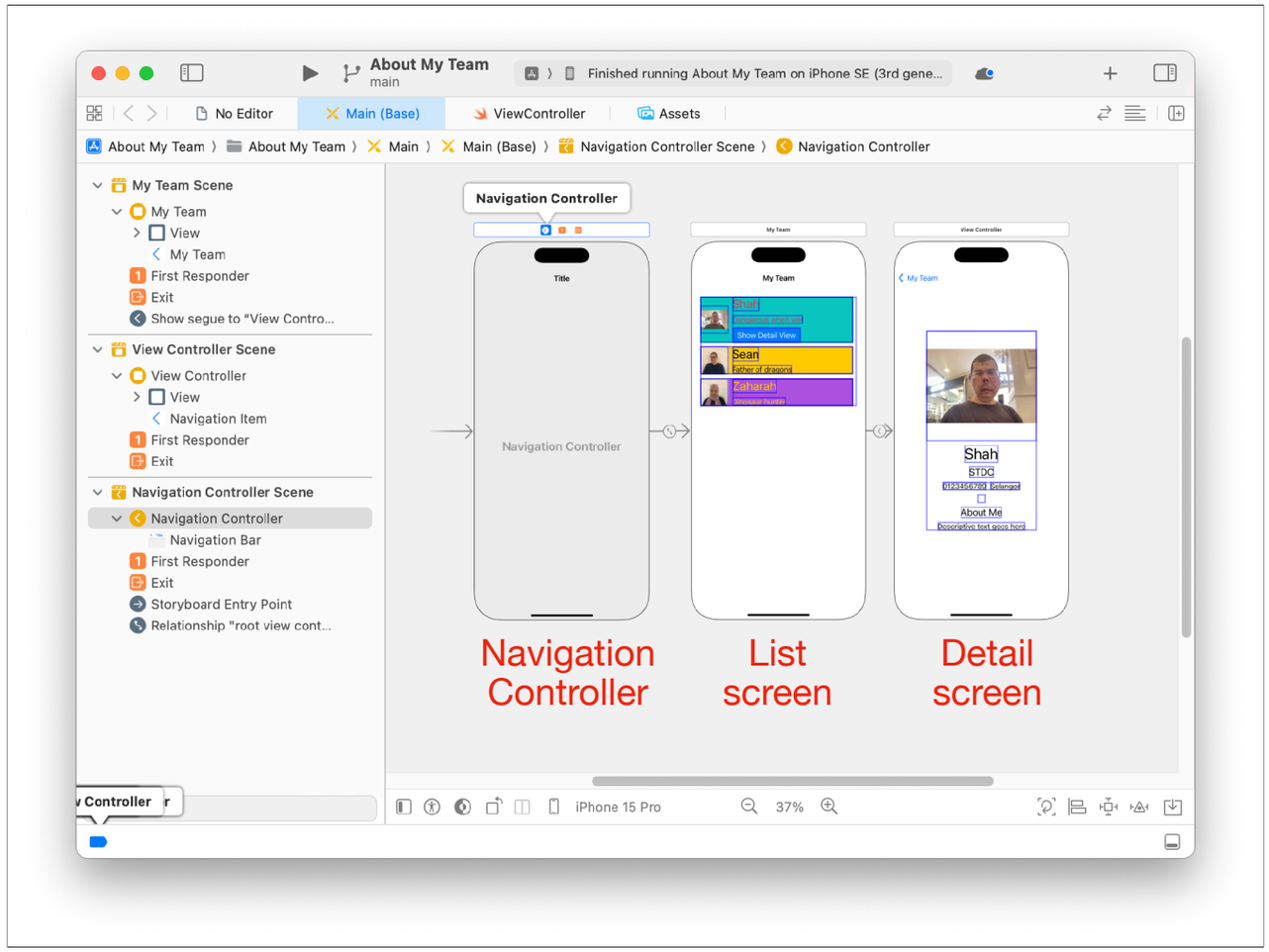
Control + Drag from the button to the original view controller scene, and choose Show.



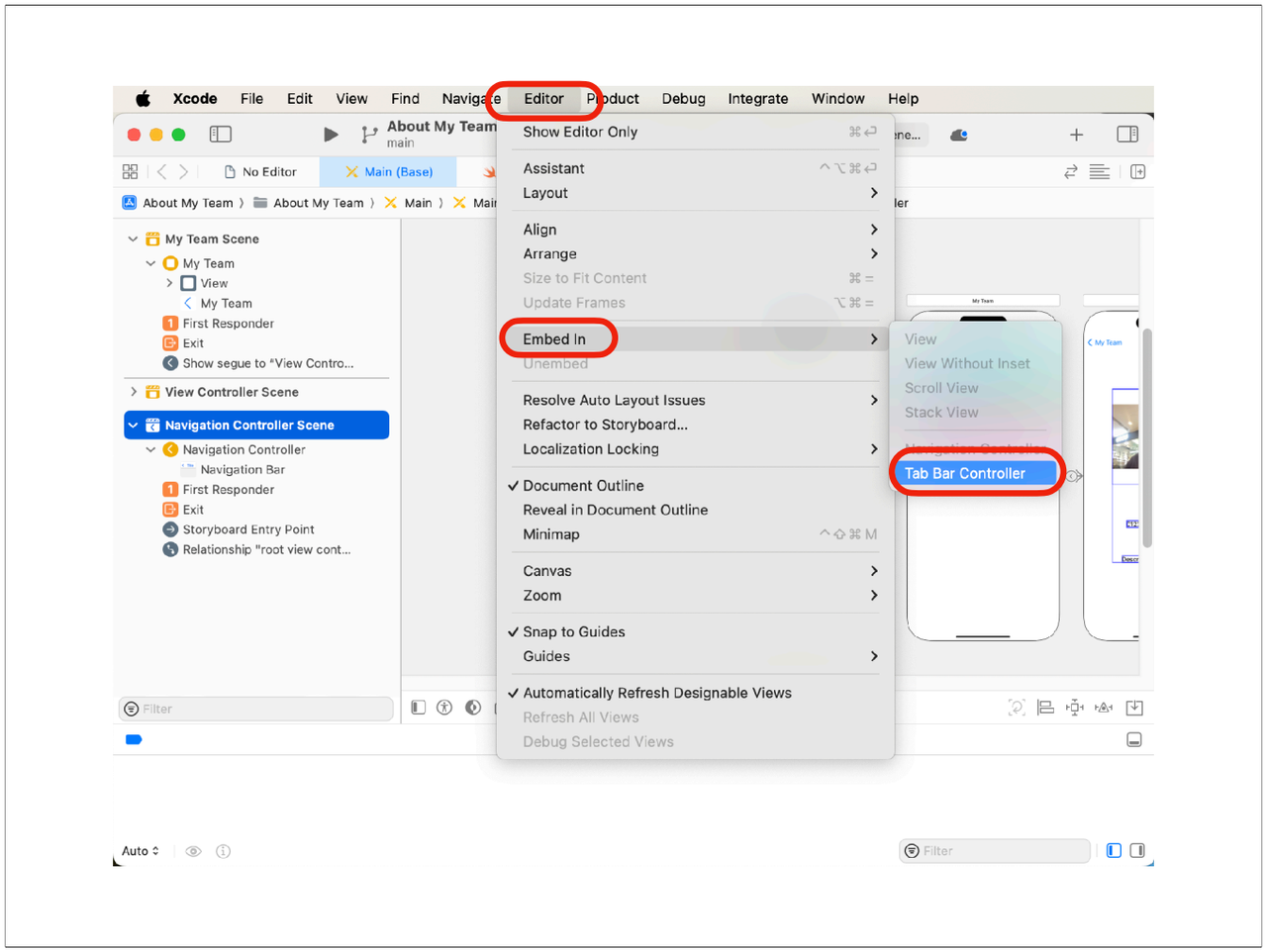
Run your app in Simulator. Click the button.



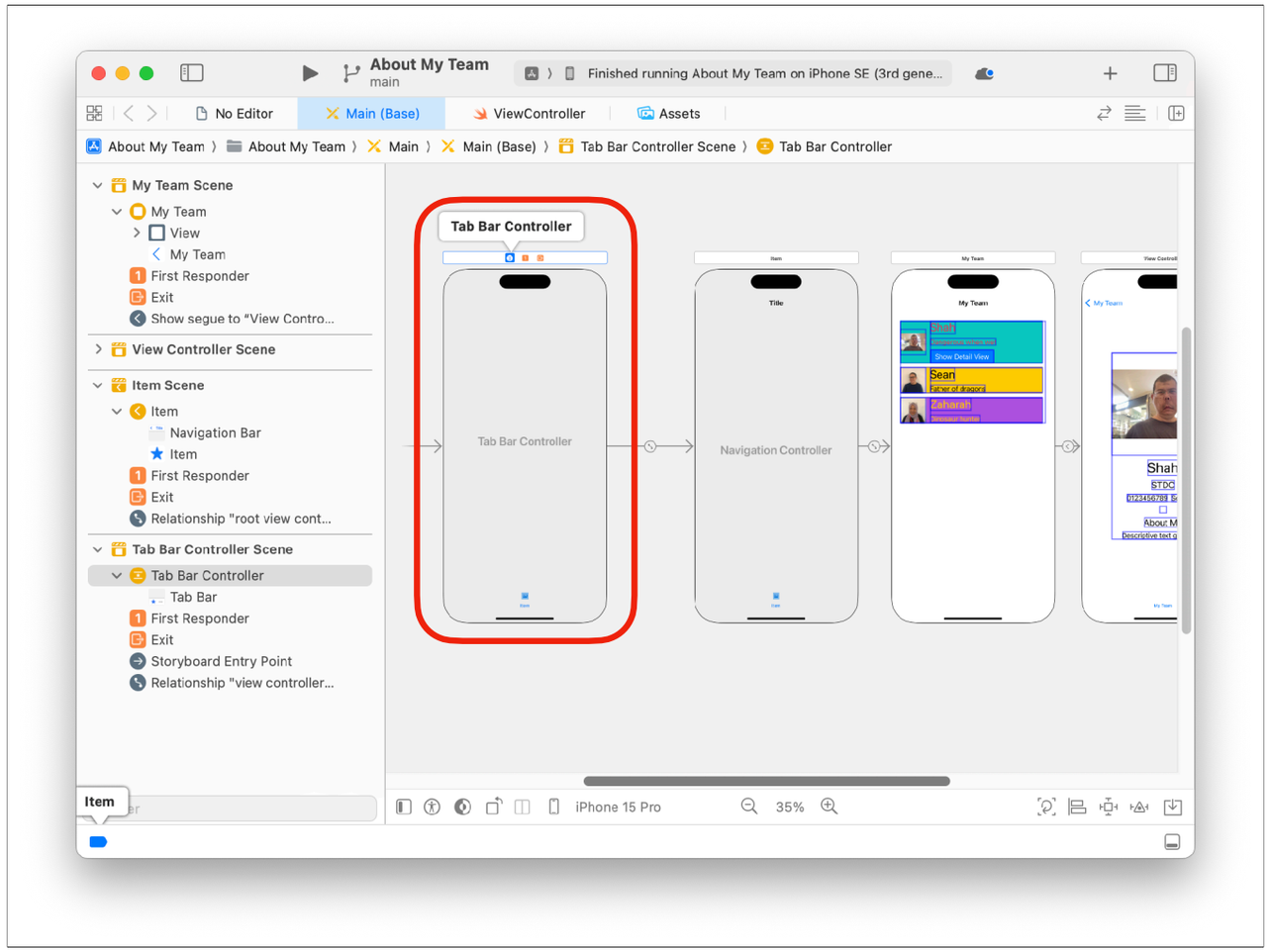
The detail screen appears. Click on the < My Team button to return to the first screen.



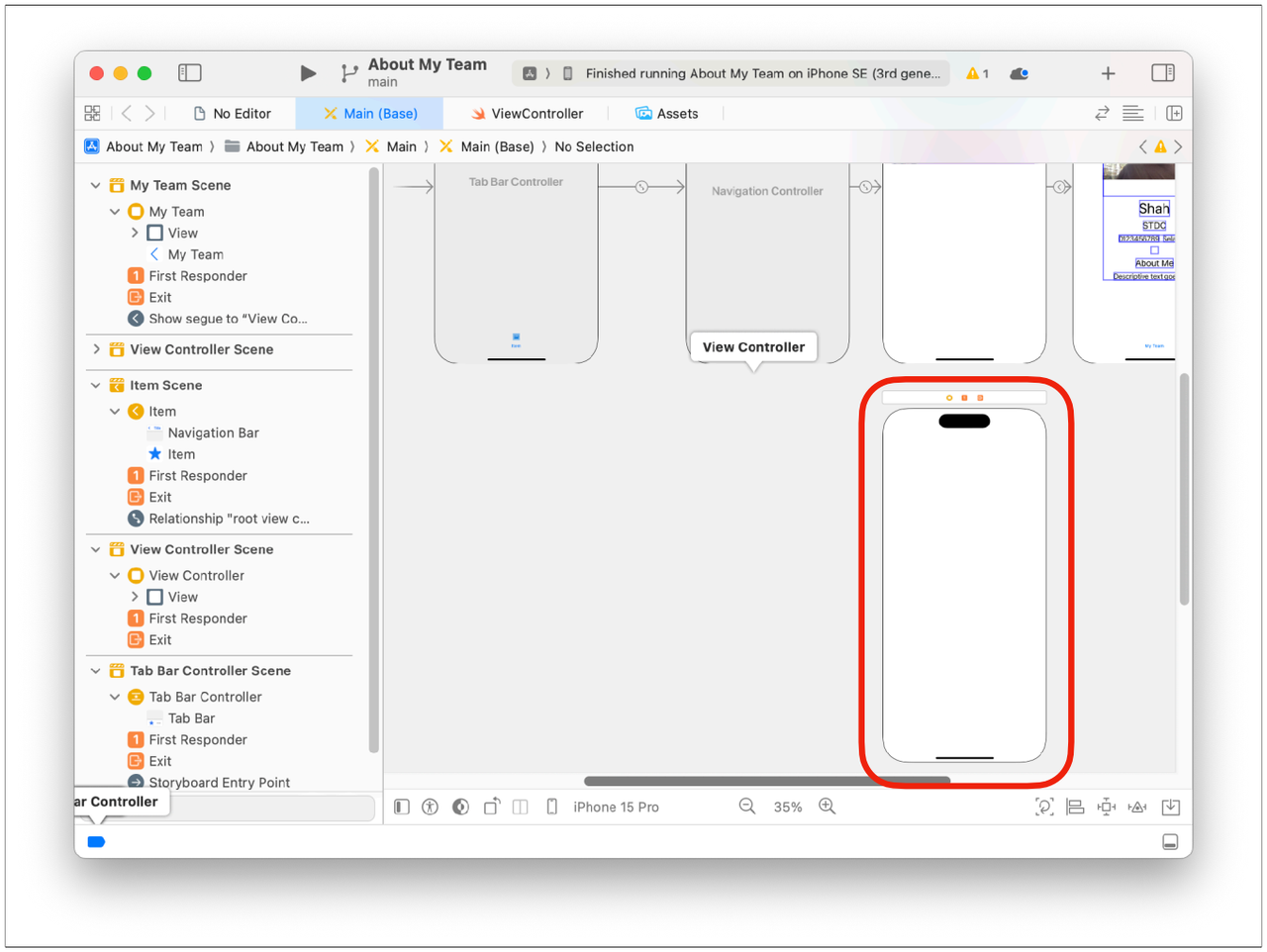
To avoid confusion, the screens will be referred to as follows: Navigation Controller, List screen and Detail screen.



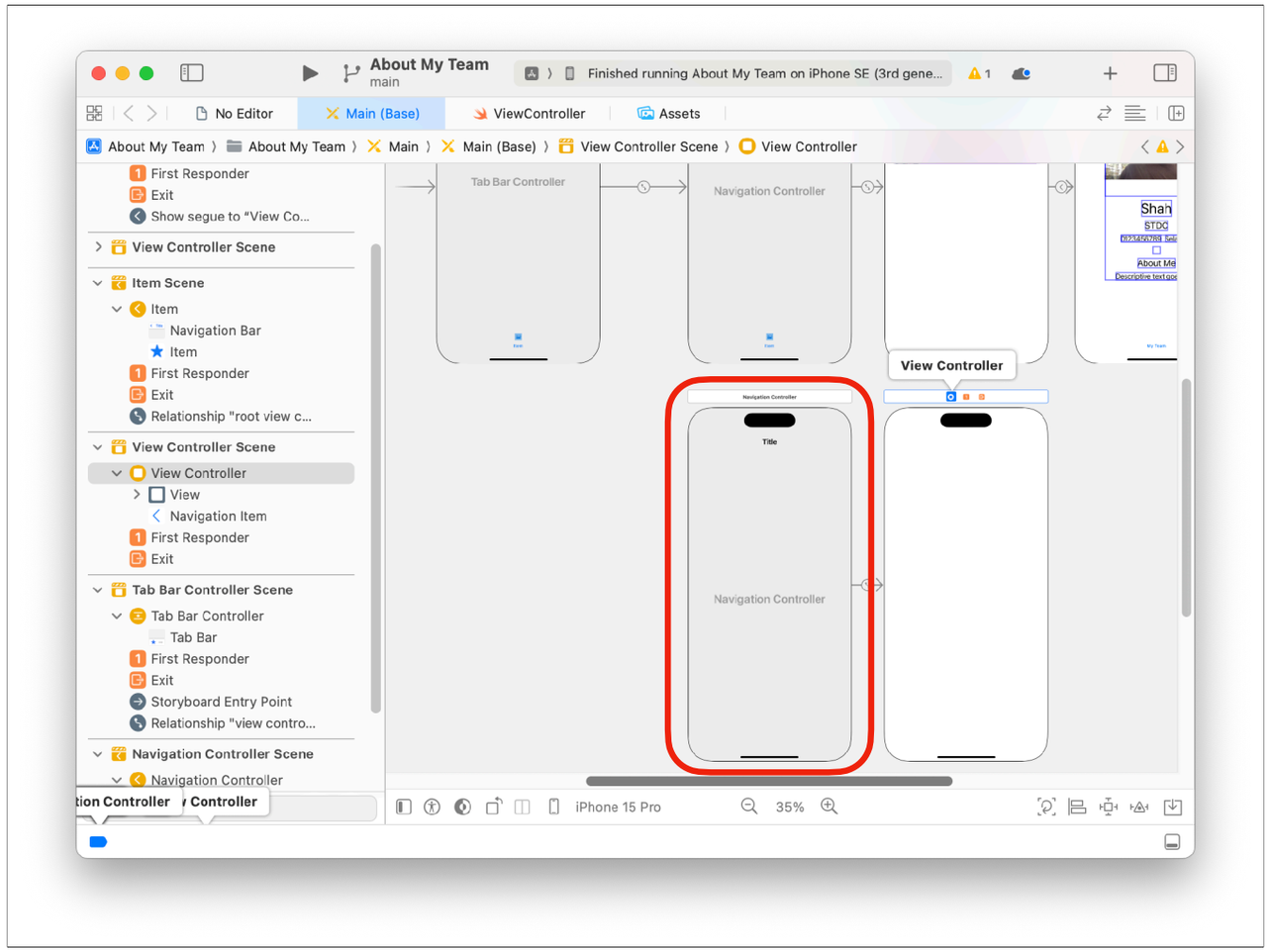
Embed the navigation controller in a tab bar controller.



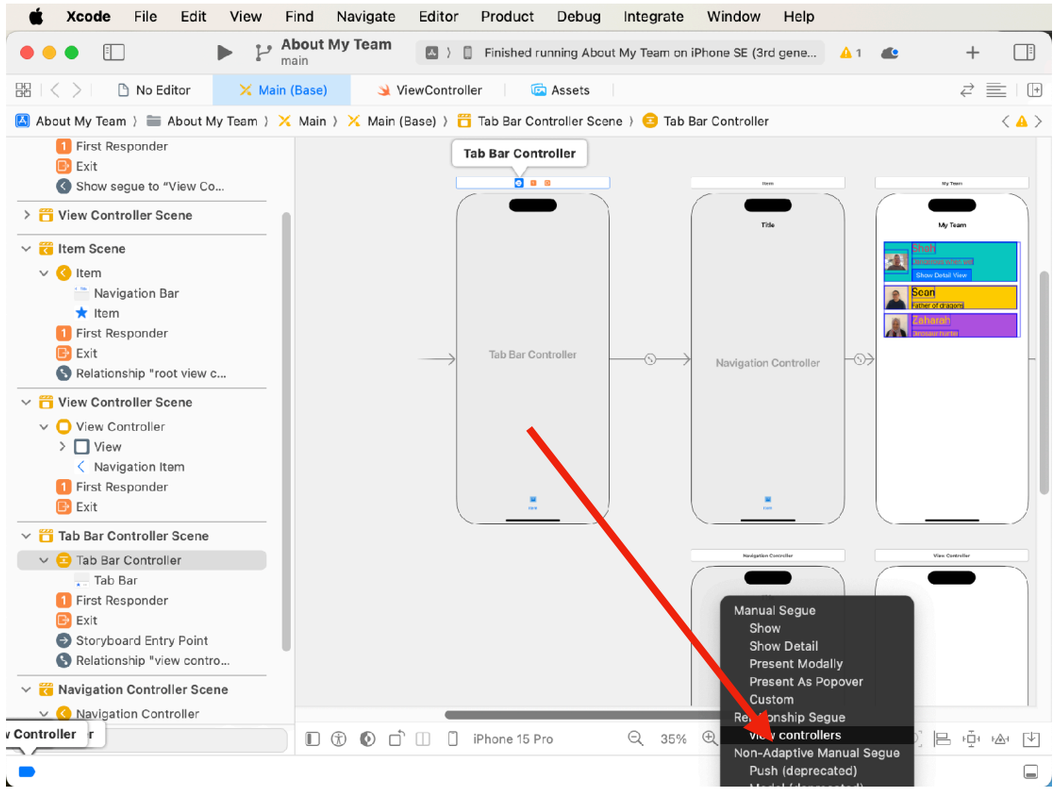
The tab bar controller appears in Main.storyboard.



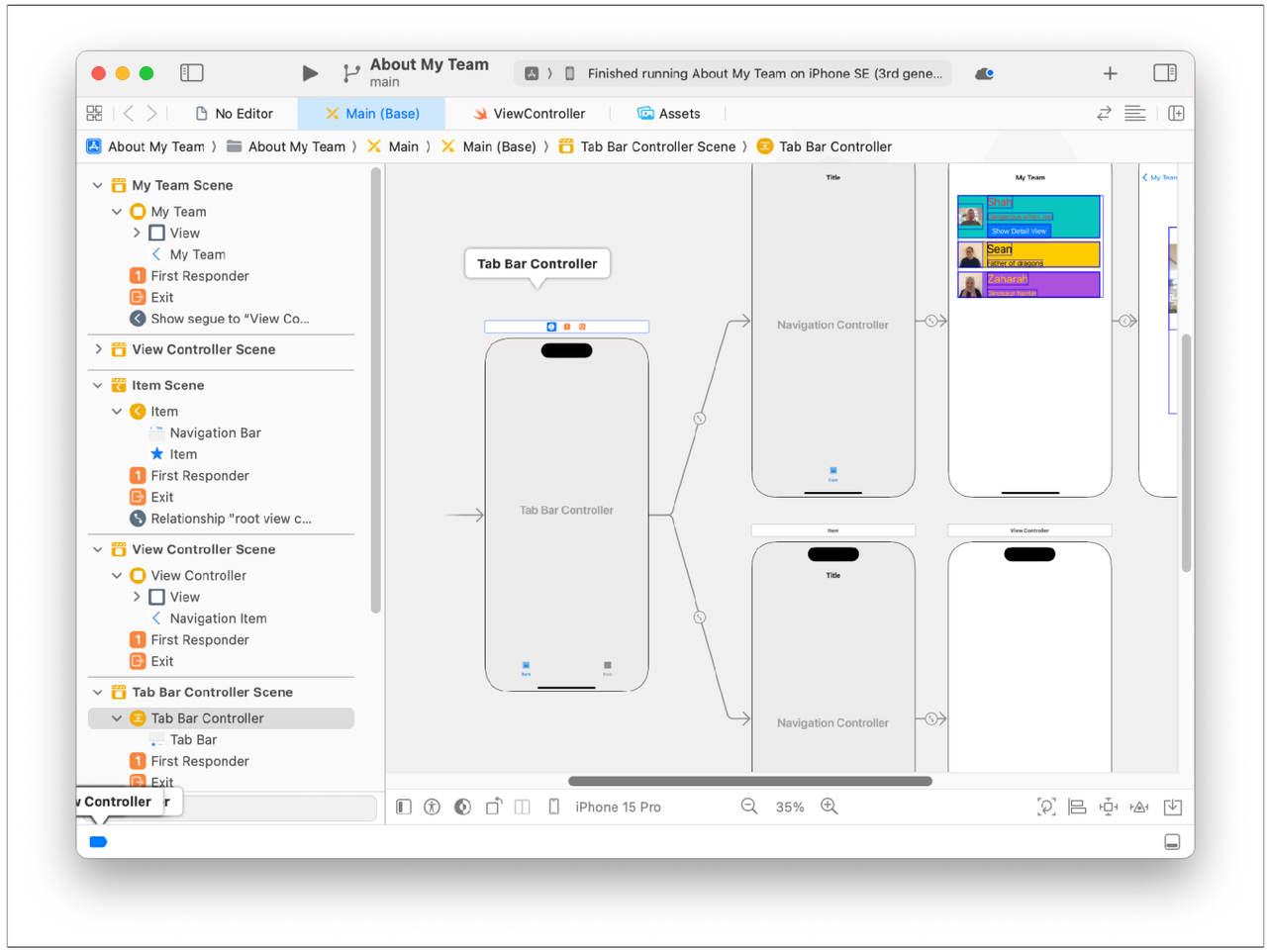
Add a new View Controller object to Main.storyboard as shown.



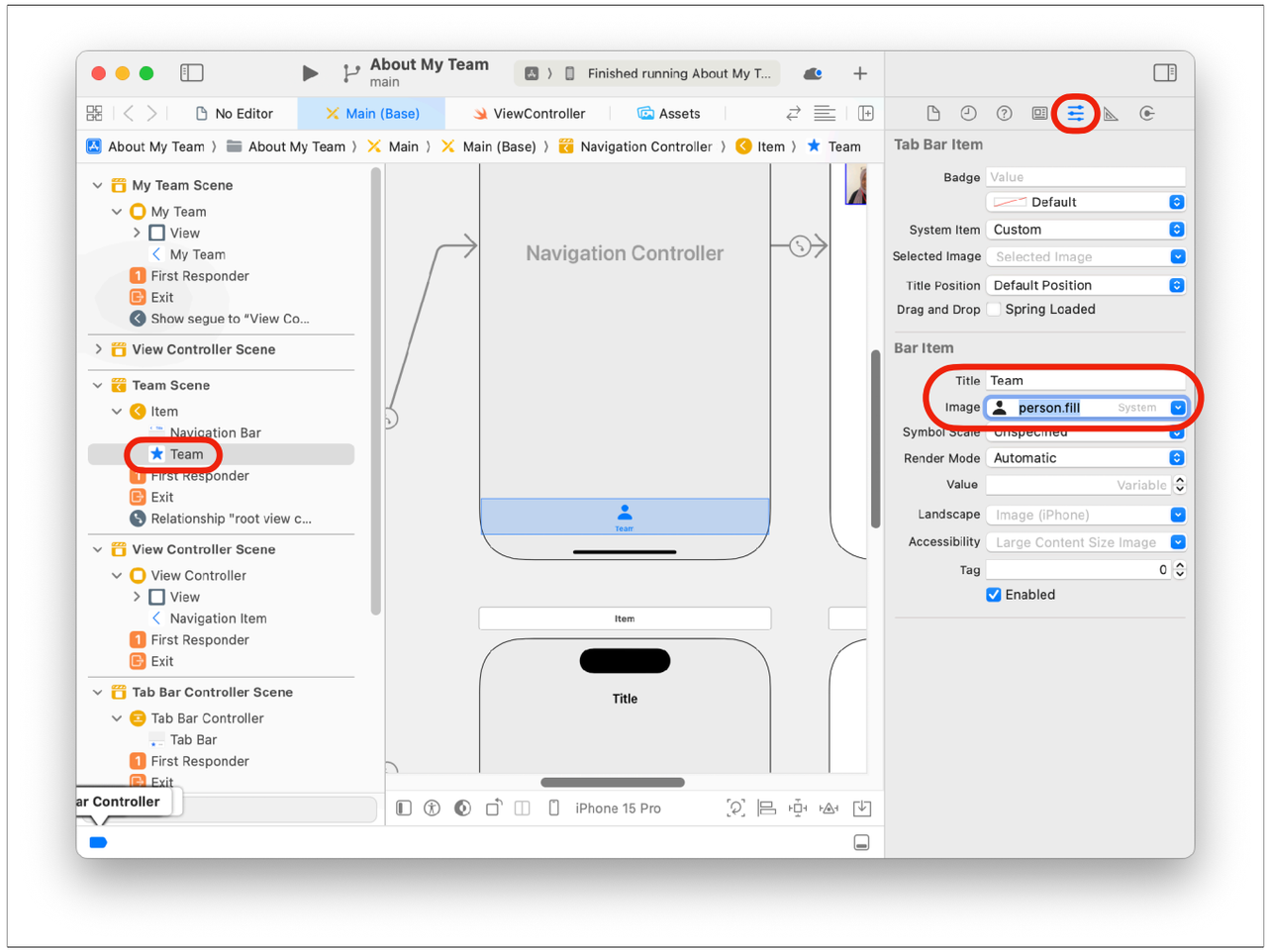
Embed it in a navigation controller.



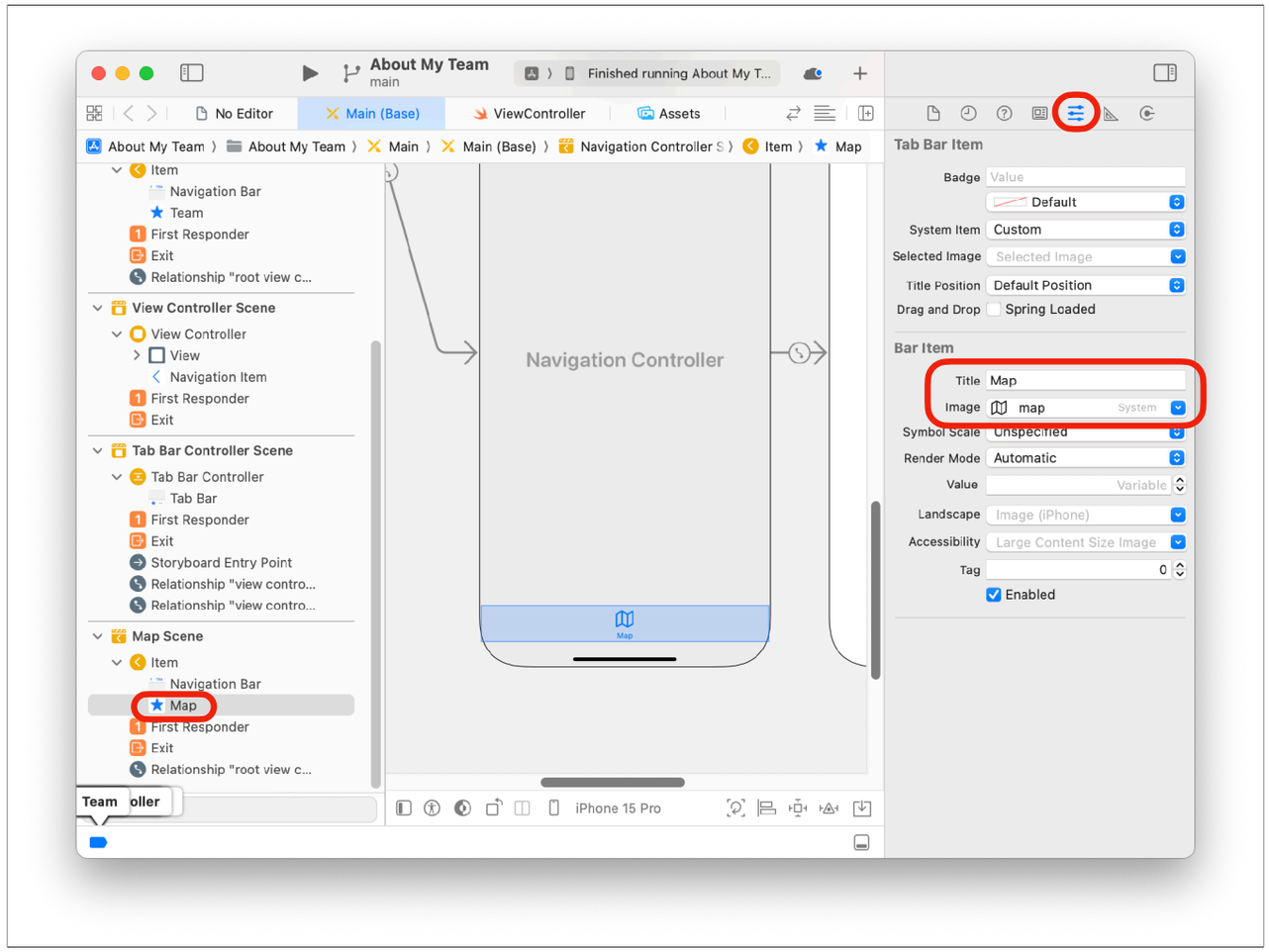
Control + Drag from the tab bar controller to the navigation controller, and choose view controllers from the pop-up menu.



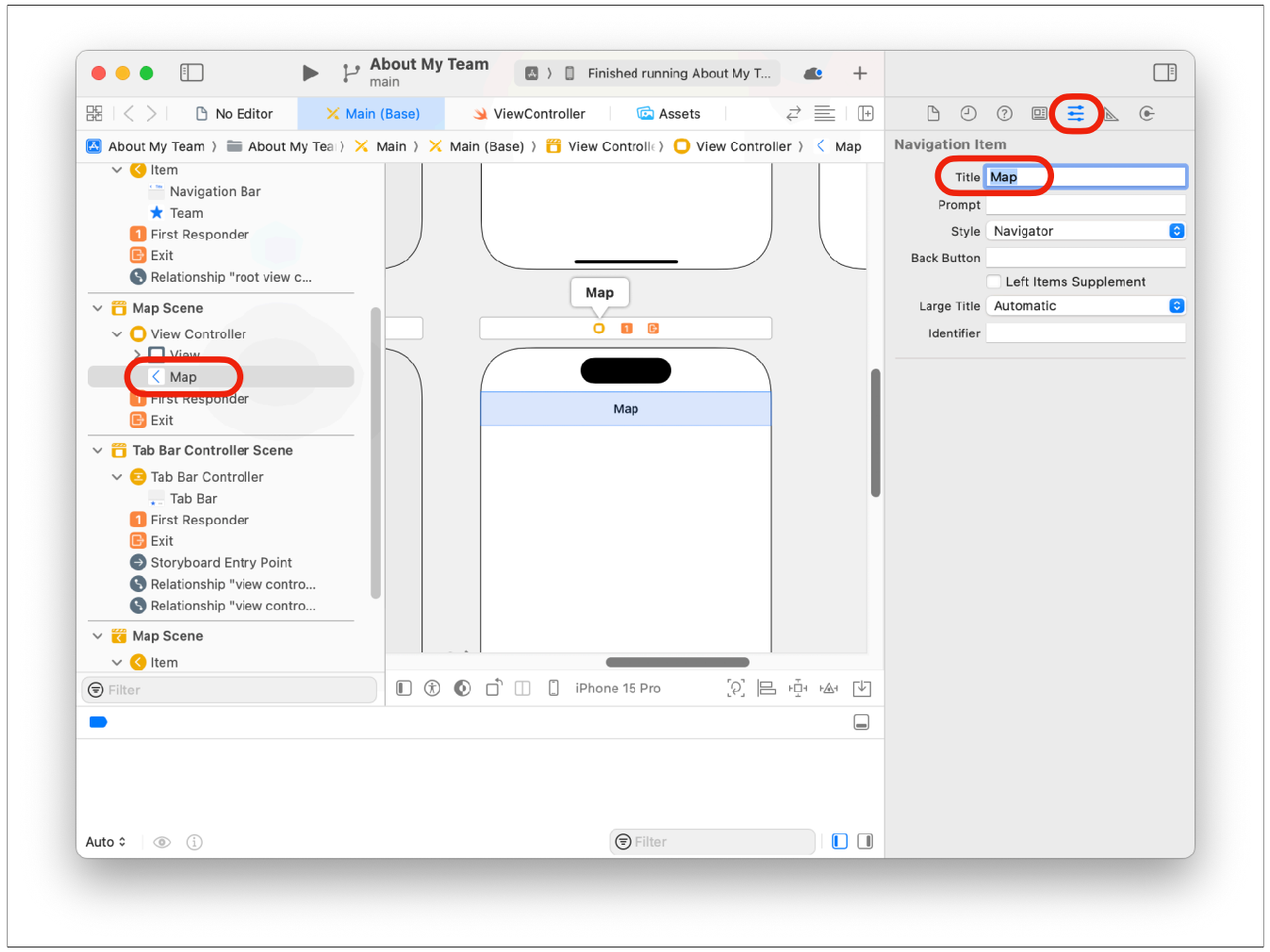
Rearrange the contents of Main.storyboard as shown.



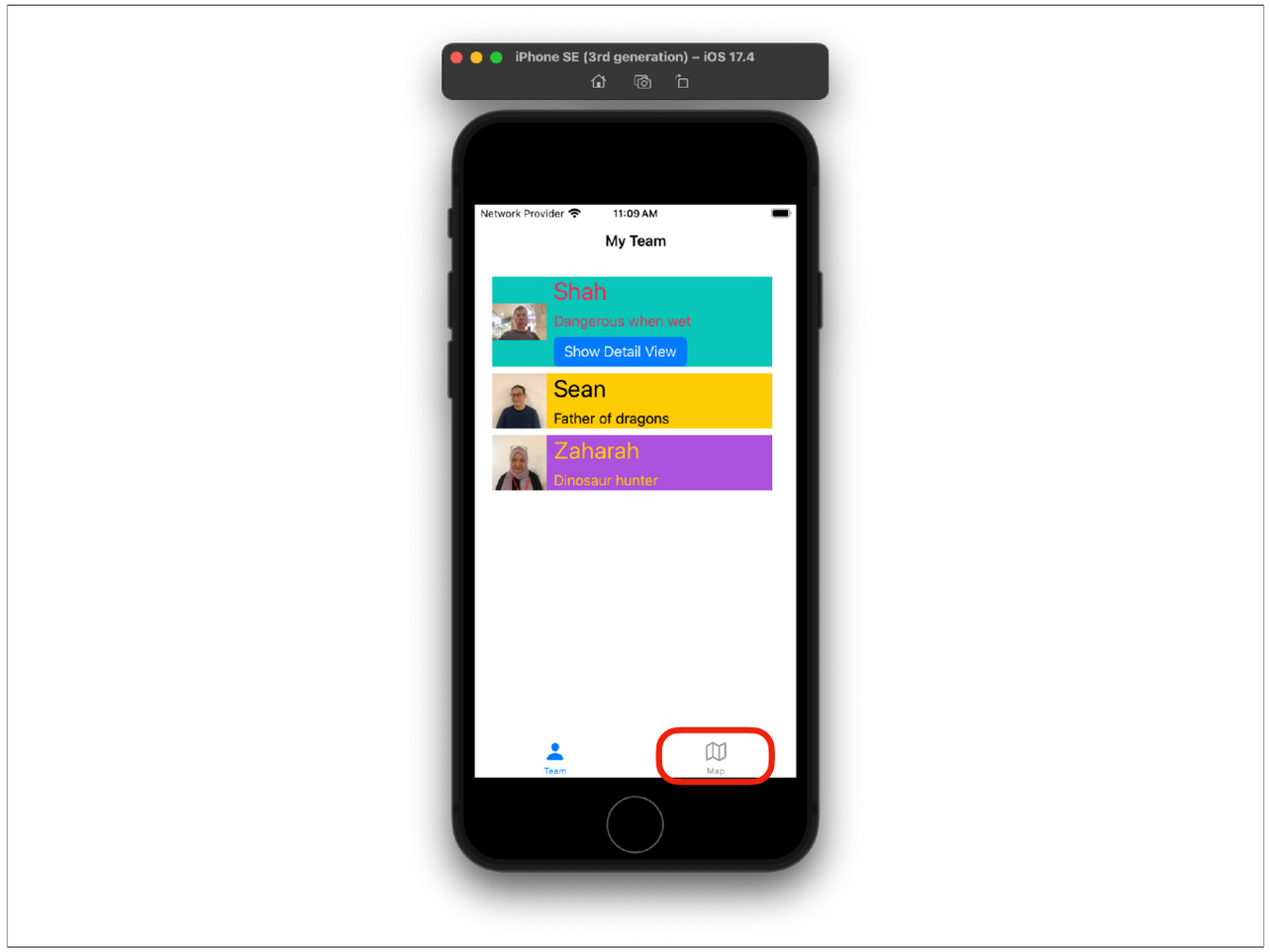
For the first navigation controller, change the tab bar title and image to Team and person.fill respectively.



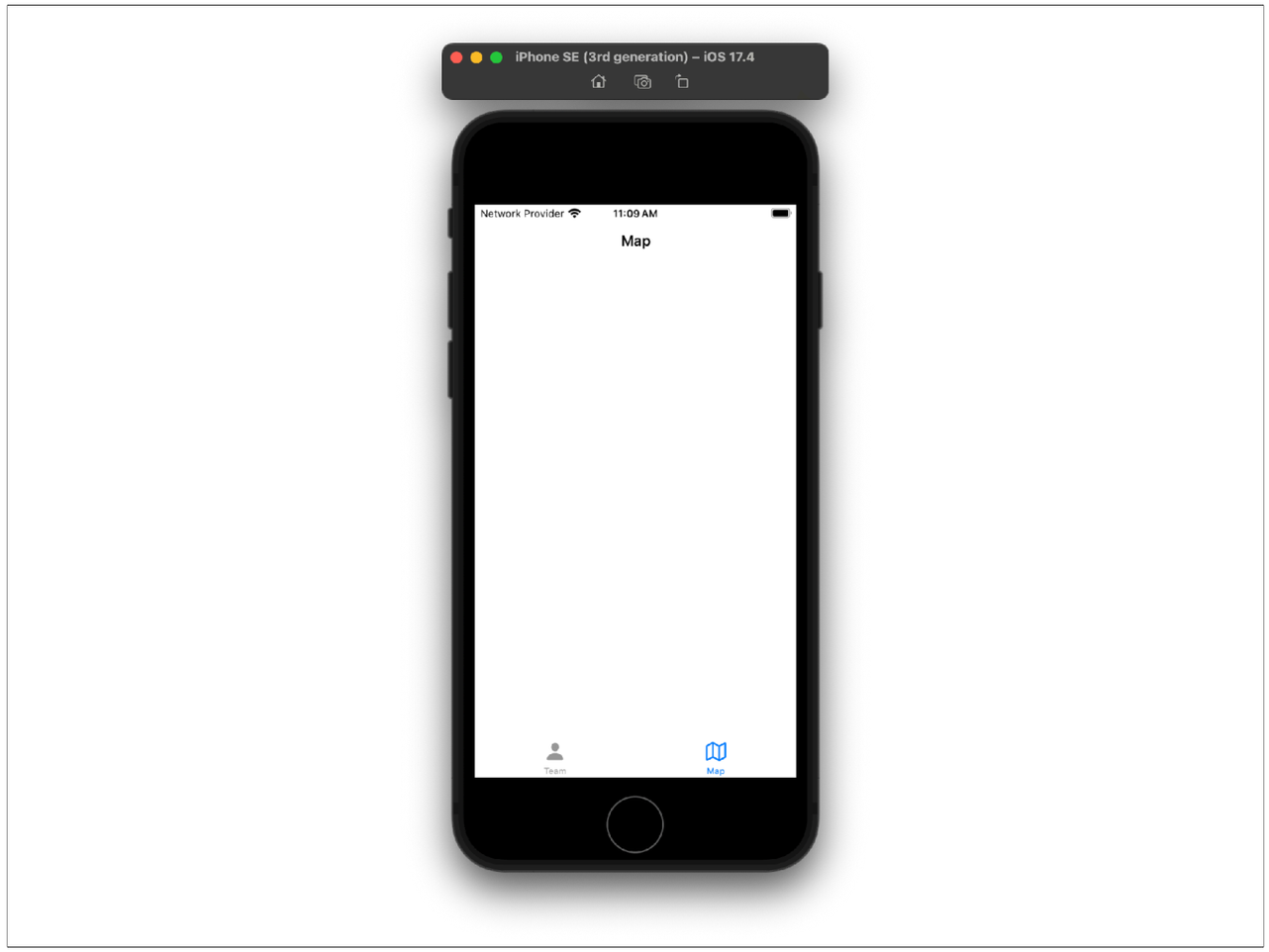
For the second navigation controller, change the tab bar title and image to Map and map respectively.



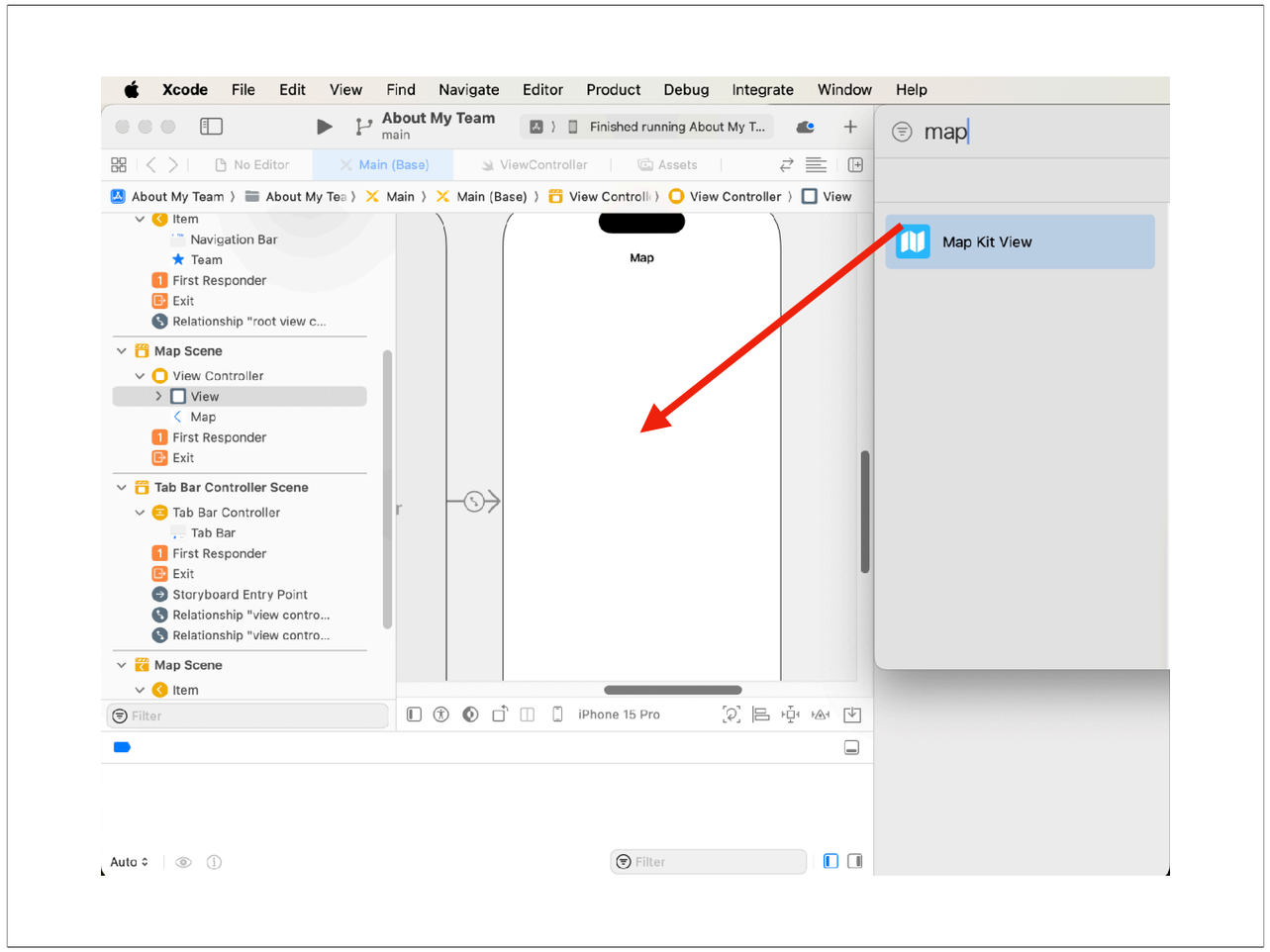
Change the navigation item title for the screen that will show the map to Map.



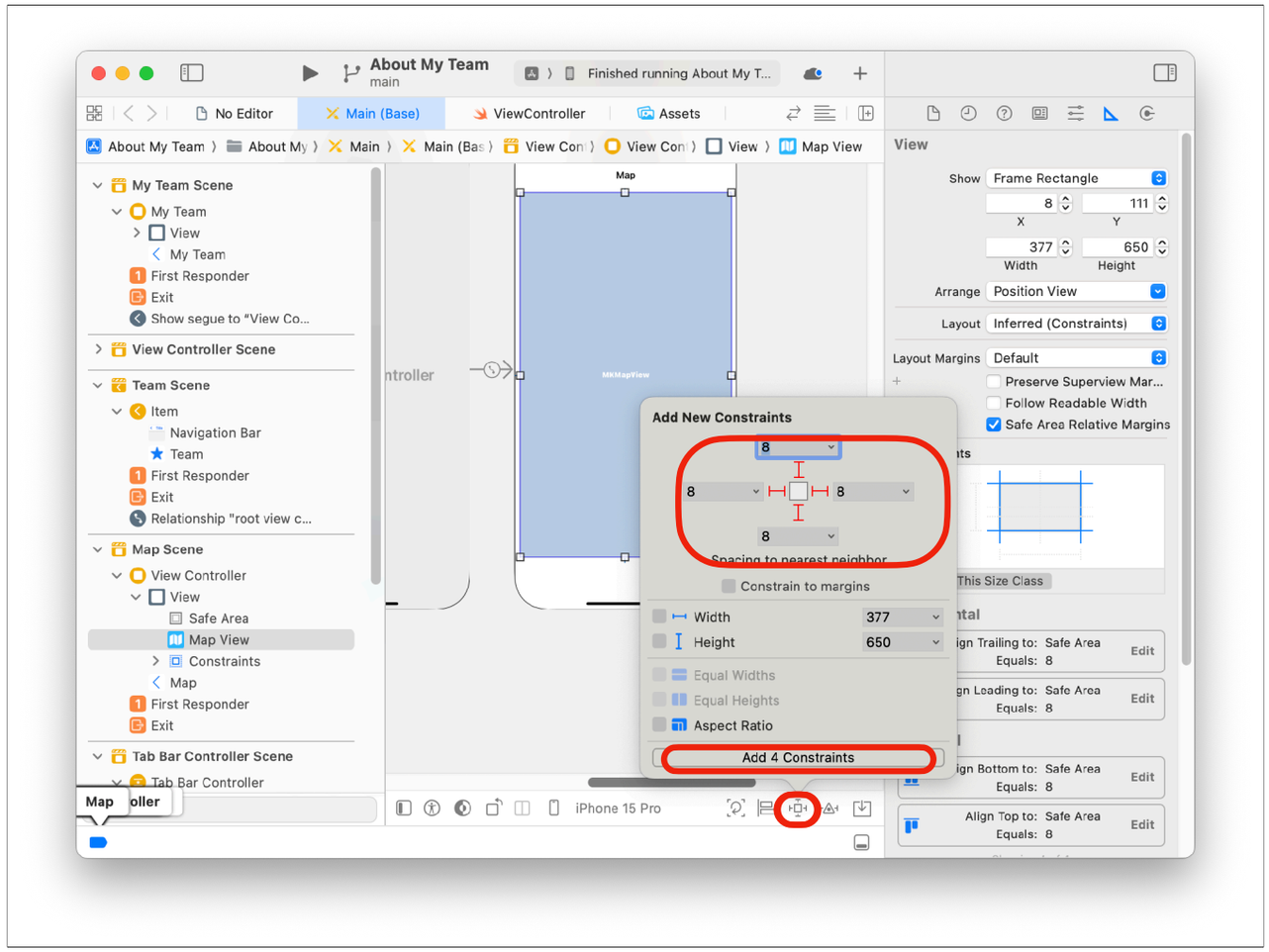
Build and run your app in Simulator. You should see two tabs at the bottom. Click the Map tab.



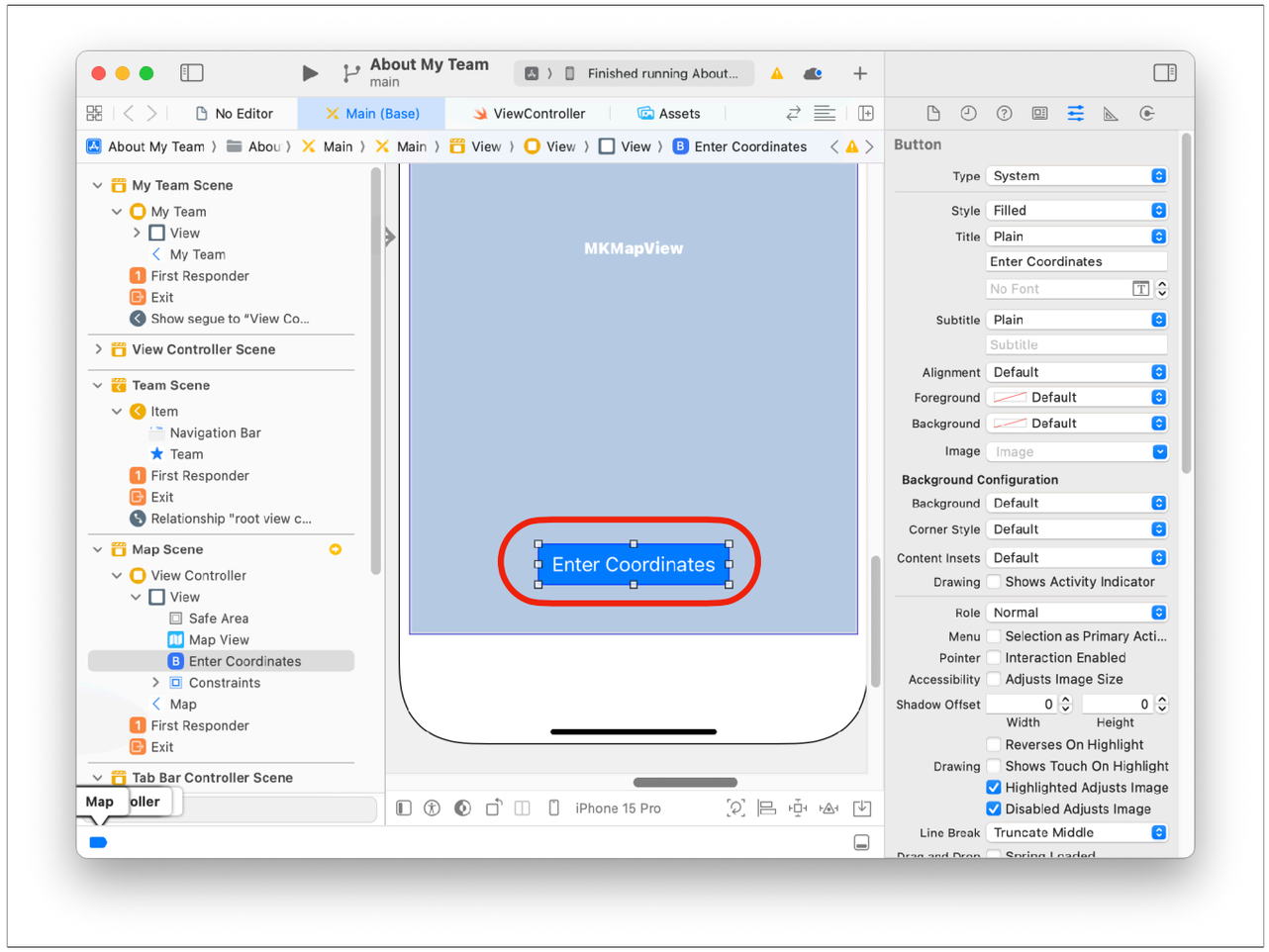
The screen that will show a map is displayed.



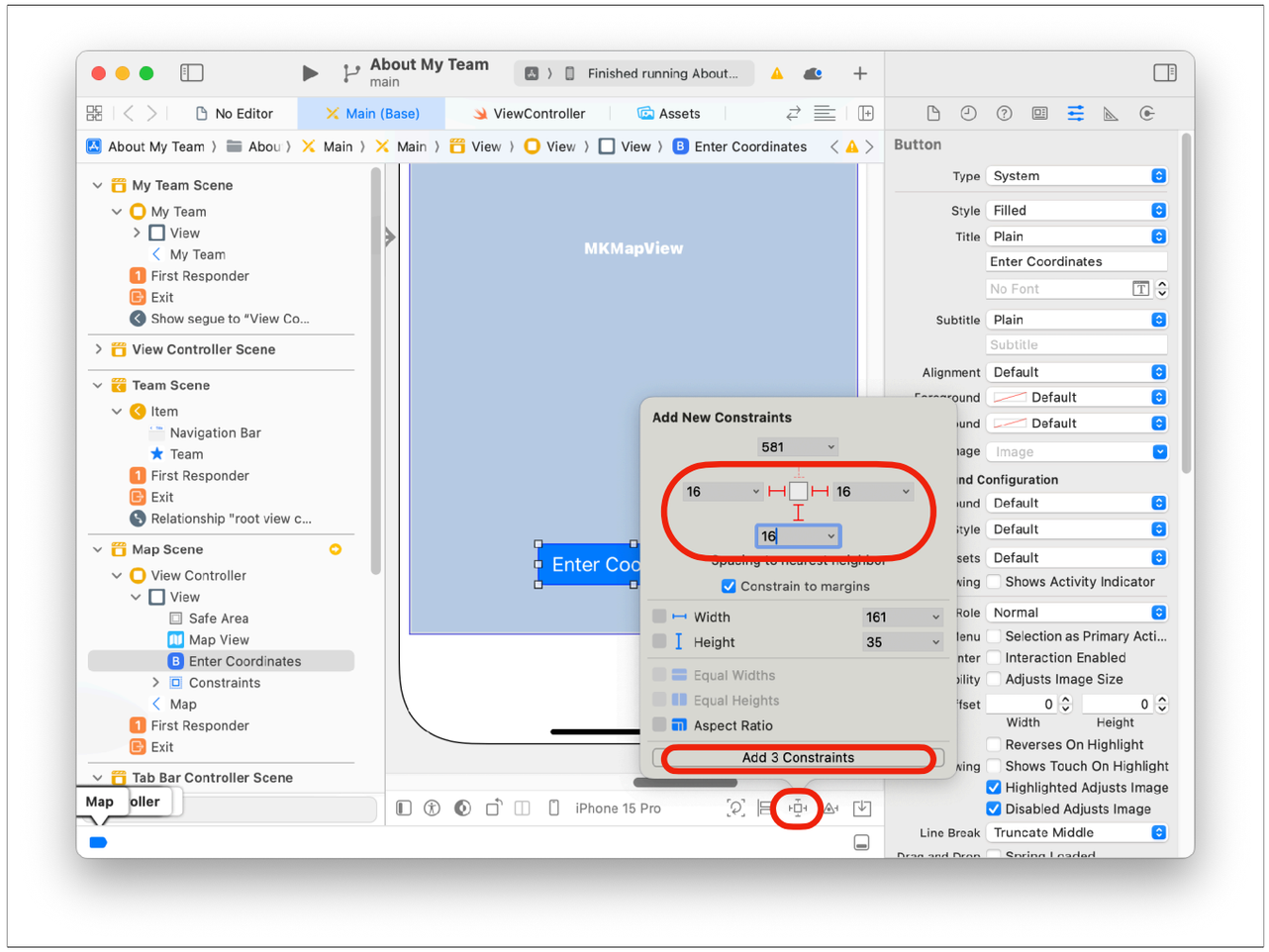
Open the Library and drag a Map Kit View to the screen that will show a map.



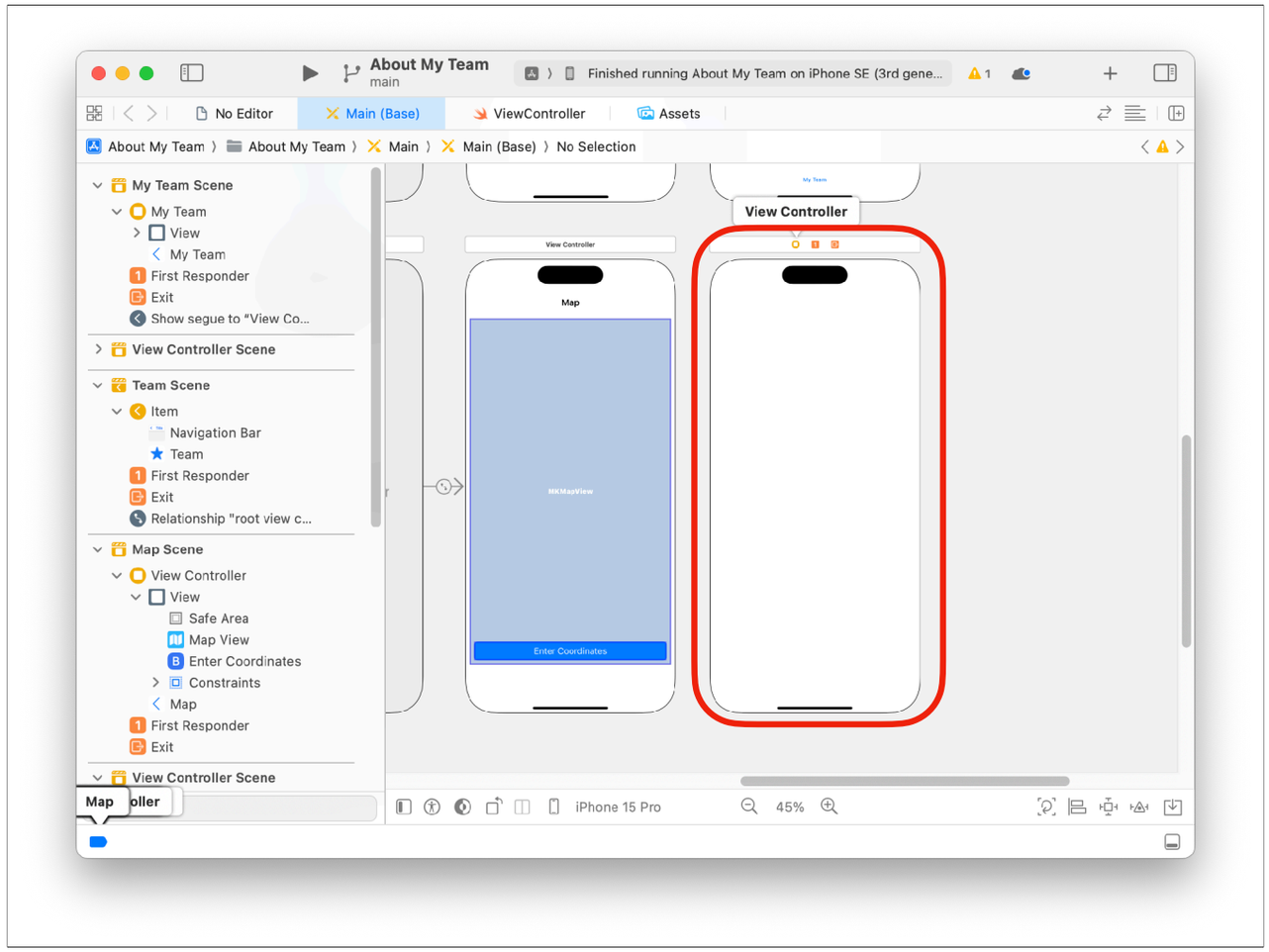
Click the Auto Layout Add New Constraints button and set all the constraints to 8.



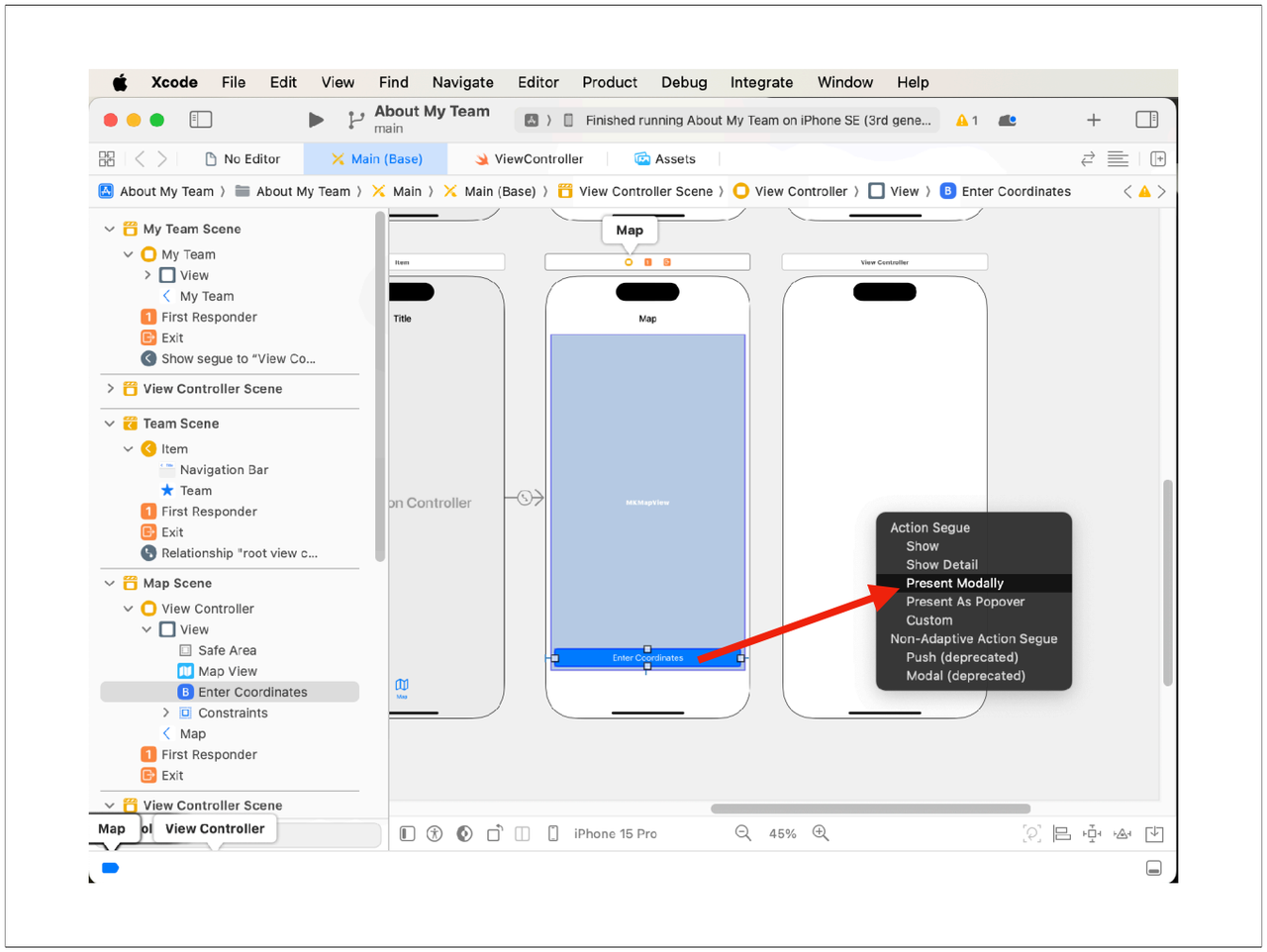
Drag in a filled button from the Library and change the button title to Enter Coordinates.



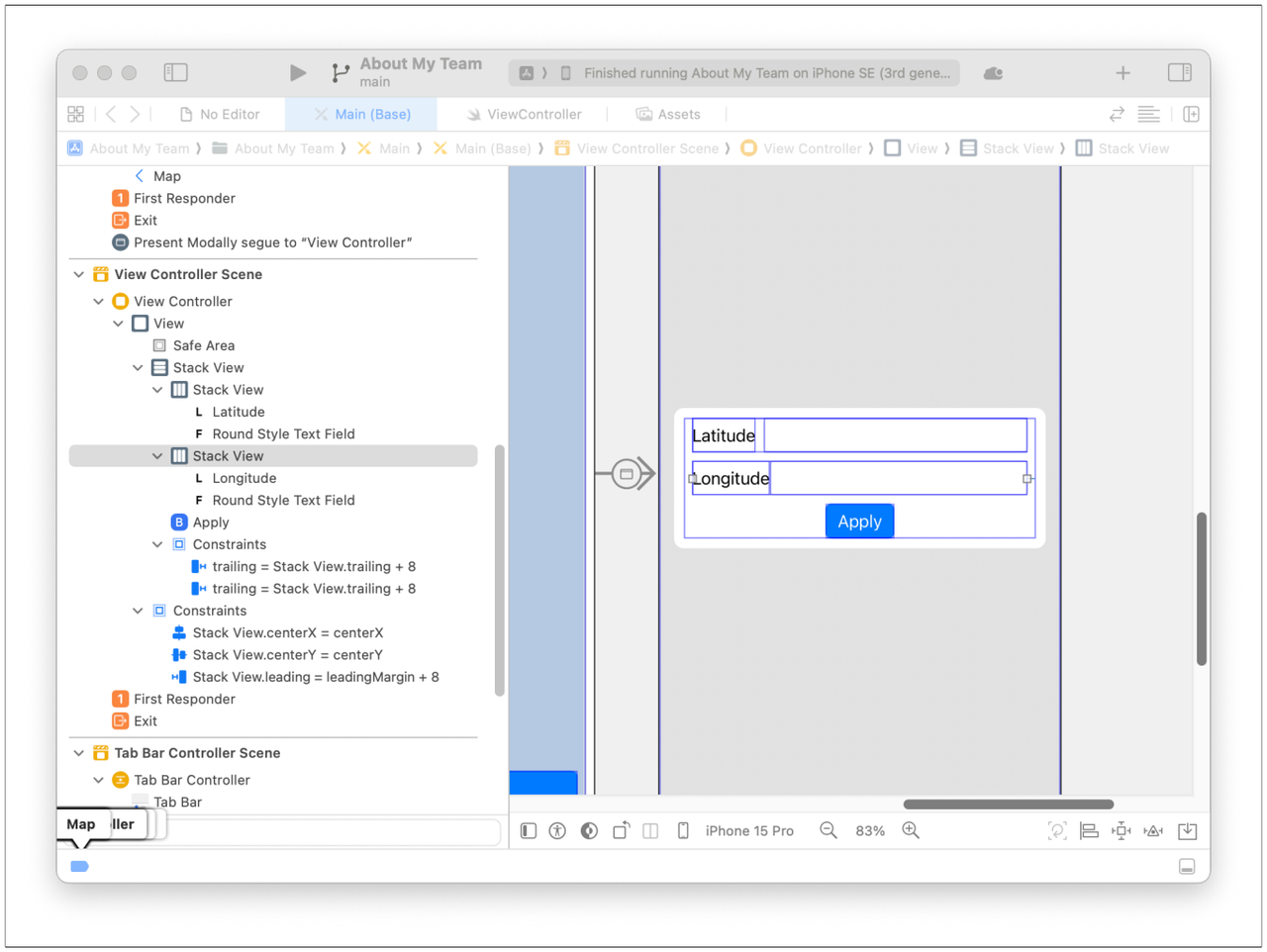
Add constraints to the button as shown.



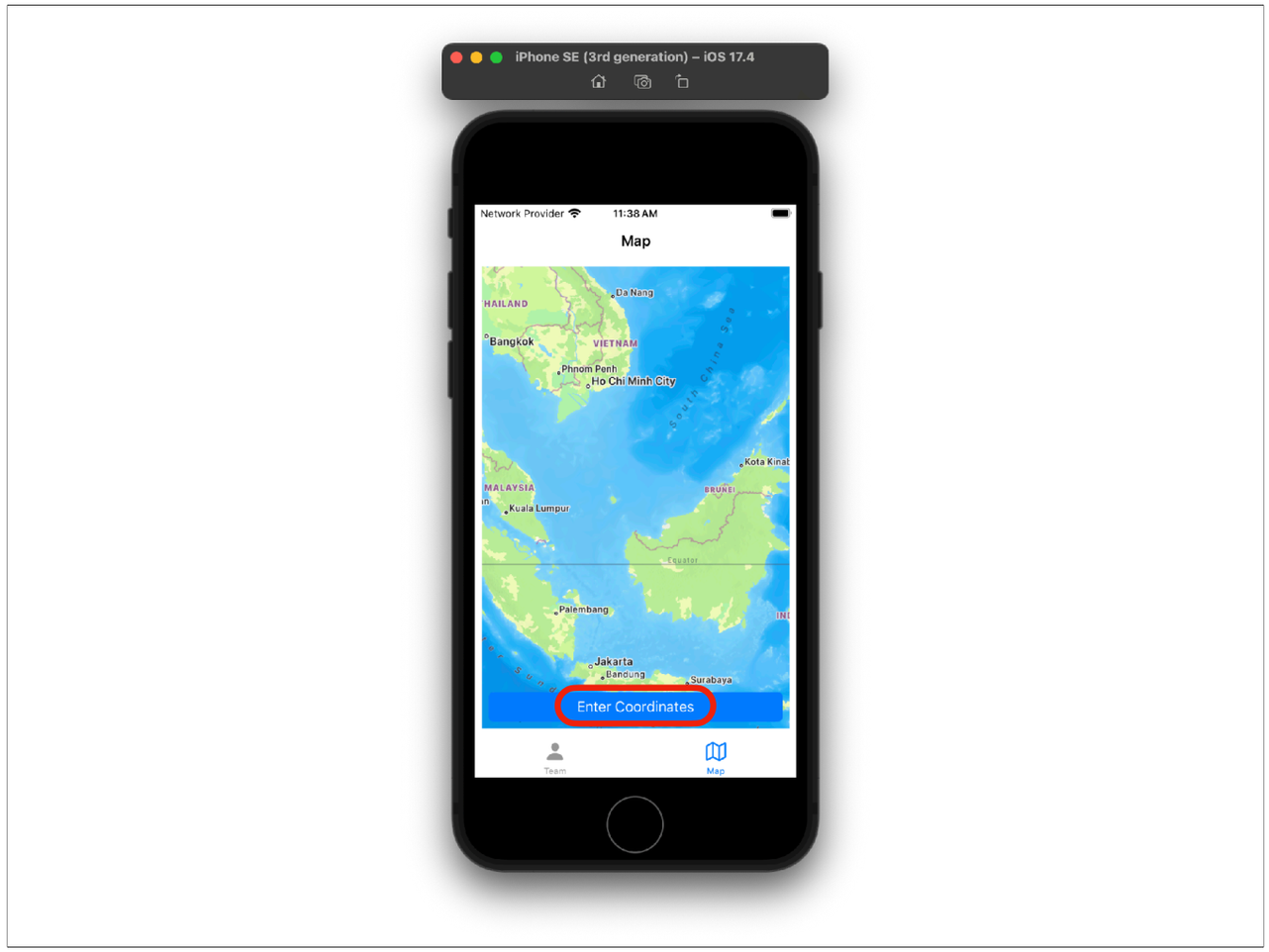
Drag in another view controller object from the Library and place it next to the map screen.



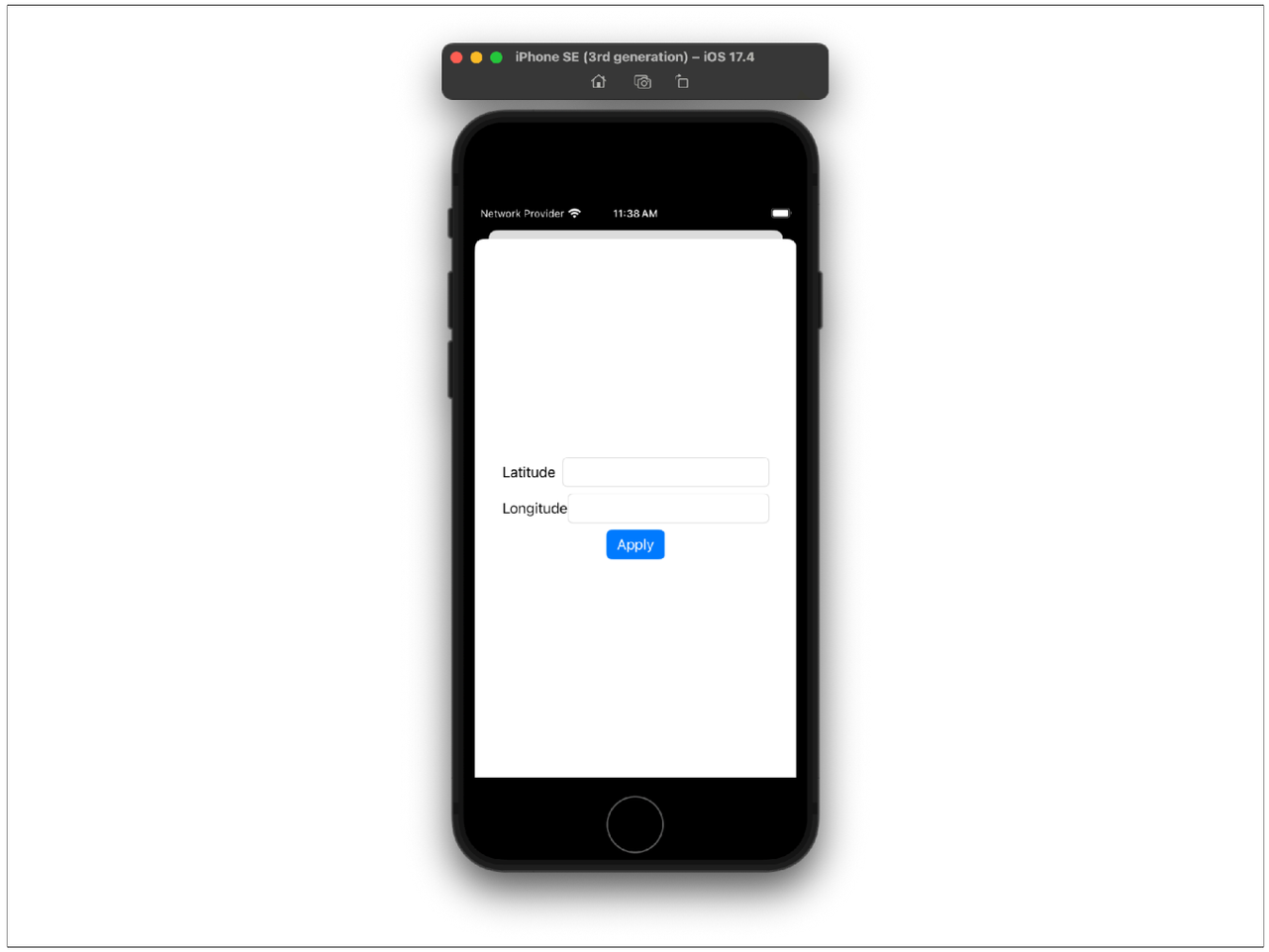
Control + Drag from the button to the newly added view controller object, and choose Present Modally.



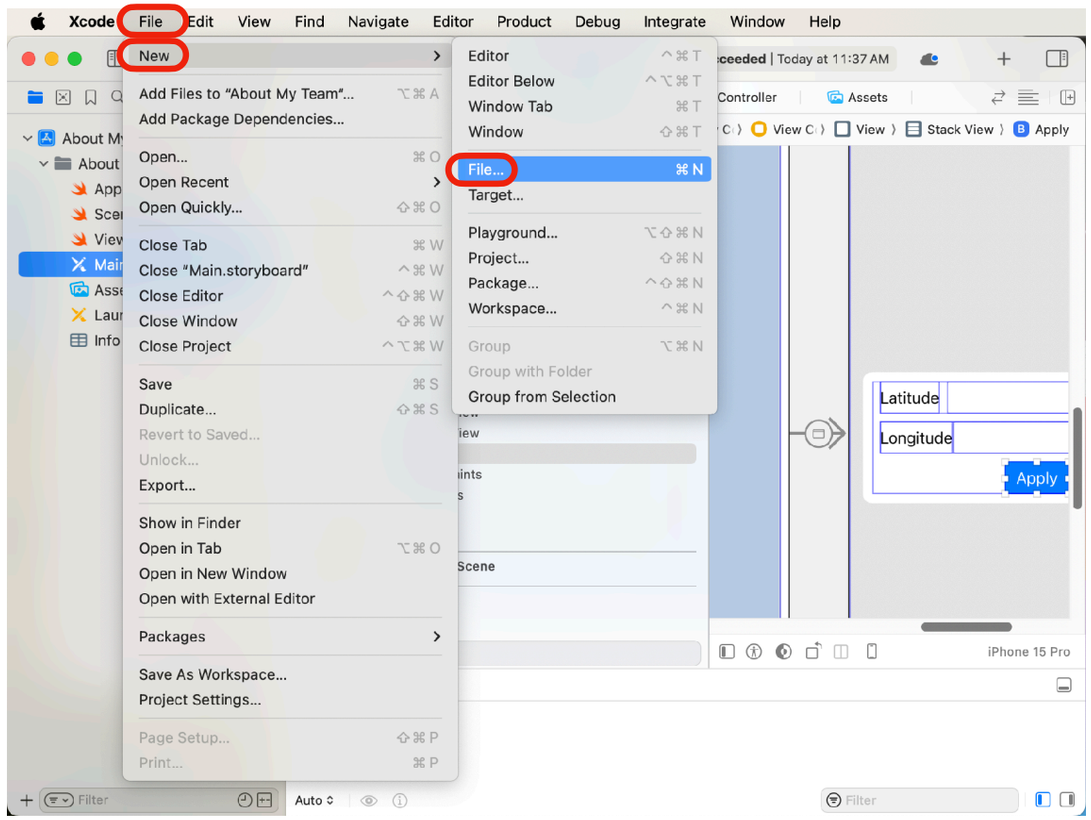
Add the following user interface elements to the newly added view controller screen as shown. It should contain two labels, two text fields and a single button.



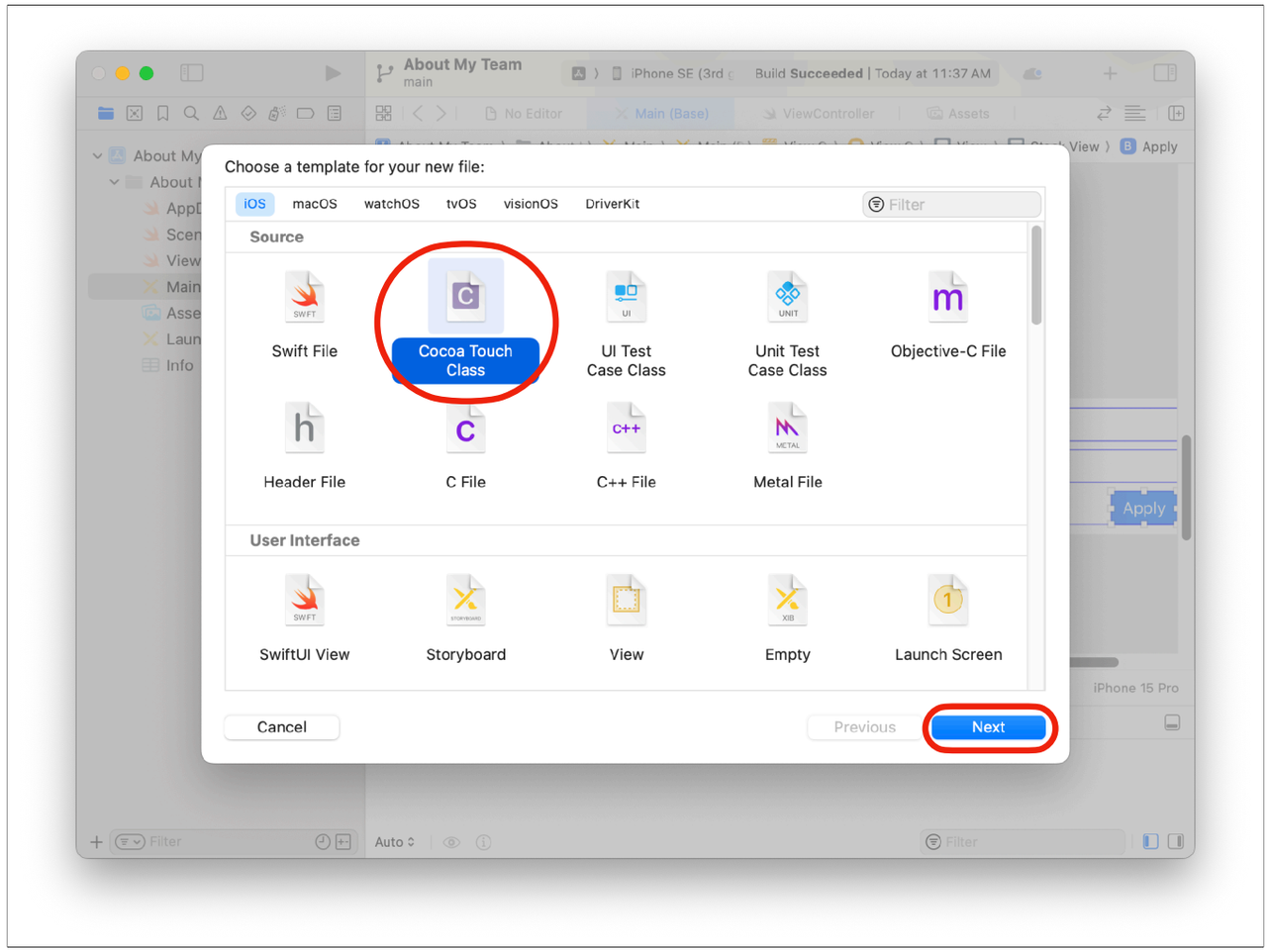
Build and run the app in Simulator. Click the Enter Coordinates button.



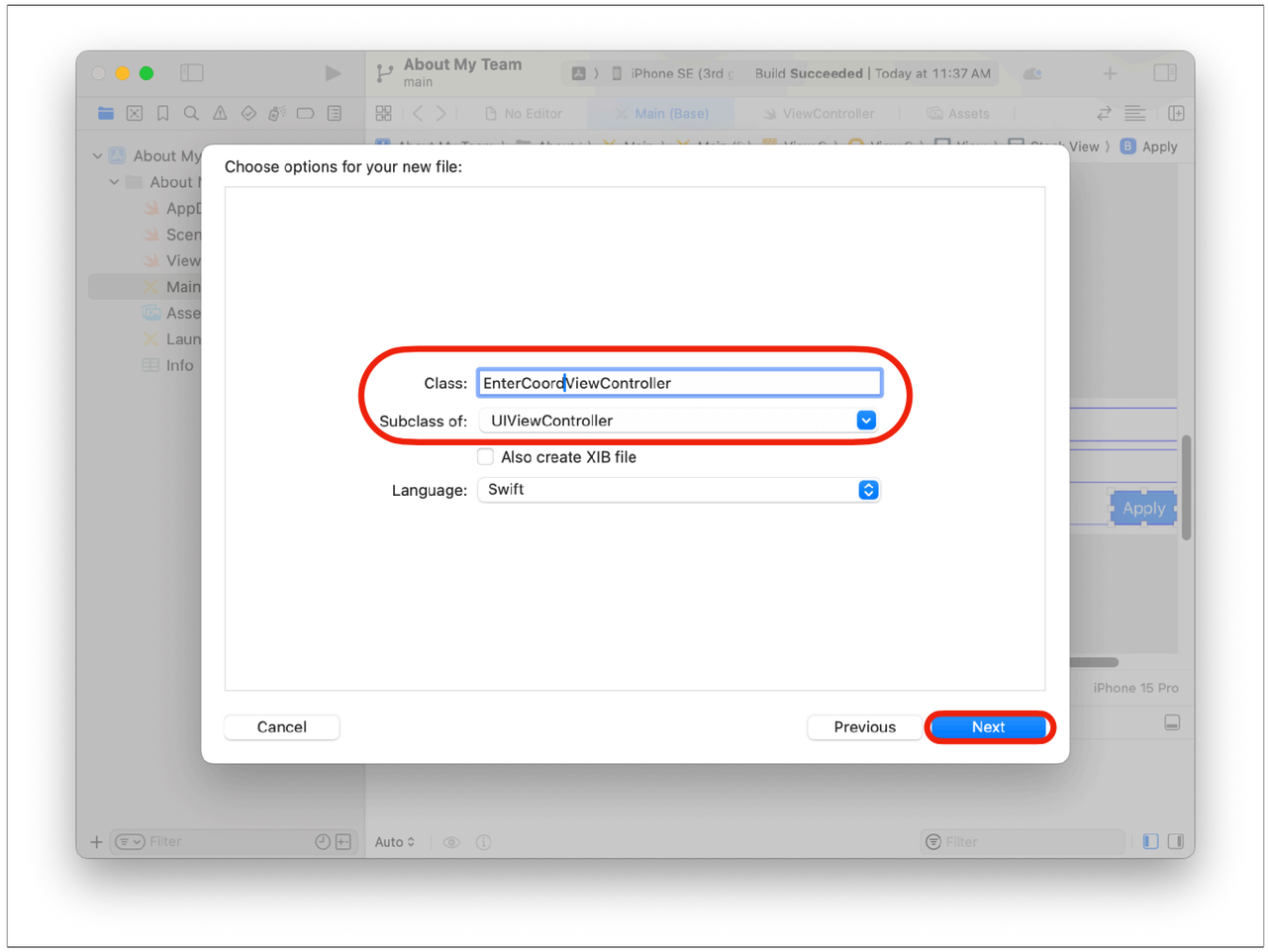
A screen will pop up to allow you to enter map coordinates, but it does not work yet.



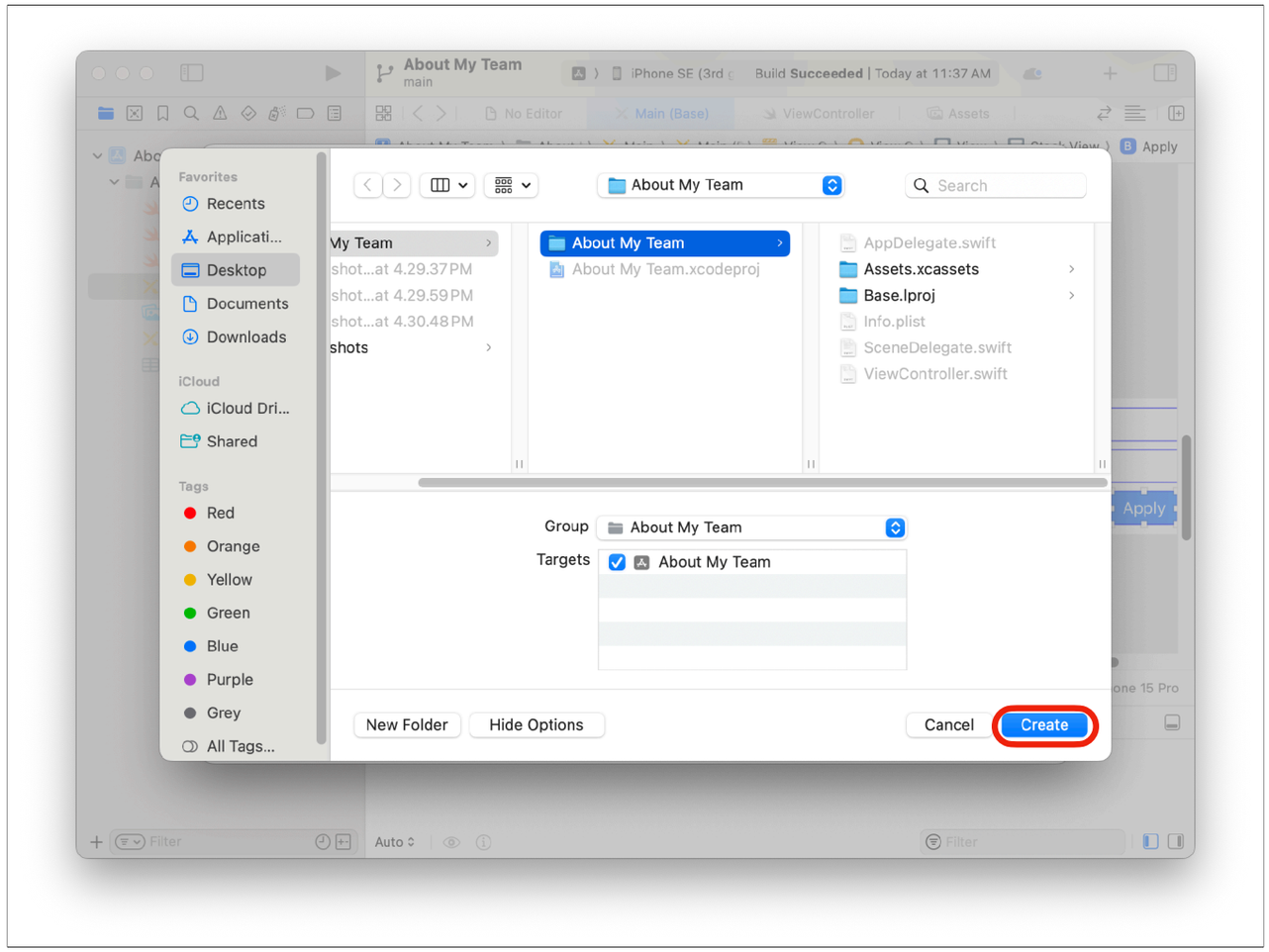
Choose File > New > File.



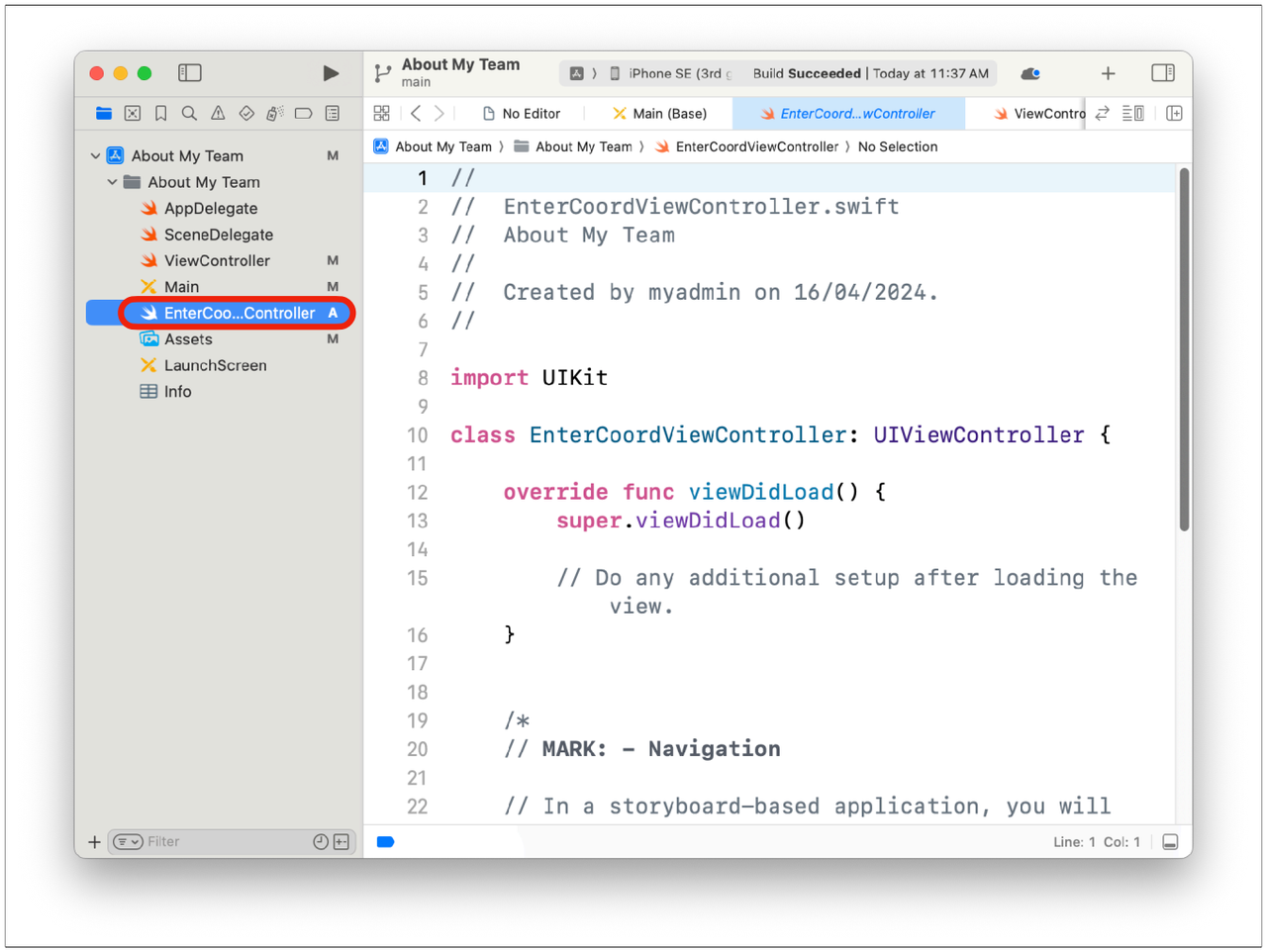
Choose Cocoa Touch Class and click Next.



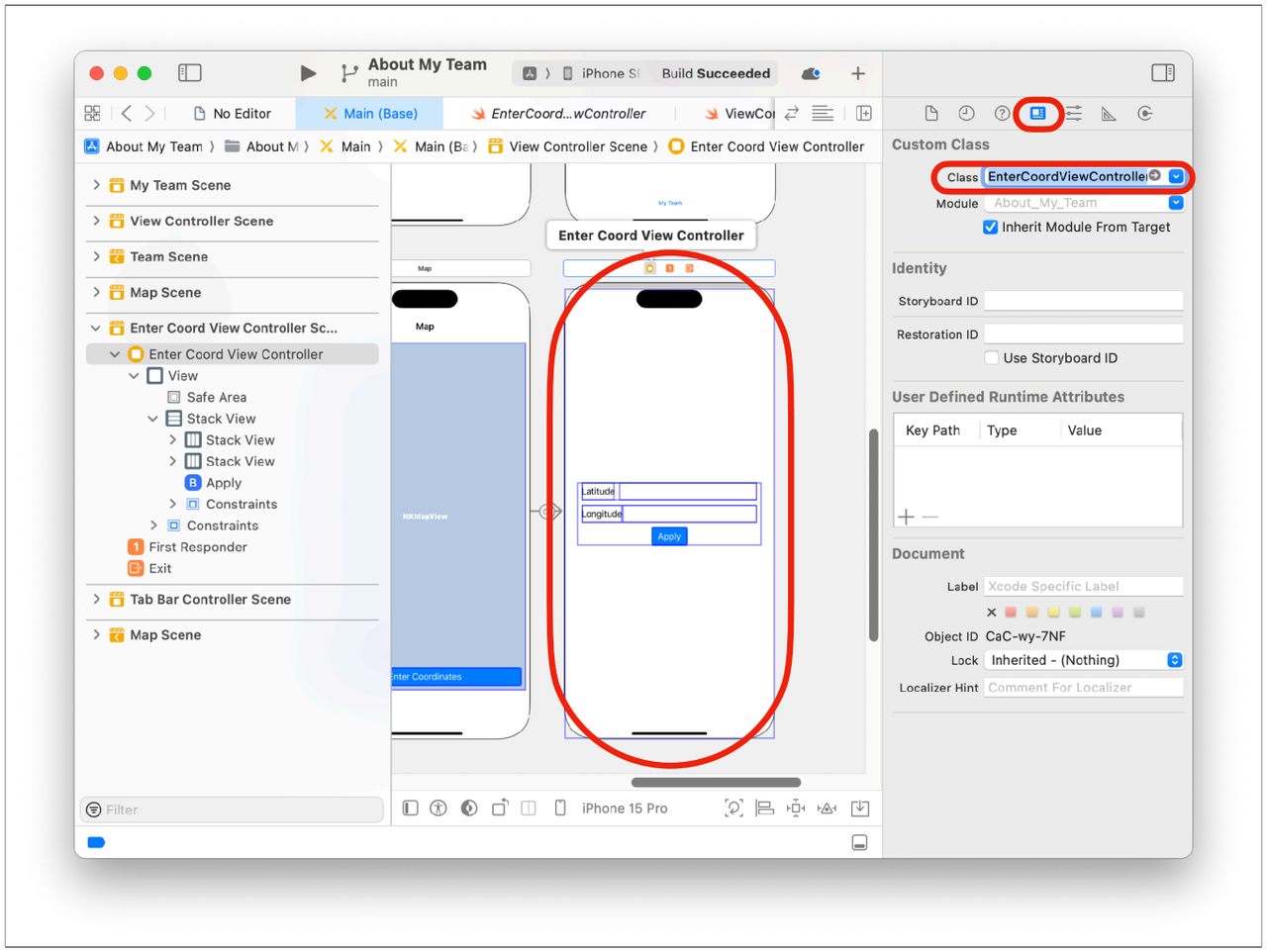
Set Subclass of: to UIViewController. Set Class to EnterCoordViewController. Click Next.



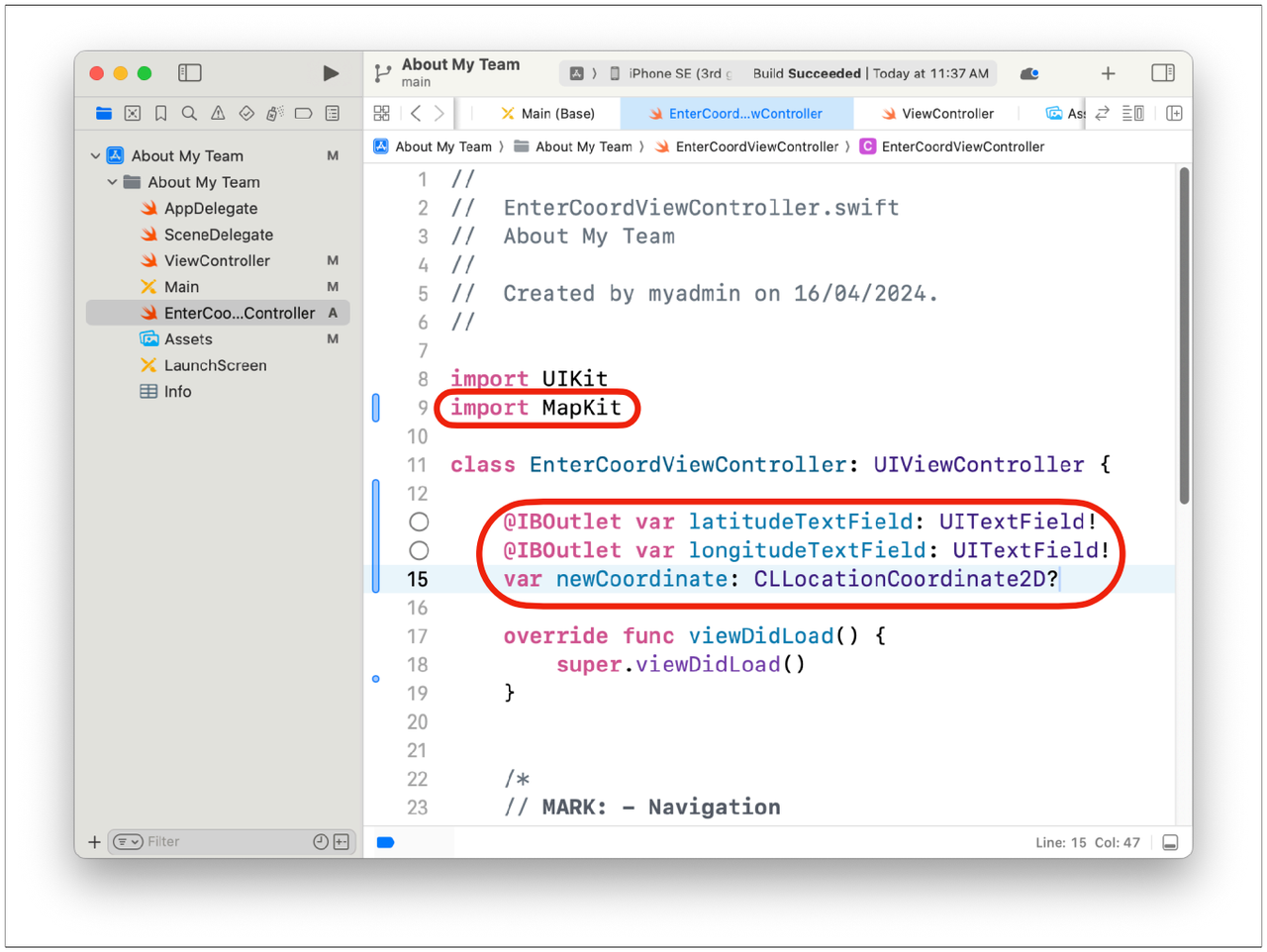
Click Create.



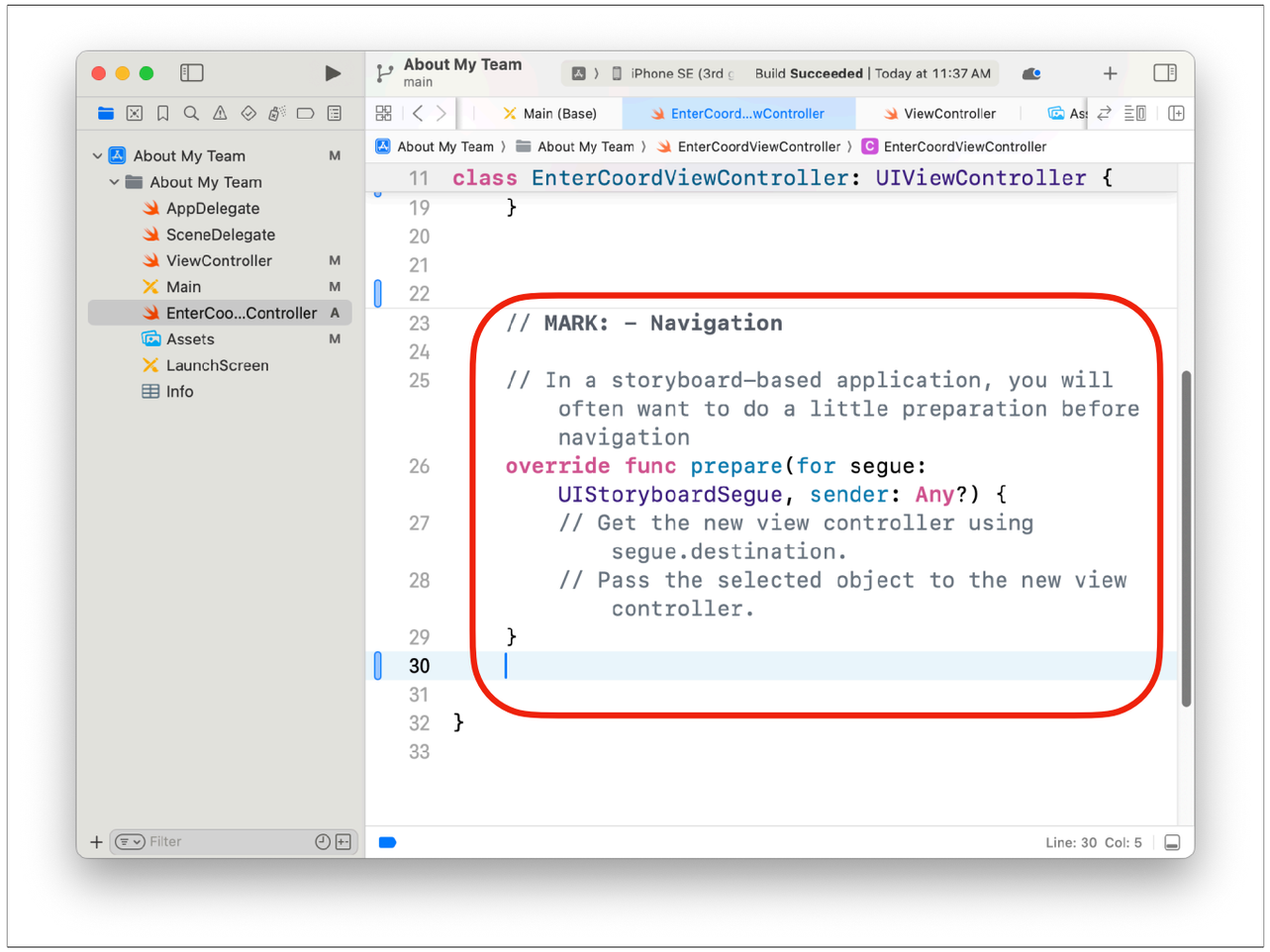
The new file appears in the Project navigator.



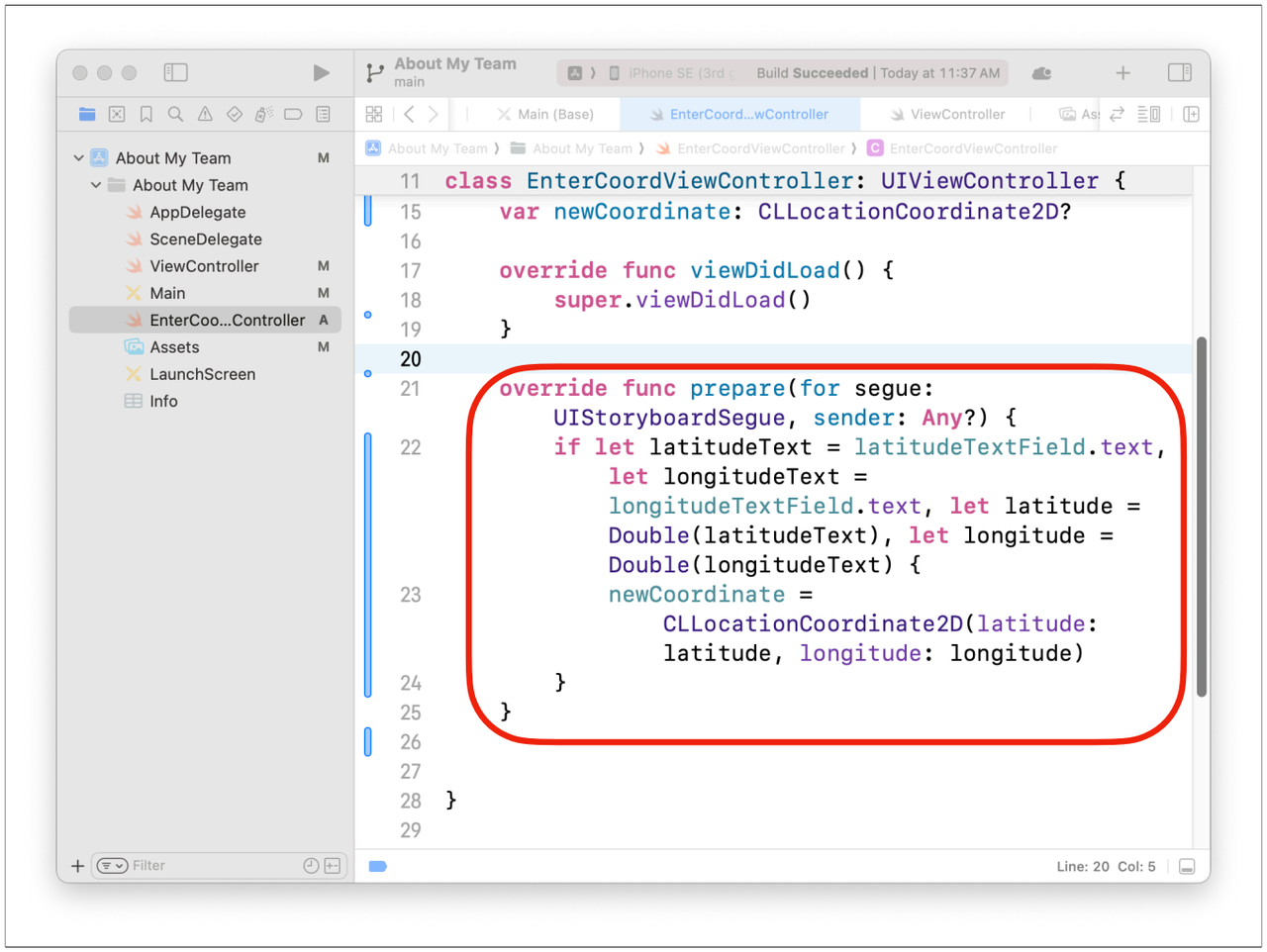
Click Main.storyboard and click the screen with the two labels and button. Click the Identity inspector and set the class to EnterCoordViewController.



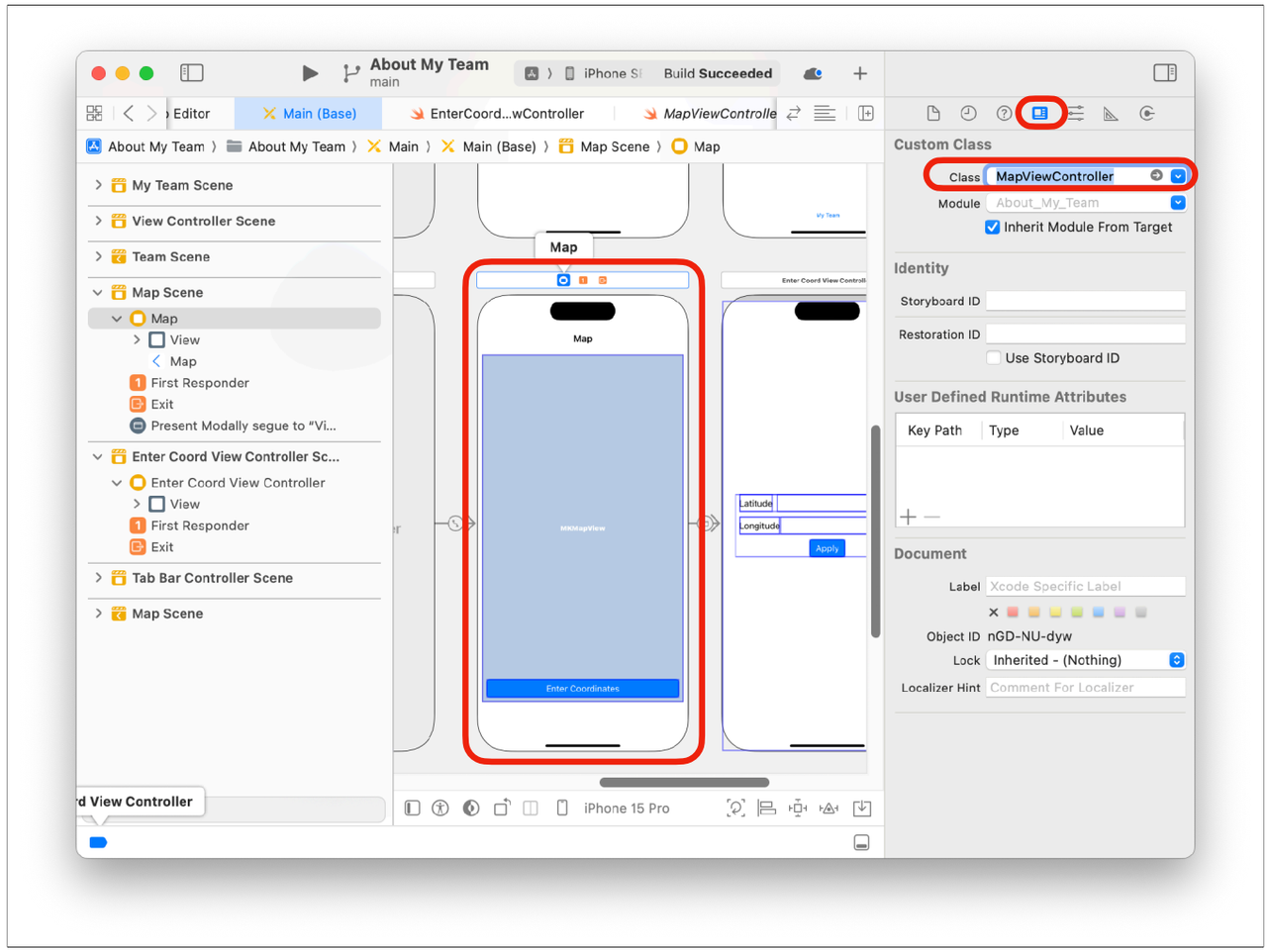
Import MapKit and add the following properties to the EnterCoordViewController file.



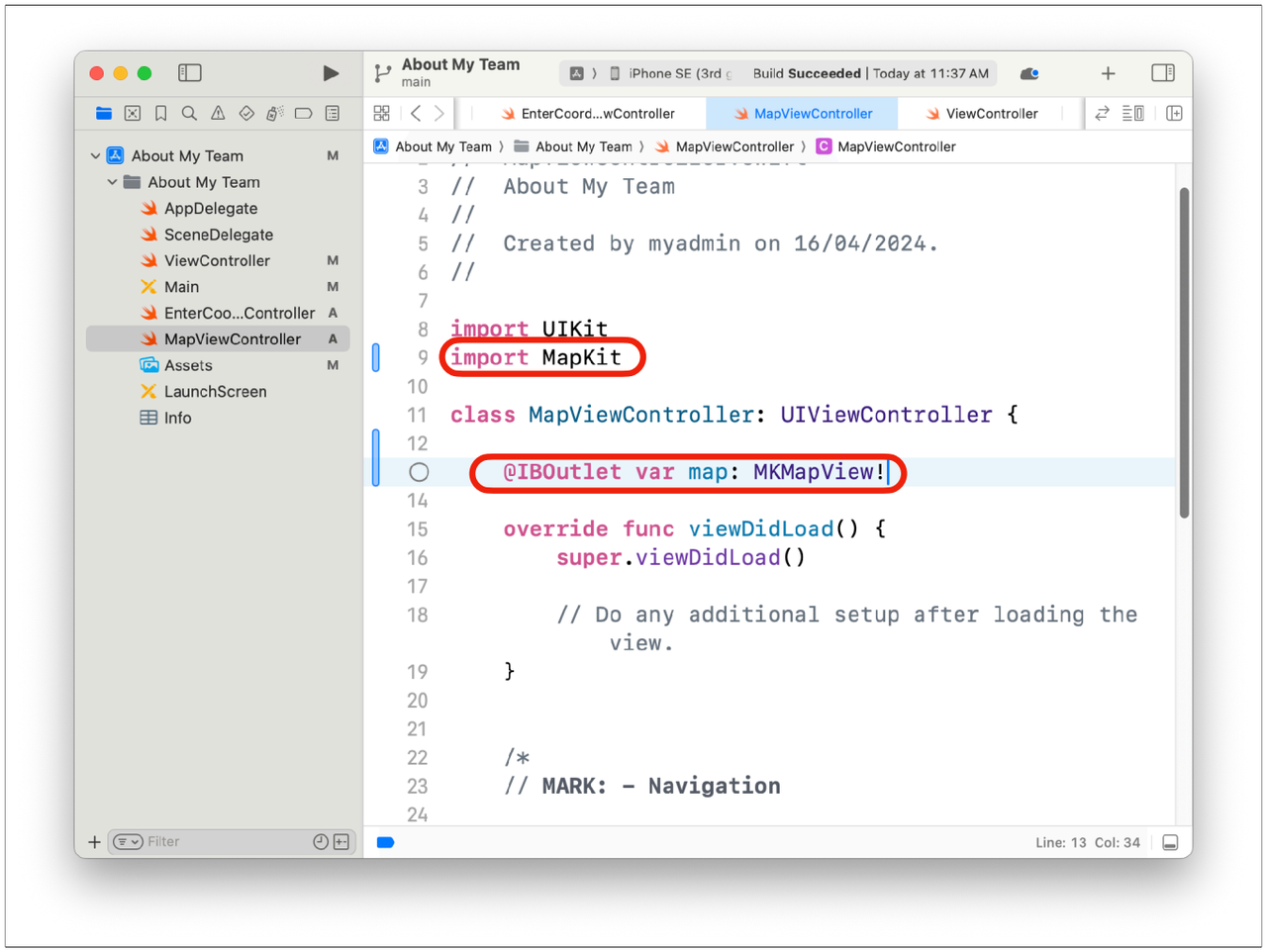
Uncomment the prepare(for:sender:) method.



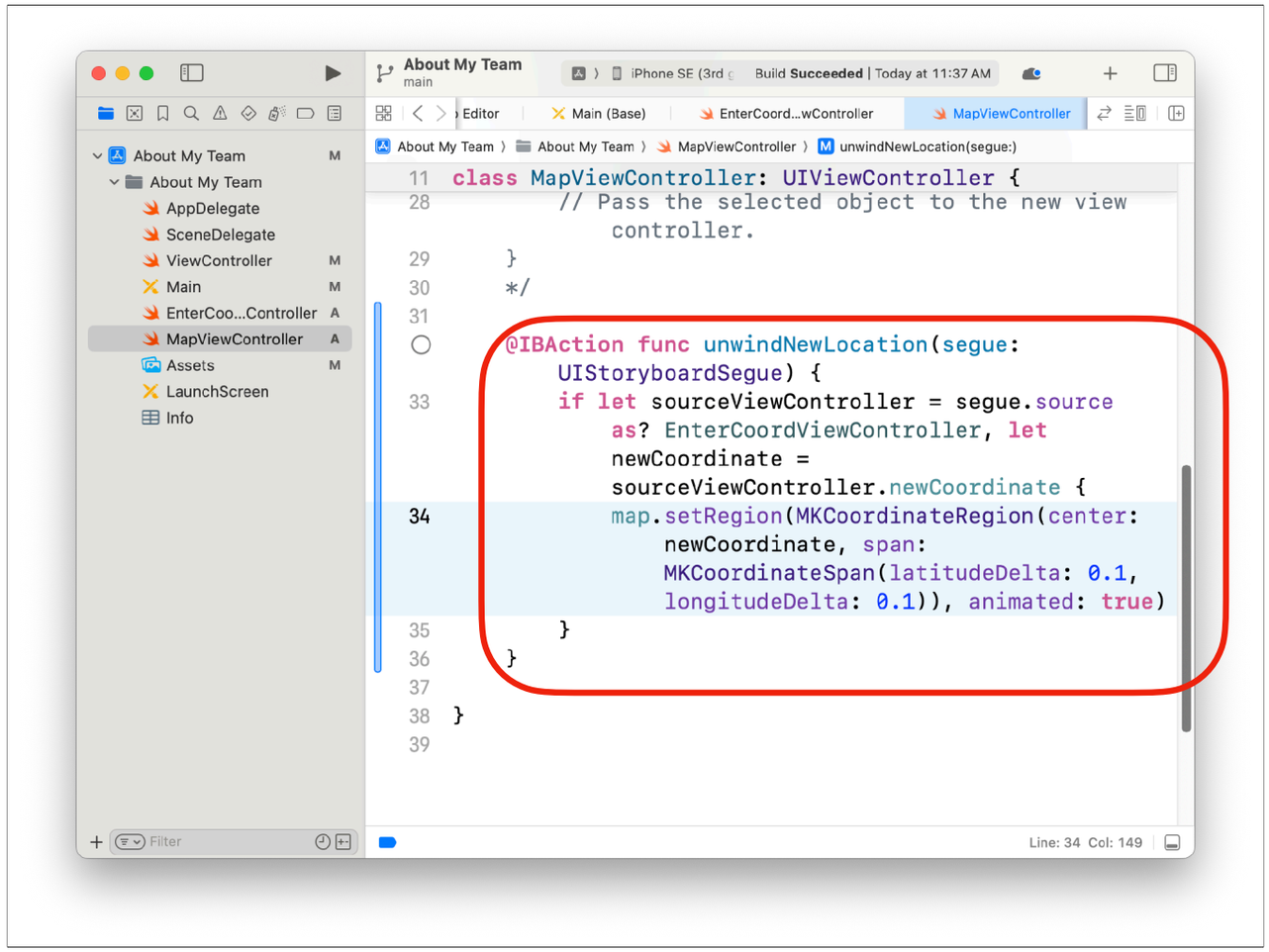
Add the following code to get the values from the text fields and use them to create a new map coordinate.



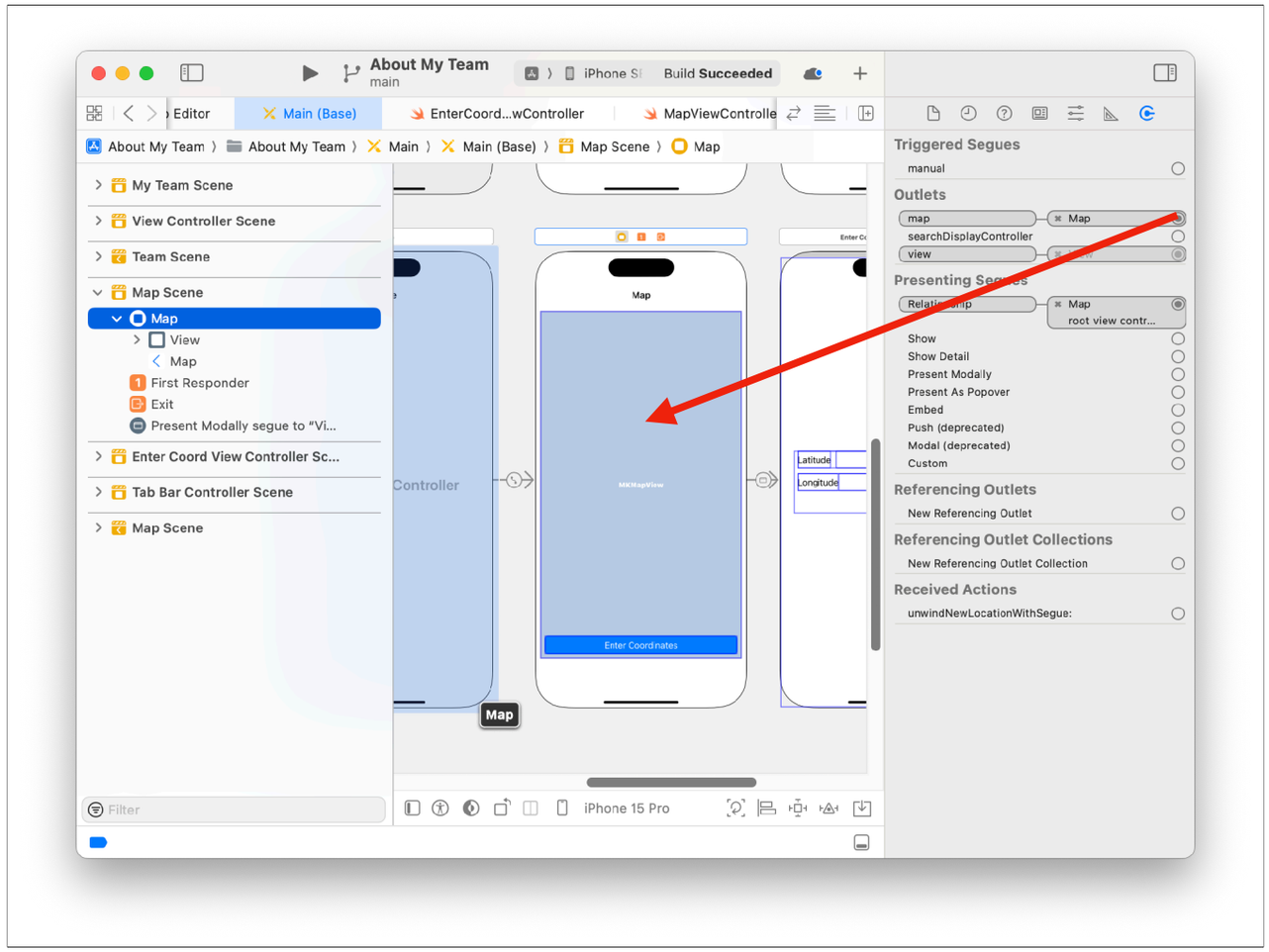
Create a new Cocoa Touch Class file named MapViewController.swift. In the Identity inspector, set the class of the view controller scene that will show a map to MapViewController.swift.



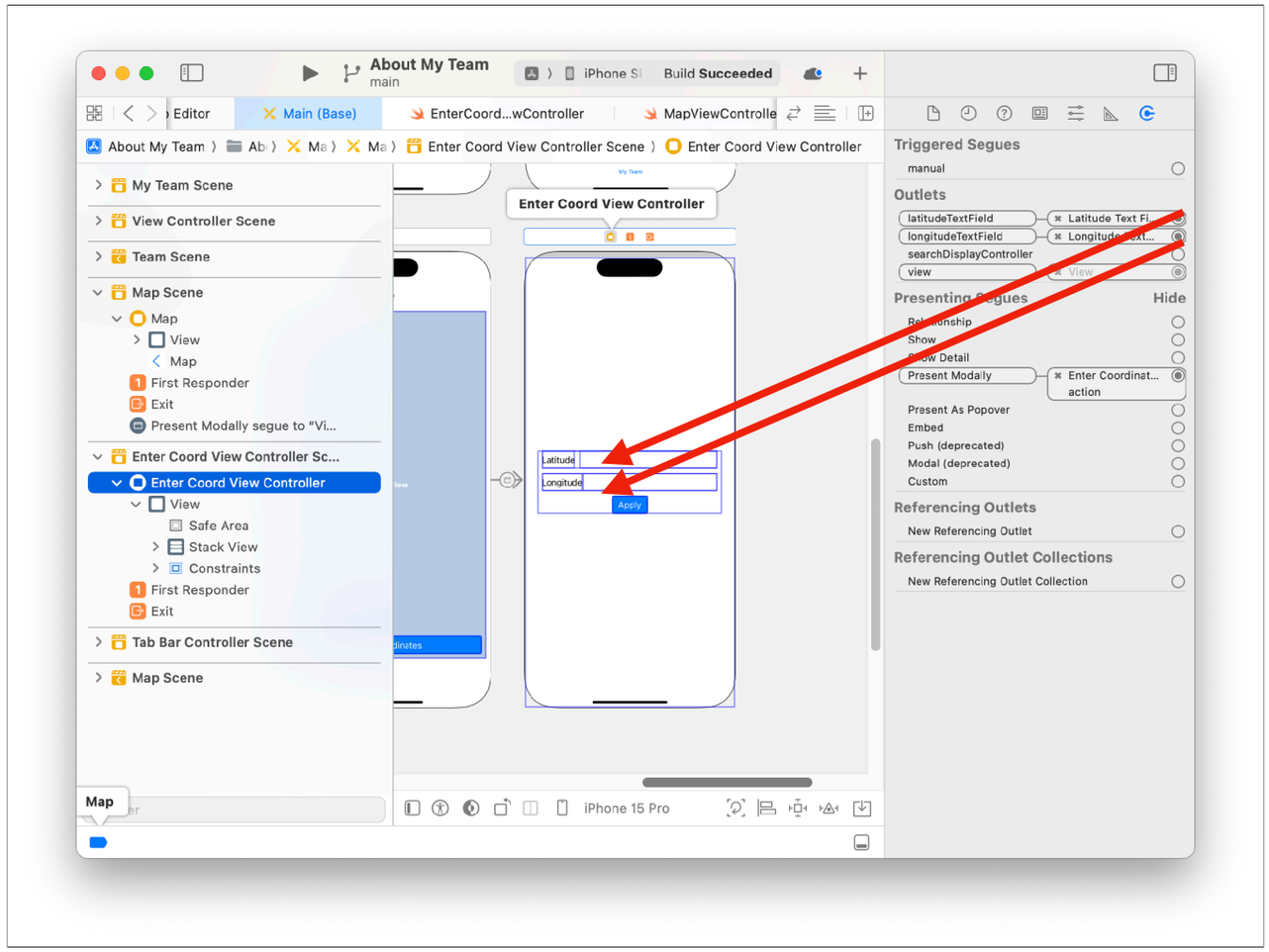
In MapViewController.swift, import MapKit add an outlet for the map view.



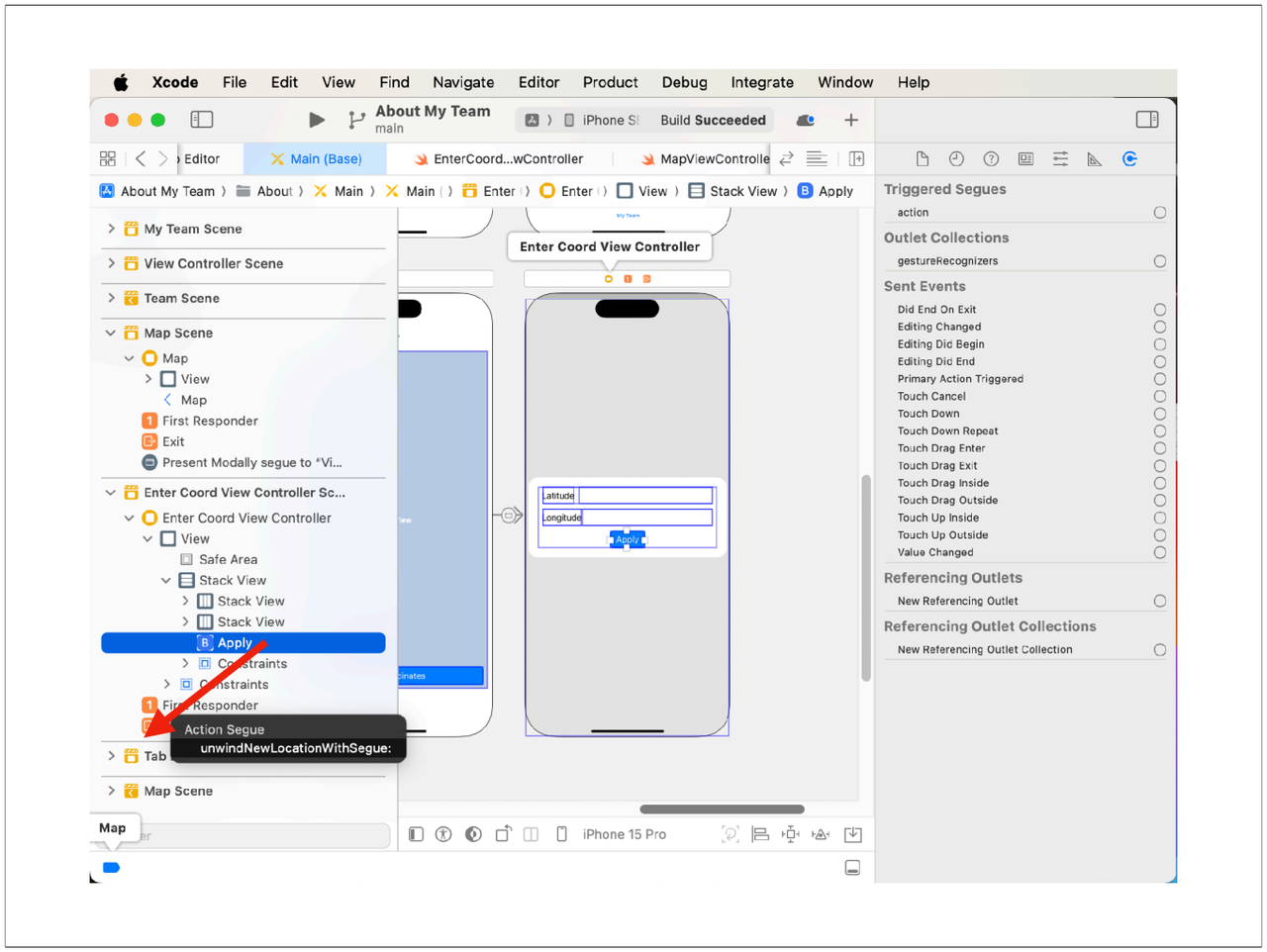
Add the code shown to get the new coordinate from the Enter Coordinate screen and use it to set the region for the map.



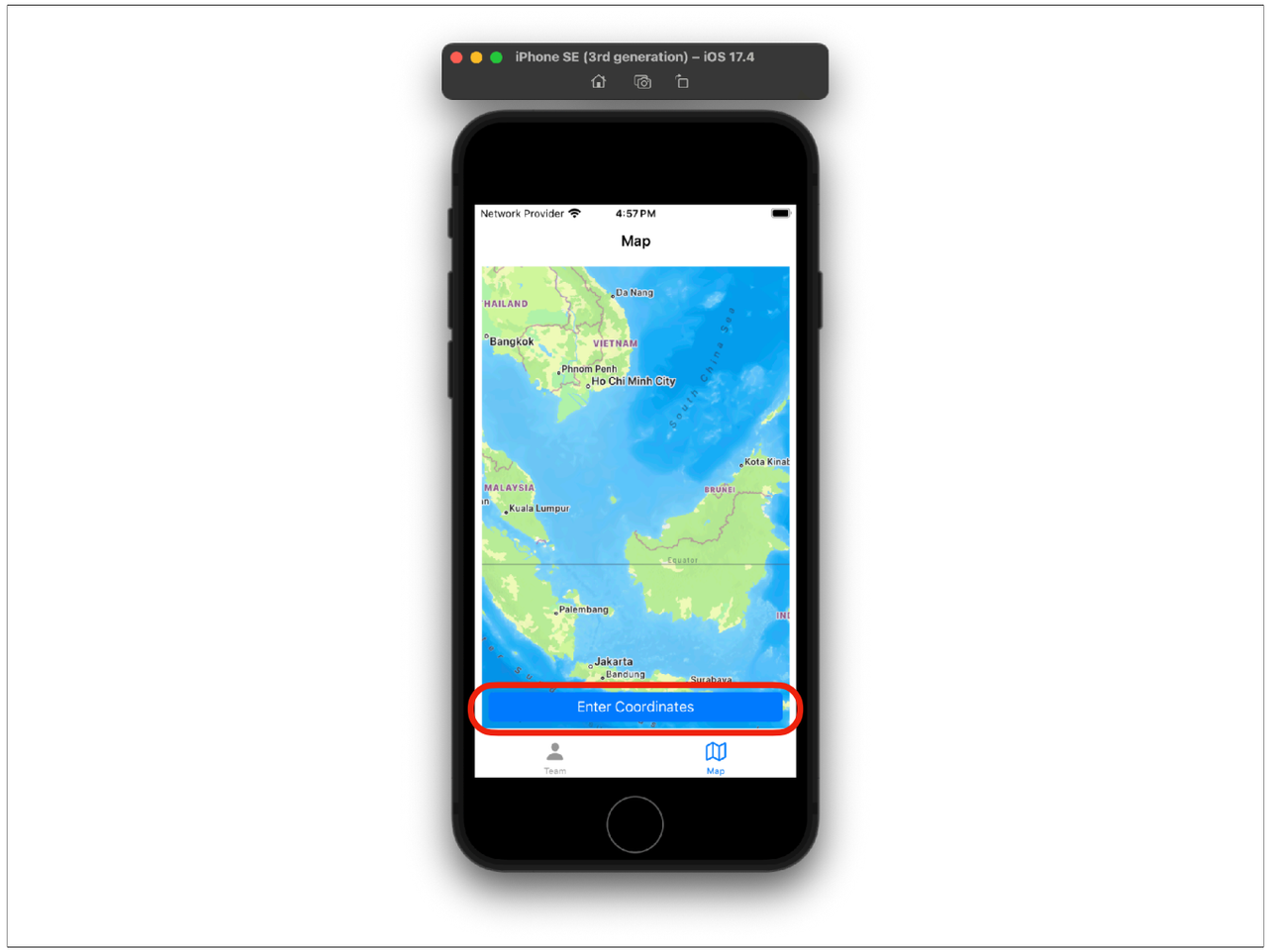
Click the Map view controller scene. In the Connections inspector, connect the map outlet to the map view in the storyboard.



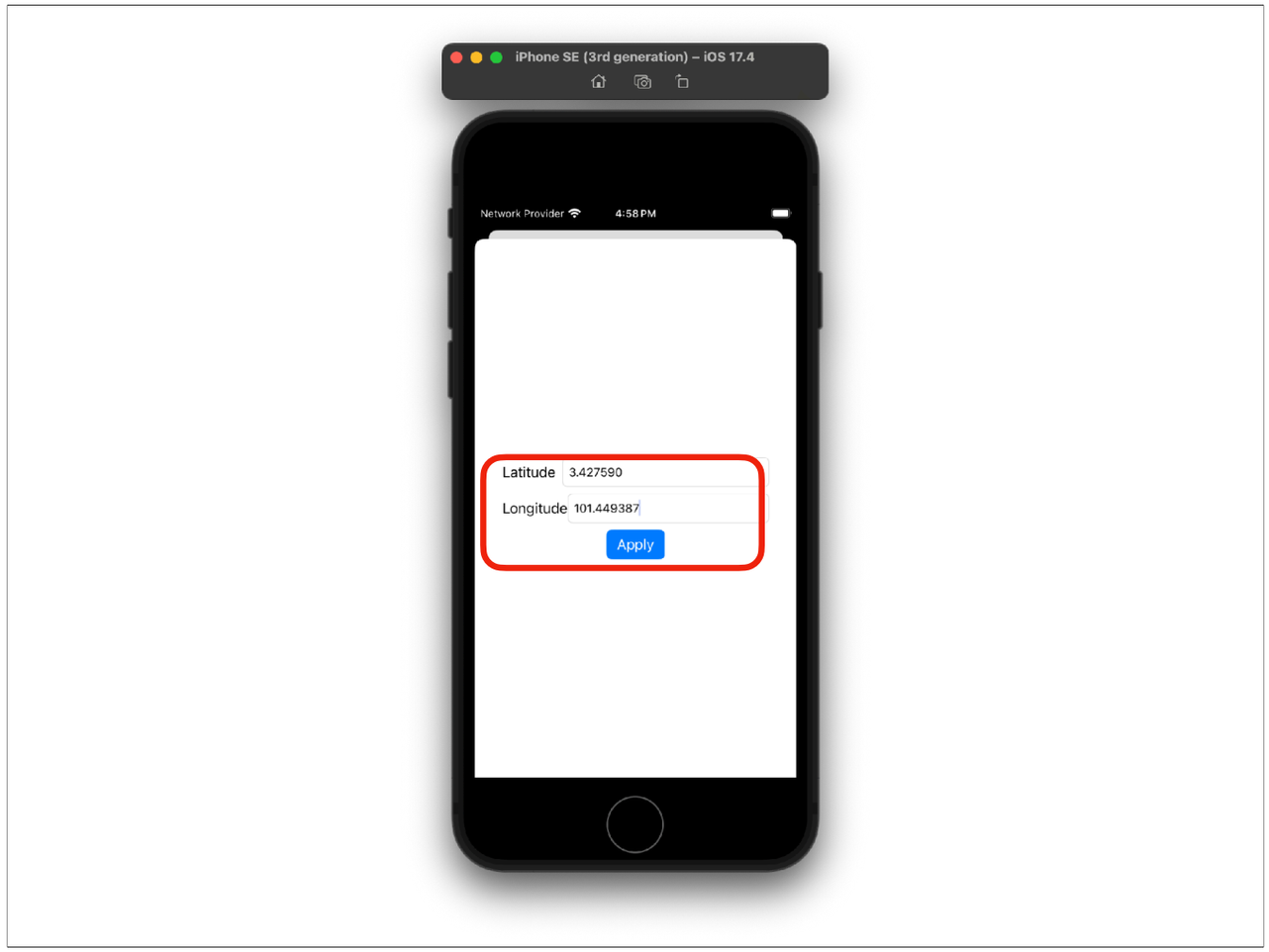
Click the Enter Coordinate view controller scene. Connect the latitude and longitude outlets to the respective text fields.



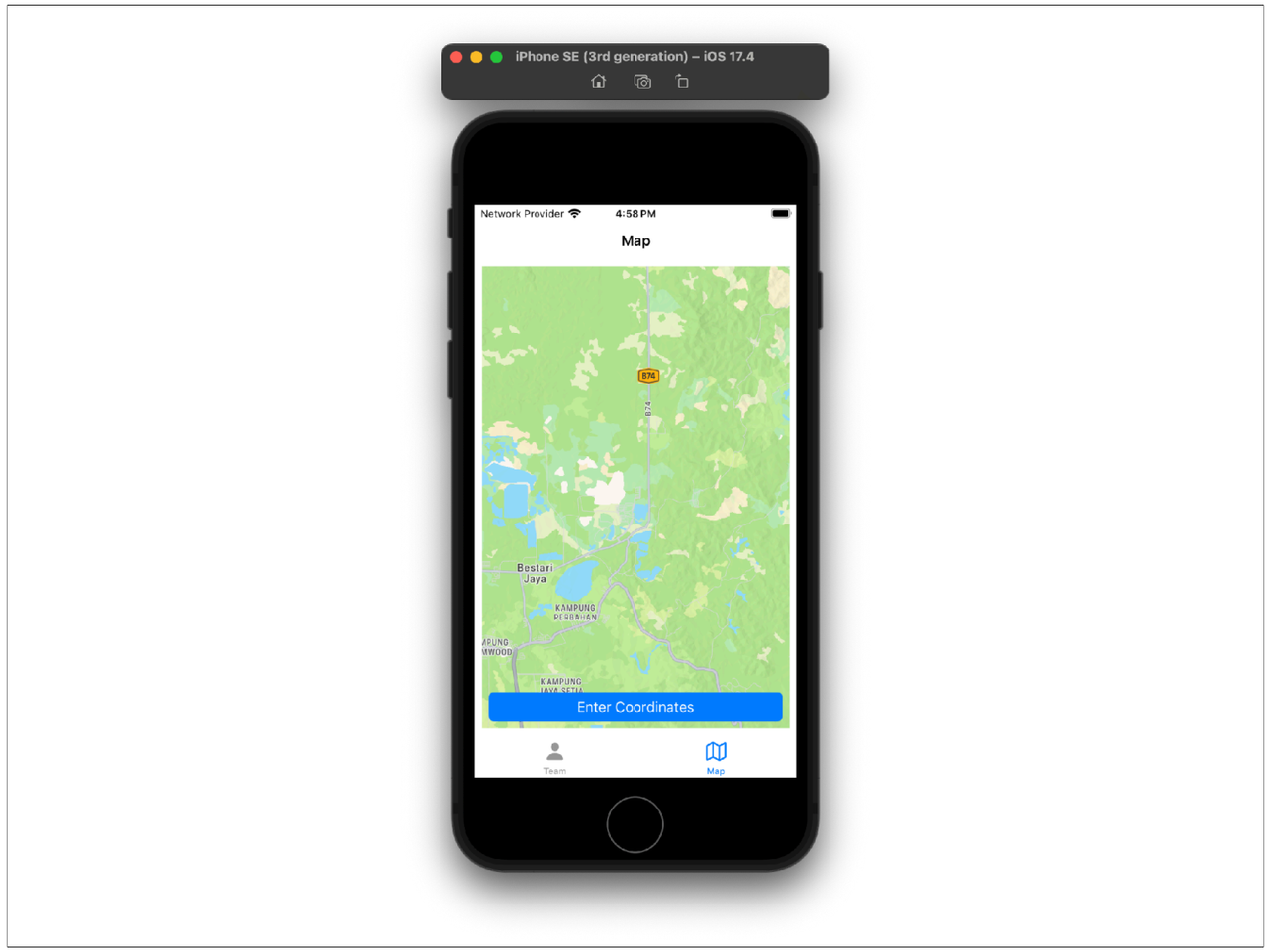
Control + Drag from the button to the scene exit and choose unwindNewLocationWithSegue from the pop up menu.



Build and run the app in Simulator and go to the Map screen. Tap the Enter Coordinates button.



Enter a latitude and longitude and click Apply.



The map will zoom in to the location specified.