About

iSearcher is an iOS app which connects to iTunes Search API and show users list of items. Visited links and deleted items persist across the launches of the app.

UserDefaults is used for data persistence as Data needed to store is just arrays with small number of strings. Loading too much Data on UserDefaults could cause slowing down your app. UserDefaults is a good choice for flags to detect first launches, preferences, etc. For very large Data persistence CoreData framework or other 3rd party libraries could be used.

MVC and Delegation design patterns are used during development of the application. MVC is an essential design pattern widely used in iOS development. This pattern makes app more maintainable and scalable for future expansions. You can easily modify your code. Code becomes easier to visualize and easier to test. Delegation is a design pattern that enables a class or structure to delegate some of its responsibilities to an instance of another type. I needed to use this pattern to make animations right after deletion of an item. Because Detail screen and item list are different View Controllers. I also used it for deleteItem function in the Search class as I needed to delete an item from Data source so Tableview can display the updated Data source with animation.

You can get more info about MVC in Apple's documentation:
https://developer.apple.com/library/archive/documentation/General/Conceptual/CocoaEncyclopedia/Model-View-Controller/Model-View-Controller.html

This app most likely get rejected by AppStore as it's a iTunes Search replica and might be found a simple app for the standards. The app looks great most of the iPhone models, but for iPhone 11 pro 2 rows in landscape mode has too much space. It could have been 3 rows for better UI.

Requirements

Minimum IOS deployment target 13.2 and support both Iphone and Ipad. No 3rd party libraries were used for this app.

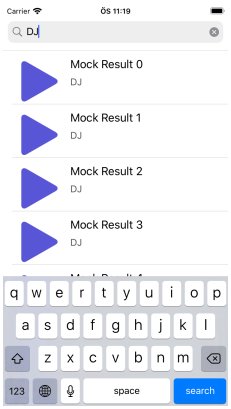
XCode 11.0+ Swift 5.0+

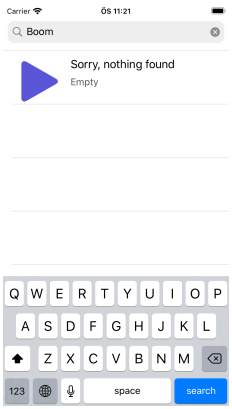
License

This app is open source. If you find a bug please open an issue. Feel free to contribute.

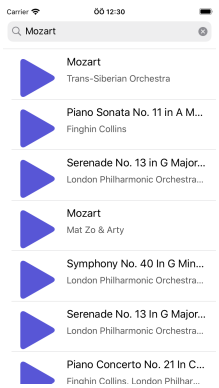
Walkthrough

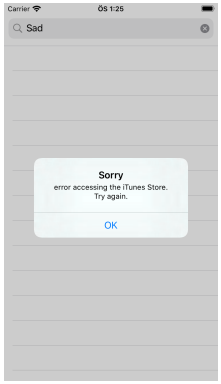
I first made a TableView and get Data from a local array to test the design before connecting iTunes API. Situation for no search results is also implemented by using another Data source.

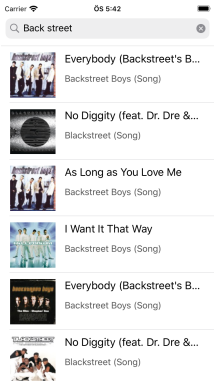




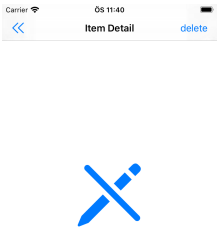
After completing design of nib files and testing for local Data Sources I connected with the API to get real results. Situation for Network problem also tested and alert with description added. Then image download extension added to the UIImageView and this method implemented in the SearchResultCell file.

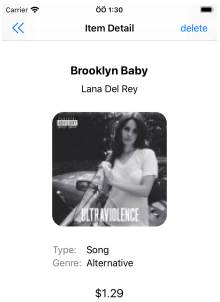


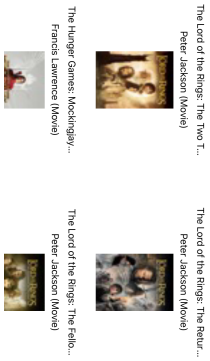




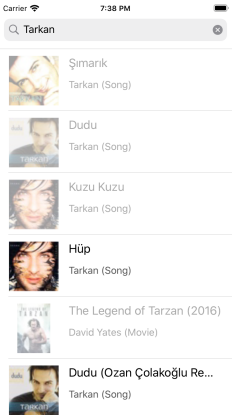
An empty detail screen added and segue connection made. Then this screen filled by data by sending Data source with selected indexPath. Another ViewController with 2 rows added for landscape screen orientations. CollectionView is used in this ViewController.

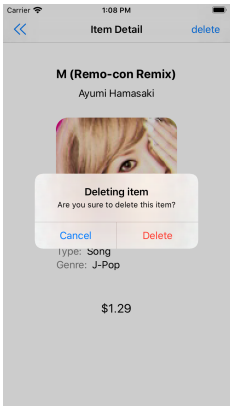






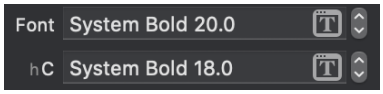
Mechanism to change color of text and change alpha value of the images fort the visited links implemented. It first tested by a local Array. Then persistence made by using UserDefaults. Similar approach made for deleted items as well.

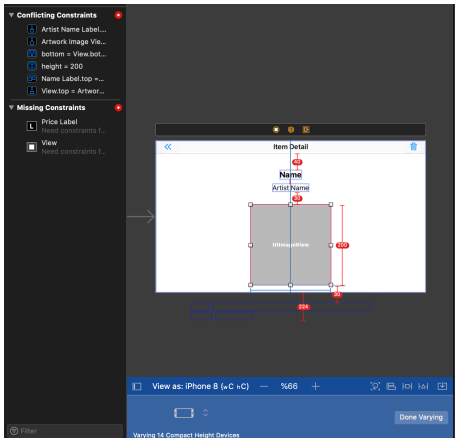


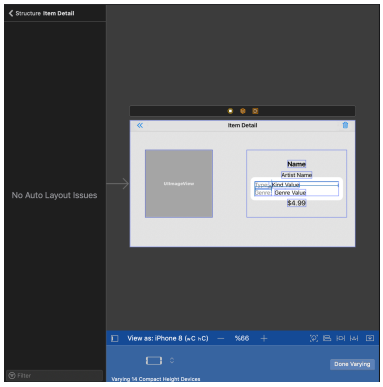


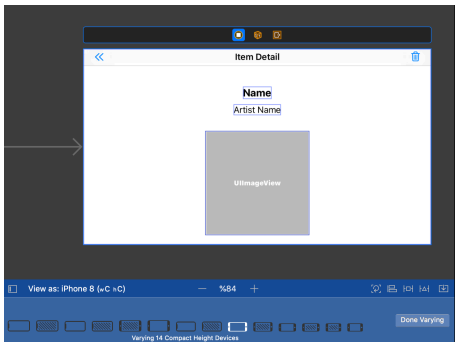
```
if let httpResponse = response as? HTTPURLResponse, httpResponse.statusCode == 200,
    let data = data {
var searchResults = self.parse(data: data)

searchResults = Array(Set(searchResults).subtracting(deletedSearch))

▶ 98 values                                                    Thread 10: step over
    newState = .noResults
```

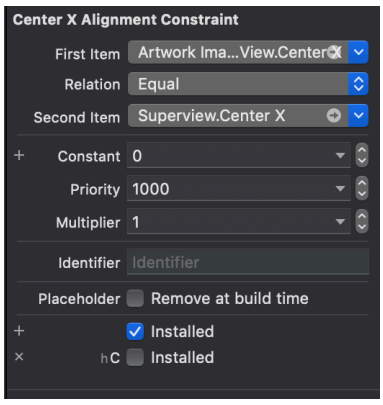
There was a lot of lay out issues for the landscape mode. Vary for traits option used to use constraints according to device orientations. Compact height (hC) option used for landscape iPhone orientations. This feature also used for Font sizes.

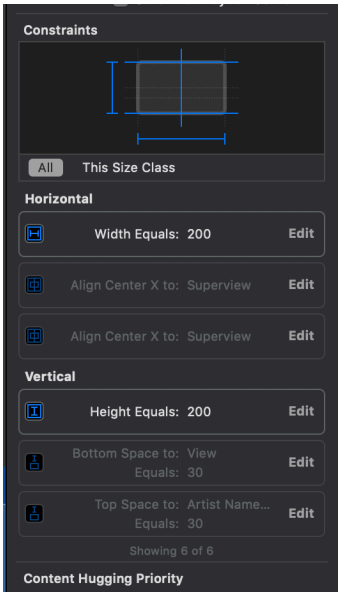


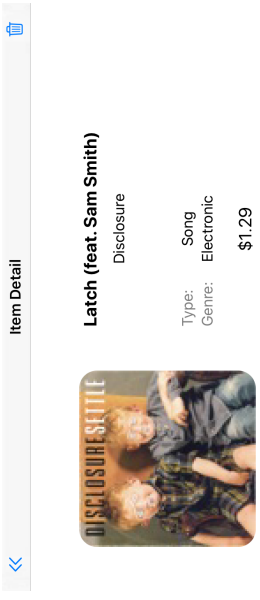




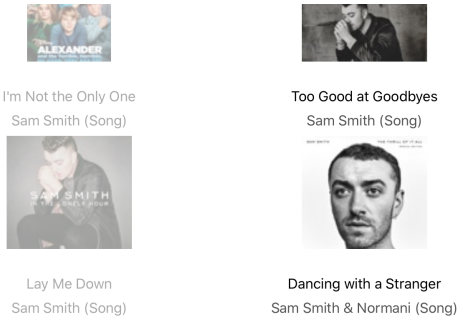


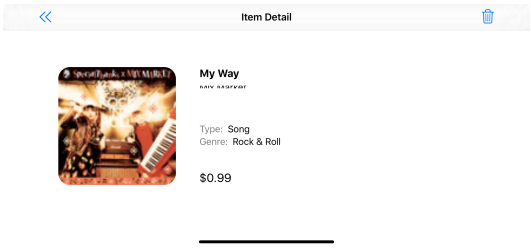


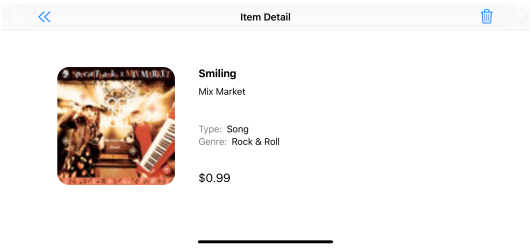




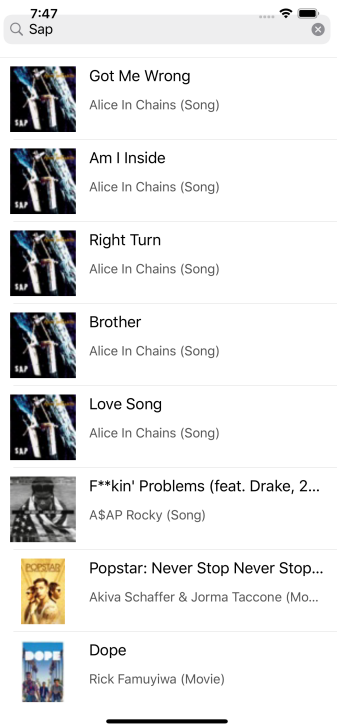
Feature for visited links and deleted item animations also implemented for the horizontal view. Code was mostly copy paste from the portrait orientation View Controller. There were some auto layout issues with horizontal view for larger displays. These problems fixed as well.

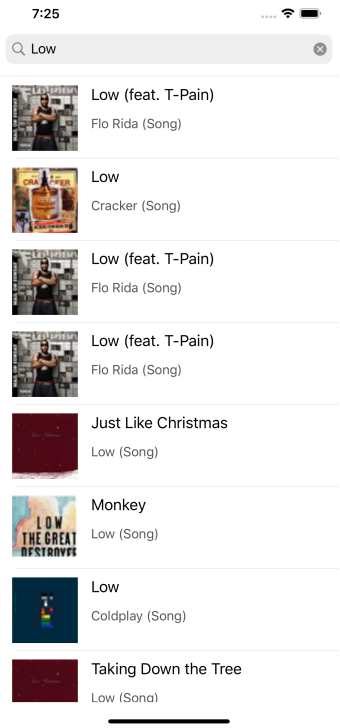






There was issues with notched displays (iPhone X, 11 etc.) as lay out constraints were according to superview. These issues fixed by setting constraints according to safe area.





Unit Tests were made. Firstly tests made for checking URLSession to be sure Network connection to iTunes API works perfectly. Checked if we can get status valid connection code 200 for correct urls given. Case for wrong urls also checked. Then tests for parsing made by using Search Class. performSearch(..) function uses parsing method inside so we can check if it's parsing data received from the API. Another case also added to check if filtering deleted items work. Folder organizations made.