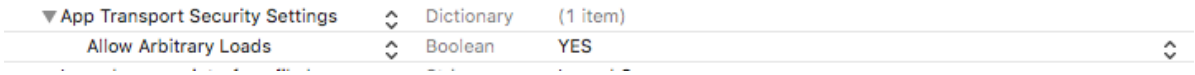
QMTL

Library Integration steps:

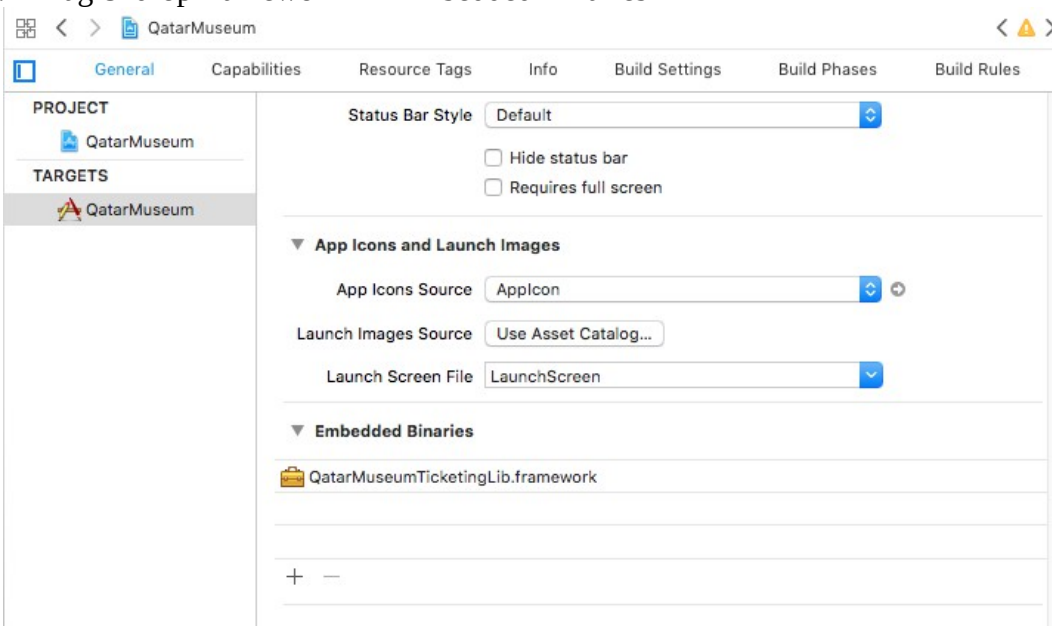
1. Add pods
    1. pod 'Alamofire', '~> 4.5'
    2. pod 'JGProgressHUD'
    3. pod 'SwiftyJSON'
    4. pod 'FSPagerView'
    5. pod 'Toast-Swift', '~> 4.0.0'
    6. pod 'FSCalendar'
    7. pod 'ActionSheetPicker-3.0', '~> 2.3.0'
    8. pod 'Kingfisher', '<= 4.9.0'
    9. pod 'QRCode'
    10. pod 'KeychainSwift', '~> 11.0'

2. Add app transport security in plist



3. Add AppConstants.swift into your project

4. Add QMTL framework into your project
    1. Drag & drop framework in “Embedded Binaries”



5. Add below code into buyTicketBtnAction

```
@IBAction func buyTicketBtnAction(_ sender: Any) {

    var storyBoard = UIStoryboard()

    UserDefaults.standard.set(AppConstants.QMTLibConstants.QMTLTicketCounterContainerViewController,
forKey: AppConstants.QMTLibConstants.initialViewControllerKey)
    let bundle = Bundle(identifier: AppConstants.QMTLibConstants.bundleId)
    storyBoard = UIStoryboard(name: AppConstants.QMTLibConstants.QMTStoryboardForEN_Id, bundle: bundle)
    let controller = storyBoard.instantiateViewController(withIdentifier:
AppConstants.QMTLibConstants.QMTLTabViewController)
    //self.navigationController?.pushViewController(controller, animated: true)
    self.present(controller, animated: true, completion: nil)

}
```

6. Add below code into culturePassButtonPressed

```
func culturePassButtonPressed() {
    var storyBoard = UIStoryboard()
    UserDefaults.standard.set(AppConstants.QMTLibConstants.QMTLUserProfileTableViewController, forKey:
AppConstants.QMTLibConstants.initialViewControllerKey)
    let bundle = Bundle(identifier: AppConstants.QMTLibConstants.bundleId)
    storyBoard = UIStoryboard(name: AppConstants.QMTLibConstants.QMTStoryboardForEN_Id, bundle: bundle)
    let controller = storyBoard.instantiateViewController(withIdentifier:
AppConstants.QMTLibConstants.QMTLTabViewController)
    //self.navigationController?.pushViewController(controller, animated: true)
    self.present(controller, animated: true, completion: nil)
}
```

Creating Universal fat library steps:

- Build ‘QatarMuseumTicketingLib’ target for iOS simulator and extract framework from products folder on your desktop.
- Rename the framework to QatarMuseumTicketingLib-sim.framework so that it is distinguishable later.
- Repeat the steps 1 and 2 for iOS device. You can select ‘iOS Device’. Don’t forget to rename the framework to QatarMuseumTicketingLib-dev.framework.
- Use the following command to combine both binaries into a single fat binary file .

```
$lipo -create ./QatarMuseumTicketingLib-sim.framework/QatarMuseumTicketingLib
./QatarMuseumTicketingLib-dev.framework/QatarMuseumTicketingLib -output ./QatarMuseumTicketingLib
```

- Copy QatarMuseumTicketingLib binary file created in above step and replace it with the binary in QatarMuseumTicketingLib-dev.framework folder.
- From folder

  QatarMuseumTicketingLib-sim.framework/Modules/QatarMuseumTicketingLib.swiftmodule/

- copy ‘x86_64.swiftdoc’ and ‘x86_64.swiftmodule’ and paste them to

  QatarMuseumTicketingLib-dev.framework/Modules/QatarMuseumTicketingLib.swiftmodule/

- By following above steps you have converted QatarMuseumTicketingLib-dev.framework from device only to a universal fat framework. Rename it to QatarMuseumTicketingLib.framework.
- Include this framework via ‘Embeded Binaries’ option in Xcode. Import the module in your file and you would be able to compile it successfully.