

# **Oracle Banking Digital Experience**

**Mobile Application Builder Guide – iOS  
Release 18.2.0.0.0**

**Part No. E97823-01**

**June 2018**

**ORACLE®**

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway

Goregaon (East)

Mumbai, Maharashtra 400 063

India

Worldwide Inquiries:

Phone: +91 22 6718 3000

Fax: +91 22 6718 3001

[www.oracle.com/financialservices/](http://www.oracle.com/financialservices/)

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate failsafe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

This software or hardware and documentation may provide access to or information on content, products and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

# Table of Contents

<b>1. Preface.....</b>	<b>4</b>
1.1 Intended Audience .....	4
1.2 Documentation Accessibility .....	4
1.3 Access to Oracle Support .....	4
1.4 Structure.....	4
1.5 Related Information Sources.....	4
<b>2. OBDX Servicing Application .....</b>	<b>5</b>
2.1 Pre requisite .....	5
2.2 Create Project .....	5
2.3 Adding UI to workspace.....	5
2.4 Open project in Xcode.....	6
2.5 Generating Certificates for Development, Production and Push Notifications .....	8
<b>3. Archive and Export.....</b>	<b>13</b>
<b>4. OBDX Authenticator Application .....</b>	<b>15</b>
4.1 Building Authenticator UI.....	15
4.2 Authenticator Application Workspace Setup.....	18
4.3 Building Authenticator Application.....	20

# 1. Preface

## 1.1 Intended Audience

This document is intended for the following audience:

- Customers
- Partners

## 1.2 Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=accandid=docacc>.

## 1.3 Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit

<http://www.oracle.com/pls/topic/lookup?ctx=accandid=info> or visit

<http://www.oracle.com/pls/topic/lookup?ctx=accandid=trs> if you are hearing impaired.

## 1.4 Structure

This manual is organized into the following categories:

*Preface* gives information on the intended audience. It also describes the overall structure of the User Manual.

The subsequent chapters describes following details:

- Configuration / Installation.

## 1.5 Related Information Sources

For more information on Oracle Banking Digital Experience Release 18.2.0.0.0, refer to the following documents:

- Oracle Banking Digital Experience Licensing Guide

## 2. OBDX Servicing Application

### 2.1 Pre requisite

- Download and Install node js as it is required to run npm commands.
- XCode to be download from Mac App Store

### 2.2 Create Project

1. Extract iOS workspace from installer and place in a folder.
2. The workspace by default contains framework for running on devices. Hence to run the application on simulator, delete and copy the 3 frameworks (OBDXExtensions.framework, OBDXFramework.framework, OBDXWatchFramework.framework) from installer/simulator to zigbank\platforms\ios directory.

### 2.3 Adding UI to workspace.

*Use any 1 option below*

- a. Building un built UI (required in case of customizations)

Extract unbuilt UI and traverse to **OBDX\_Installer/installables/ui/channel/\_build** folder and perform below steps

Windows –

```
npm install -g grunt-cli
npm install
set IS_GRUNT=true
node render-requirejs/render-requirejs.js mobile
npm install cwebp-bin
```

Copy "vendor" directory from \_build/node\_modules/cwebp-bin/ to  
\_build/node\_modules/gruntcwebp/node\_modules/cwebp-bin

```
grunt --max_old_space_size=5120 mobilebuild --platform=ios
```

Linux –

```
sudo npm install -g grunt-cli
sudo npm install
export IS_GRUNT=true
node render-requirejs/render-requirejs.js mobile
npm install cwebp-bin
```

Copy "vendor" directory from \_build/node\_modules/cwebp-bin/ to  
\_build/node\_modules/gruntcwebp/node\_modules/cwebp-bin

```
grunt --max_old_space_size=5120 mobilebuild --platform=ios
```

- b. Using built UI (out of box shipped with installer)
  - i. Unzip dist.tar.gz for android from installer and copy folders(components,extensions,framework,images,json,lzn,pages,partials,resource, index.html, build.fingerprint) to workspace (platforms/ios/www/)

**Delete originations folder inside images (images/originations) and ensure webhelp folder is not copied.**

## 2.4 Open project in Xcode

Open Xcode by clicking ZigBank.xcodeproj at zigbank/platforms/ios/

1. Adding URLs to app.plist (ZigBank/Resources)

a. FOR NONOAM (DB Authenticator setup)

SERVER_TYPE	NONOAM
KEY_SERVER_URL	http://mum00chx.in.oracle.com:3333
WEB_URL	http://mum00chx.in.oracle.com:3333
PinnedCertificateName	Name of SSL certificate without extension of OBDX App Server

a. OAM Setup (Refer to installer pre requisite documents for OAuth configurations)

SERVER_TYPE	OAM
KEY_SERVER_URL	Eg. http://mum00chx.in.oracle.com:8003 (This URL must be of OHS without webgate)
WEB_URL	Eg. http://mum00chx.in.oracle.com:3333
KEY_OAUTH_PROVIDER_URL	http://mum00aon.in.oracle.com:14100/oauth2/rest/token
APP_CLIENT_ID	<Base64 of clientid:secret> of Mobile App client
APP_DOMAIN	OBDXMobileAppDomain
WATCH_CLIENT_ID	<Base64 of clientid:secret> of wearables
WATCH_DOMAIN	OBDXWearDomain
SNAPSHOT_CLIENT_ID	<Base64 of clientid:secret> of snapshot
SNAPSHOT_DOMAIN	OBDXSnapshotDomain
LOGIN_SCOPE	OBDXMobileAppResServer.OBDXLoginScope
PinnedCertificateOAMName	Name of SSL certificate without extension of OAM Server
PinnedCertificateName	Name of SSL certificate without extension of OBDX App Server

b. IDCS Setup

SERVER_TYPE	IDCS
-------------	------

KEY_SERVER_URL	Eg. http://mum00chx.in.oracle.com:8003 (This URL must be of OHS without webgate)
WEB_URL	Eg.http://mum00chx.in.oracle.com:3333
KEY_OAUTH_PROVIDER_URL	http://obdx-tenant01.identity.c9dev0.oc9qadev.com/oauth2/v1/token
APP_CLIENT_ID	<Base64 of clientid:secret> of Mobile App client
WATCH_CLIENT_ID	<Base64 of clientid:secret> of wearables
SNAPSHOT_CLIENT_ID	<Base64 of clientid:secret> of snapshot
LOGIN_SCOPE	obdxLoginScope
OFFLINE_SCOPE	urn:opc:ldm:__myscopes__ offline_access

## 2. Adding chatbot support to mobile application (Optional)

CHATBOT_ID	The tenant ID
CHATBOT_URL	The web socket URL for the ChatApp application in IBCS

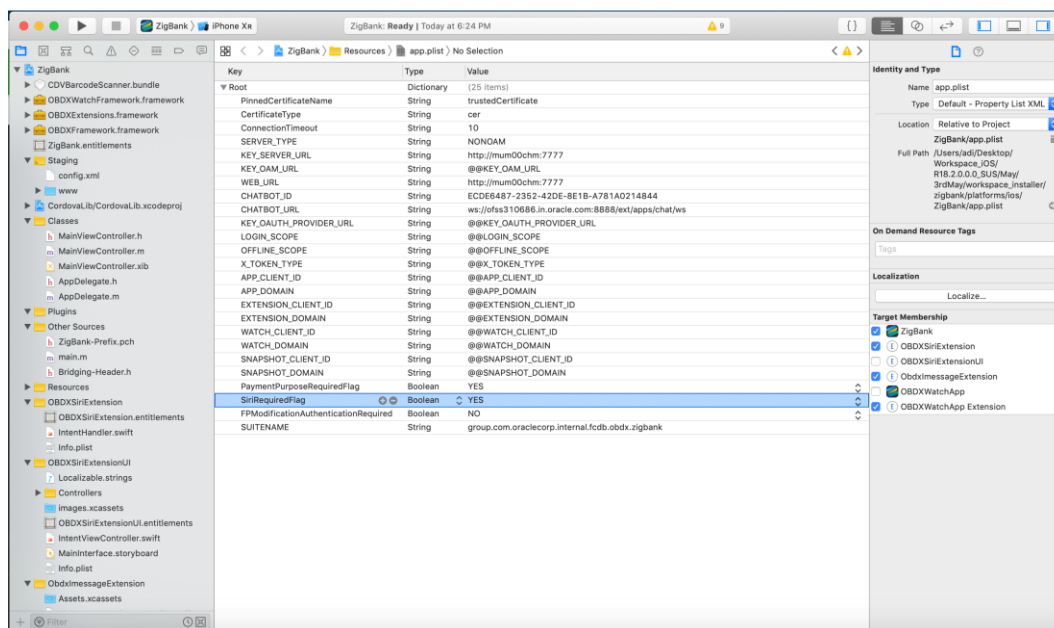
## 3. Common configurations

SUITENAME	Group identifier for sharing keystore information. Same as given in app groups
CertificateType	Extension of SSL Pinned certificates (Eg cer/der)

To use SSL Pinning in the application:

1. Add the certificate file (.cer) into Project's Resource folder. Open the Xcode project, right click on the Resources folder. Select "Add Files to Zigbank". Select the certificate file. Keep "Copy items checked". Also the "Zigbank" target should be selected. After adding the certificate and setting the required SSL parameters in the app.plist as mentioned in the above table, SSL pinning will be enabled. The server url should also point to https URL.

To disable Siri in application (in case bank does not want to use this functionality), disable touch point through admin user and then as per below screenshot turn the SiriRequiredFlag to NO



## Adding Bundle Identifiers

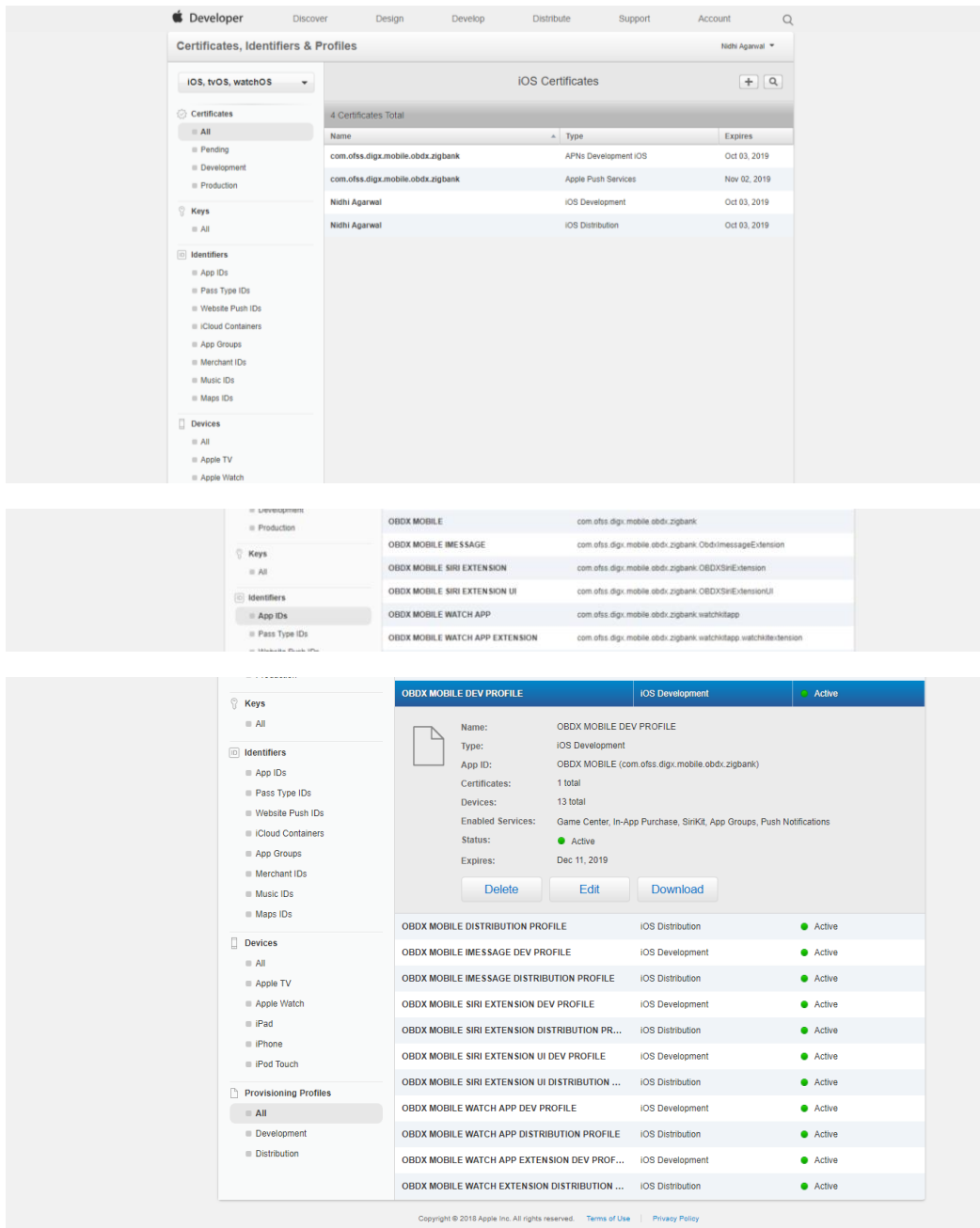
Bundle identifiers need to be added in the Info.plist of each of the frameworks along with the Signing Capabilities tab in Xcode. For example, the bundle identifier used is abc.def.ghi.jkl. The steps to be followed are,

- Right click on OBXFramework.framework (in Xcode's Project Navigator) -> Show in Finder
- When the finder directory opens the right click OBXFramework.framework -> Show package contents.
- Open Info.plist and set Bundle identifier as abc.def.ghi.jkl.OBXFramework
- Repeat the steps for the other three frameworks as well, with the following values:
  - Bundle identifier for Cordova.framework : abc.def.ghi.jkl.Cordova
  - Bundle identifier for OBXExtensions.framework : abc.def.ghi.jkl.OBXExtensions
  - Bundle identifier for OBXWatchFramework.framework : abc.def.ghi.jkl.OBXWatchFramework

## 2.5 Generating Certificates for Development, Production and Push Notifications

Create all certificates (by uploading CSR for keychain utility), provisioning profiles and push certificates as shown below by login in developer console. For development add device UUIDs and add same to provisioning profiles. Add capabilities as shown below and ensure the bundle identifier matches the one of the application in Xcode





Ensure AppGroups capability is added to all profiles and for mobile profile SiriKit, App Groups, Push Notifications must be added.

☐ **Mac Installer Distribution**  
 This certificate is used to sign your app's Installer Package for submission to the Mac App Store.

☐ **Developer ID Installer**  
 This certificate is used to sign your app's Installer Package for distribution outside of the Mac App Store.

☐ **Developer ID Application**  
 This certificate is used to code sign your app for distribution outside of the Mac App Store.

**Services**

☐ **Apple Push Notification service SSL (Sandbox)**  
 Establish connectivity between your notification server and the Apple Push Notification service sandbox environment to deliver remote notifications to your app. A separate certificate is required for each app you develop.

☒ **Apple Push Notification service SSL (Sandbox & Production)**  
 Establish connectivity between your notification server, the Apple Push Notification service sandbox, and production environments to deliver remote notifications to your app. When utilizing HTTP/2, the same certificate can be used to deliver app notifications, update ClockKit complication data, and alert background VoIP apps of incoming activity. A separate certificate is required for each app you distribute.

☐ **macOS Apple Push Notification service SSL (Production)**  
 Establish connectivity between your notification server and the Apple Push Notification service production environment. A separate certificate is required for each app you distribute.

☐ **Pass Type ID Certificate**  
 Sign and send updates to passes in Wallet.

☐ **Website Push ID Certificate**  
 Sign and send updates for Websites.

Note the certificate/bundle name

Apple Developer		
Certificates, Identifiers & Profiles		
Certificates	Certificates +	
Identifiers		
Devices		
Profiles		
NAME	TYPE	PLATFORM
com.ofss.digx.obdx.zigbank	Apple Push Services	iOS

Note the Team ID from top right corner

Navigate to the “Keys” section and create APNS key

Note APNS key and download the .p8 file. Copy the .p8 to config/resources/mobile

Apple Developer

## Certificates, Identifiers & Profiles

[< All Keys](#)

### View Key Details

[Download](#)
[Revoke](#)

**Name**  
 APNSDEV

**Key ID**  
 RBPLJN6ZU5

**Enabled Services**

NAME	CONFIGURATION
Apple Push Notifications service (APNs)	

Update the password as shown below –

Sr. No.	Table	PROP_ID	CATEGORY_ID	PROP_VALUE	Purpose
1	DIGX_FW_CONFIG_ALL_B	APNS	DispatchDetails	<Password>	Provides key of .p8 certificate
2	DIGX_FW_CONFIG_ALL_B	APNSKeyStore	DispatchDetails	DATABASE or CONNECTOR	Specifies whether to pick certificate password from database or from connector. Default DB (No change)
3	DIGX_FW_CONFIG_ALL_B	Proxy	DispatchDetails	<protocol,proxy_address>	Provides proxy address, if any, to be provided while connecting to APNS server. Delete row if proxy not required. Example: HTTP,148.50.60,80
4	DIGX_FW_CONFIG_ALL_B	CERT_TYPE	DispatchDetails	For dev push certs add row with value 'dev'	For prod push certificates this row is not required
5	DIGX_FW_CONFIG_ALL_B	ios_cert_path	DispatchDetails	resources/mobile/AuthKey_RBPLJN6ZU5.p8	Update the certificate path/name if required. Should be relative to config directory
6	DIGX_FW_CONFIG_ALL_B	APNS_BUNDLE	DispatchDetails	Eg. com.ofss.digx.obdx.zigbank	Certificate Name
7	DIGX_FW_CONFIG_ALL_B	APNS_TEAMID	DispatchDetails	Eg. 3NX1974C93	Team ID of Apple developer account

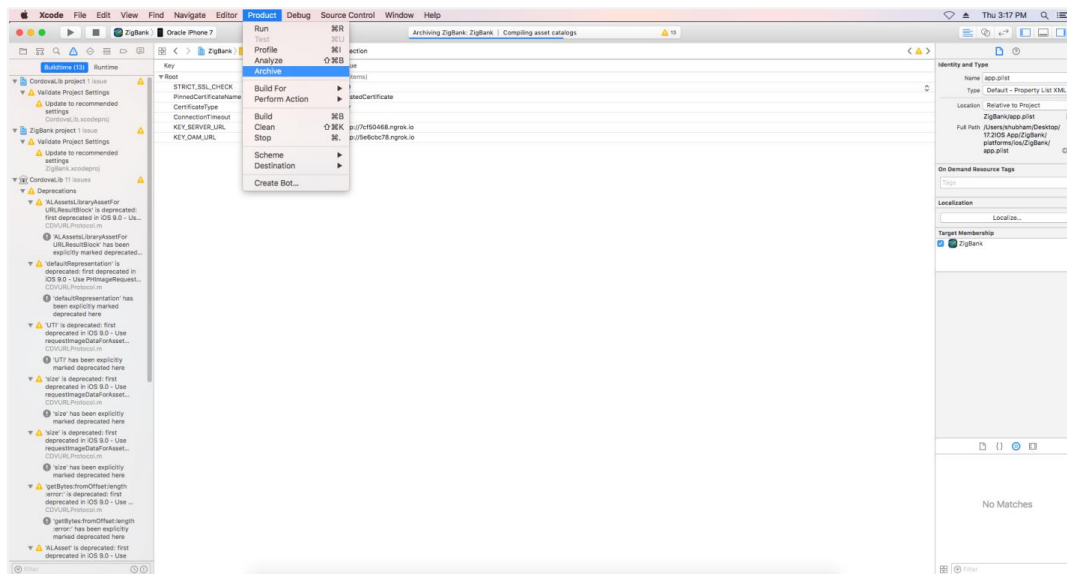
If CONNECTOR is selected in Step 2 update key as below

Properties for tokens to be configured as –

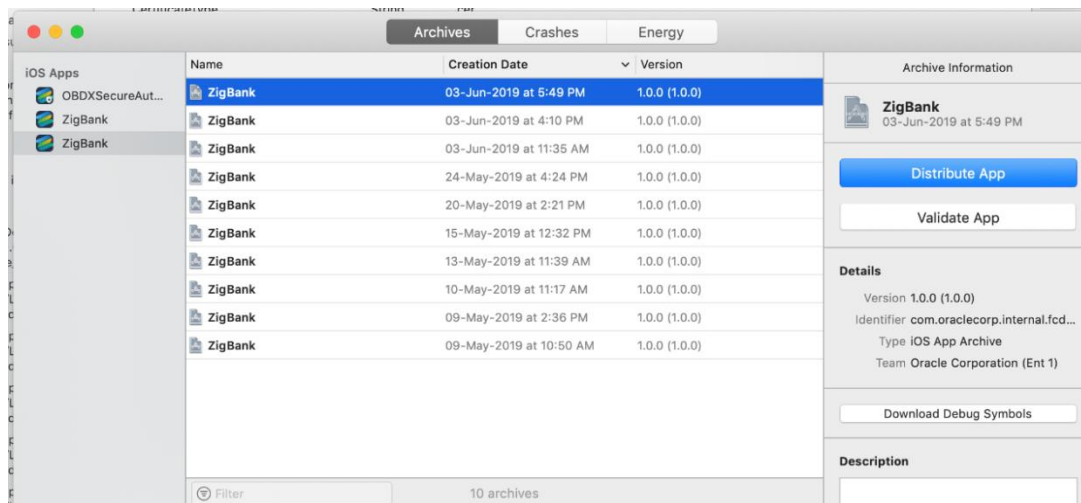
Sr. No	Table	PROP_ID	CATEGORY_ID	PROP_VALUE (Default Value)	Purpose
1	DIGX_FW_CONFIG_ALL_B	APMOBAPP_EXPIRYTIME	authenticationConfig	864000	Time in secs after which user will have to reregister for alternate login in mobile app
2	DIGX_FW_CONFIG_ALL_B	APSNAPSHOT_EXPIRYTIME	authenticationConfig	1296000	Time in secs after which user will have to reregister for snapshot (for mobile app & wearable)
3	DIGX_FW_CONFIG_ALL_B	APWEARABLE_EXPIRYTIME	authenticationConfig	1296000	Time in secs after which user will have to reregister for login in wearables
4	DIGX_FW_CONFIG_ALL_B	APSIRICHATBOT_EXPIRYTIME	authenticationConfig	1296000	Time in secs after which user will have to reregister for Siri (There is no separate registration, it will happen automatically after alternate login is enabled)

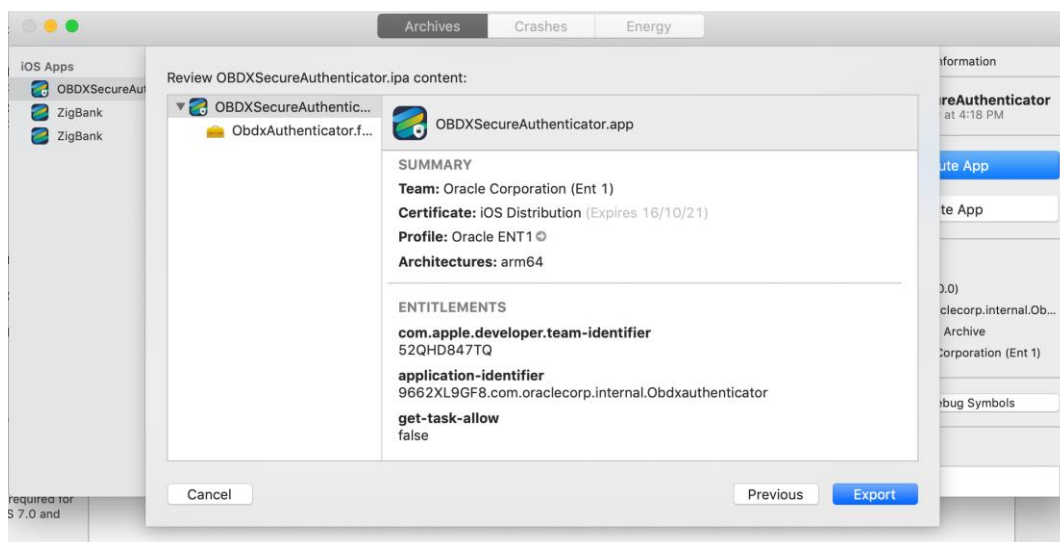
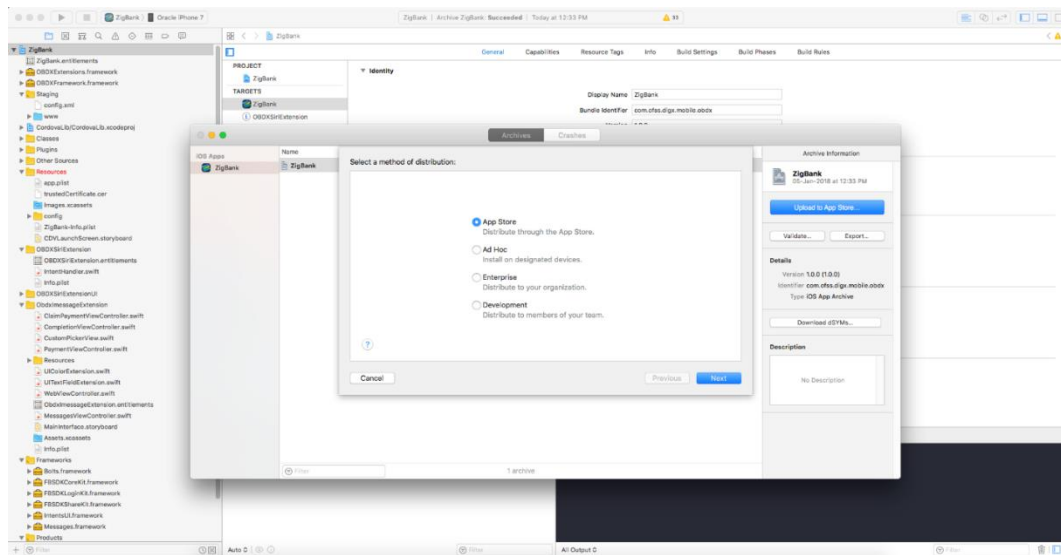
### 3. Archive and Export

- a. In the Menu bar click on **Product -> Archive (Select Generic iOS Device)**



- b. After archiving has successfully completed. Following popup will appear
- c. Click on Export in the right pane of the popup -> Click **Distribute App -> Choose Provisioning Profile -> select Export one app for all Compatible Devices -> Next -> Next** and generate the ipa.

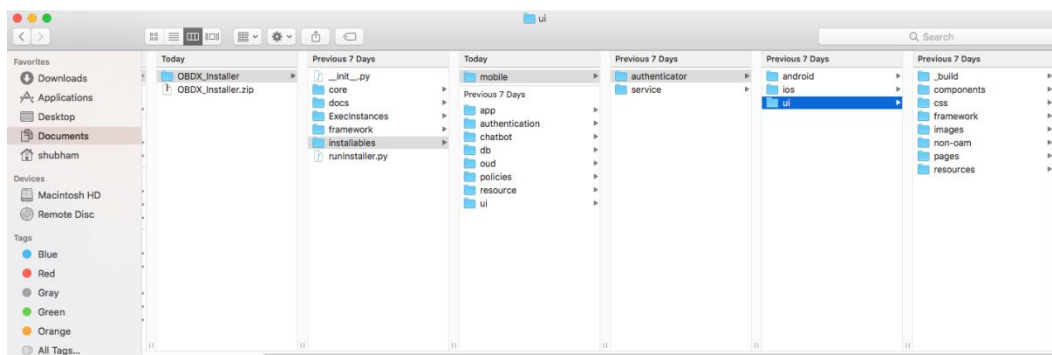




## 4. OBDX Authenticator Application

### 4.1 Building Authenticator UI

1. Extract OBDX\_Installer.zip. It contains **OBDX\_Installer/installables/mobile/authenticator/ui** folder. The folder structure is as shown :



#### (a) OAM based Authentication

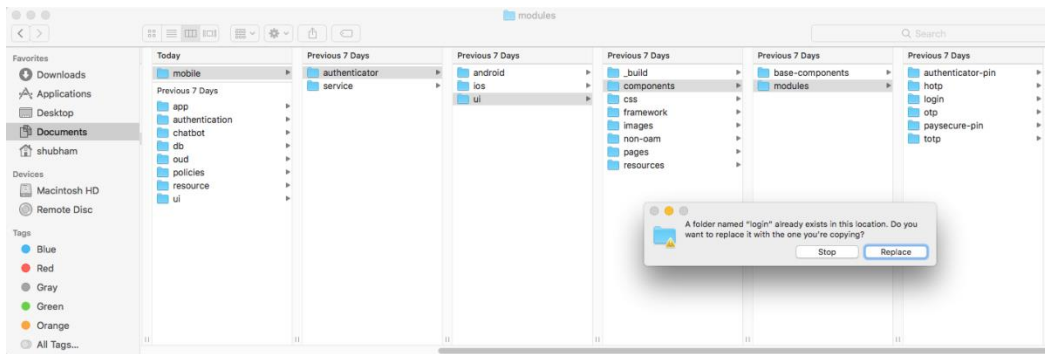
2. Open Terminal at “\_build” level.
3. Run following command :

```
sudo npm install -g grunt-cli
sudo npm install
node render-requirejs/render-requirejs.js
grunt authenticator --verbose
```

4. After running above commands and getting result as “Done, without errors.” a new folder will be created at “\_build” level with name as “dist”.

#### (b) NON-OAM Based Authentication

1. Copy “non-oam/login” folder and Replace it at location “components/modules/” [in ui folder] location. This will replace existing “login” folder.



2. Open Terminal at “\_build” level.

3. Run following command :

```
sudo npm install -g grunt-cli

sudo npm install

node render-requirejs/render-requirejs.js

grunt authenticator --verbose
```

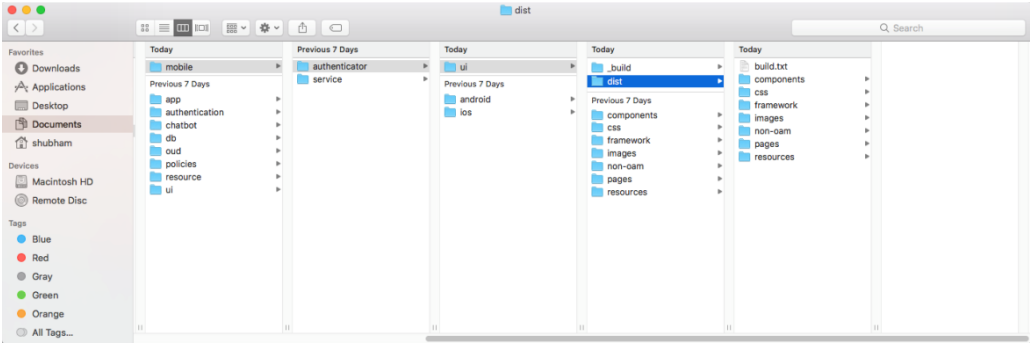
4. After running above commands and getting below result. This error can be ignored and a new folder will be created in “ui” folder with name as “dist”.

```
s/ojcore*]]]
Tracing dependencies for: resources/nls/generic
Tracing dependencies for: jquery
Tracing dependencies for: knockout
Tracing dependencies for: knockout-mapping
Tracing dependencies for: text
Tracing dependencies for: promise
Tracing dependencies for: ojs/ojcore
Tracing dependencies for: css
Tracing dependencies for: ojs/ojknockout
Tracing dependencies for: baseModel
Tracing dependencies for: baseService
Tracing dependencies for: require-config
Error: Parse error using esprima for file: /Users/adi/Desktop/Patchset/18.2/28thJune2019/OBDX_Patch_Installer/installables/mobile/authenticator/unbuilt_ui
/authenticator_ui/dist/cordova.js
TypeError: Cannot read property 'type' of undefined
In module tree:
  require-config
>> Error: Parse error using esprima for file: /Users/adi/Desktop/Patchset/18.2/28thJune2019/OBDX_Patch_Installer/installables/mobile/authenticator/
unbuilt_ui/authenticator_ui/dist/cordova.js
>> TypeError: Cannot read property 'type' of undefined
>> In module tree:
>>   require-config

Execution Time (2019-06-28 15:22:49 UTC+5:30)
loading tasks      16.1s ██████████ 58%
clean:preBuildCleanup 12ms 0%
copy:main          8.7s ██████████ 31%
htmlmin:main       56ms 0%
inlinecss:main      8ms 0%
uglify:updatedBuild 1.3s 5%
string-replace:genericReplacements 15ms 0%
require            3ms 0%
requirejs:compile  1.6s 6%
Total 27.9s

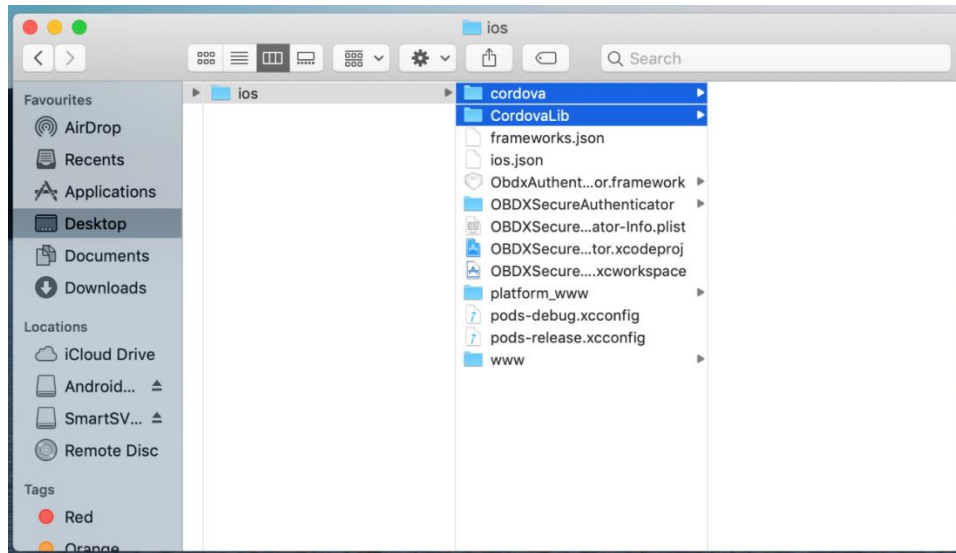
dhcp-10-120-140-254: _build root#
```





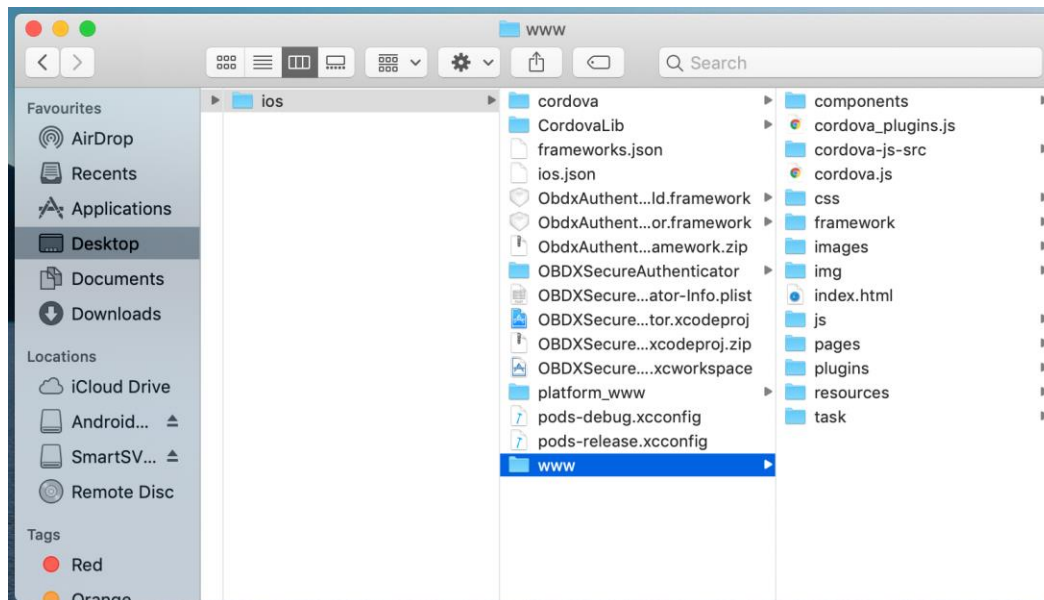
## 4.2 Authenticator Application Workspace Setup

1. Unzip and navigate to iOS workspace as shipped in installer.

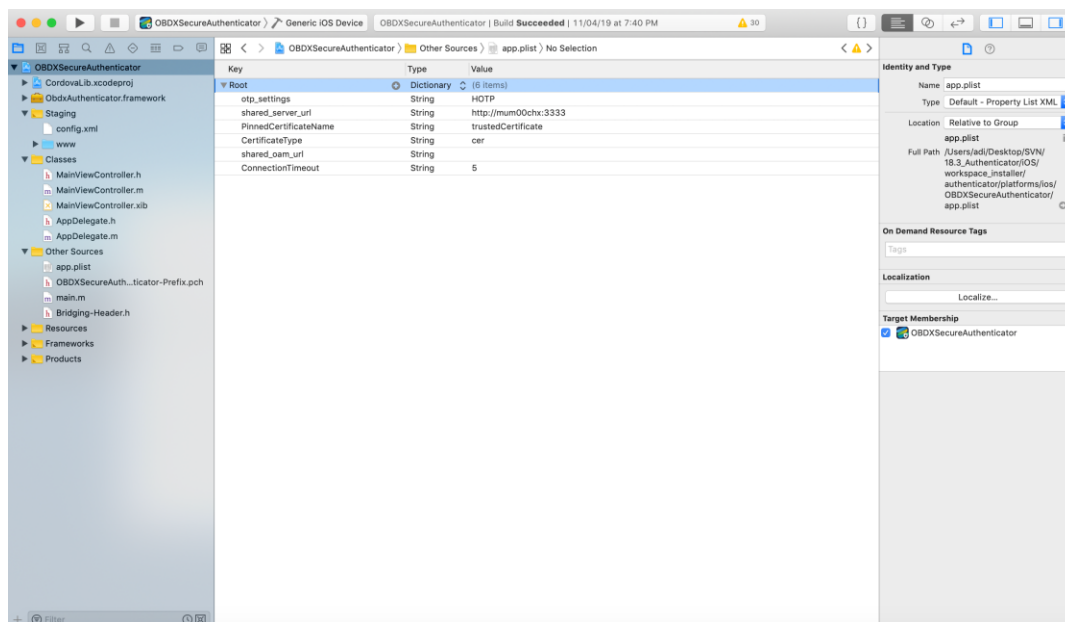


2. Open the “**OBDX\_Installer/installables/mobile/authenticator/ui/ios/www**” folder in the finder and paste and replace the following generated UI files from “*ui/dist*” folder :
  - components
  - css
  - framework
  - images
  - pages
  - resources

Finally the **Installer/installables/mobile/authenticator/ui/ios/www** folder must look like:



3. Double click on **OBDXSecureAuthenticator.xcodeproj** to open the project in Xcode



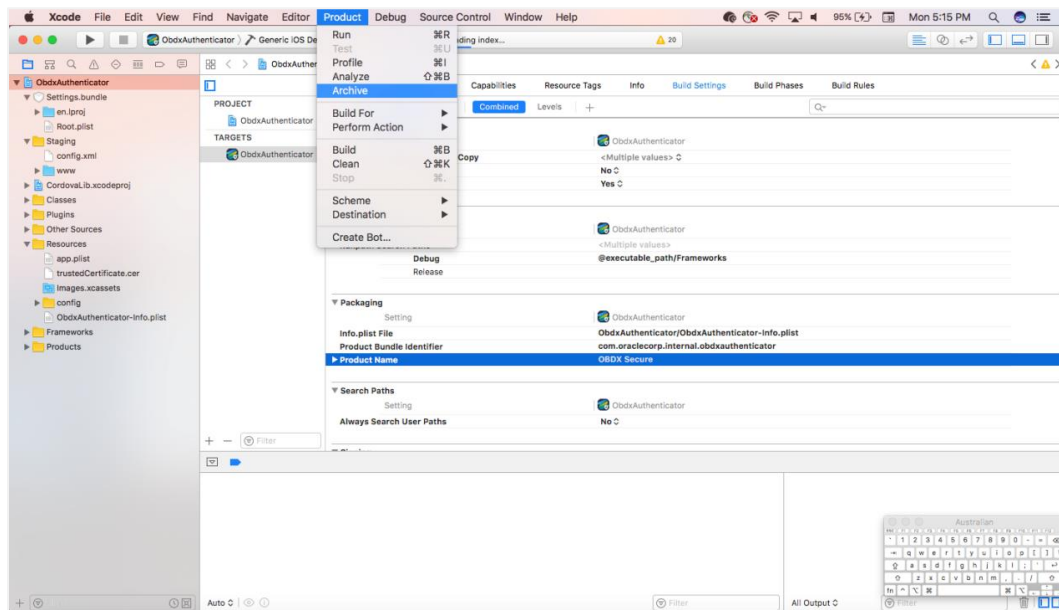
Update HOTP or TOTP in above screenshots and update the server URL.

4. The application can be archived using steps in Section 4.3 for running on device

5. To run the application on simulator, copy & replace the framework from simulator/ObdxAuthenticator.framework to /authenticator/platforms/ios/

## 4.3 Building Authenticator Application

1. Set the simulator to *Generic iOS device*. Then go to *Product -> Archive*.



2. Choose your Archive and then click "*Export*". .ipa file will be generated

