iOS SDK Integration Document

1. Prerequisites:

1. Purchase Mobile Plan

Clients need to purchase mobile membership using server-to-server API calls. (it will be provided via mail)

2. OAuth Token

Get SDK OAuth token using `generateToken Api` (it will be provided via mail)

3. Order UUID

To activate any mobile membership use `Order UUID` from `onboardCustomer Api` (document will be provided via mail)

2. Getting Started

1. Installation

XCode ver: 12.4+

iOS Deployment Target: 12.0

OneAssistSDK for iOS is available at cocoa pods. Add the line below to the Podfile

`pod "OneAssistSDK", "0.0.5"`

and then run `$pod install`

[How to add cocoapods to the iOS project]

Also, add the following permission to the **info.plist** file of the app.

```
<key>NSPhotoLibraryUsageDescription</key>
<string>Need permission for activation</string> // replace with custom
text.
```

Swift version supported: 5.0

iOS Deployment target: 11.0

Third-party dependencies: Alamofire, GoogleMLKit/TextRecognition, DeviceKit, CocoaLumberjack/Swift

2. Usage

Use the below import statement to import the SDK.

```
import OneAssistSDK
```

SDK interface will consist of OAAction, OAActivation, and OAError.

2.1 OAAction is the enum that will represent the stage at which the activation flow is or the next action item.

	OAAction	Description	Next Action

1	initializedSDK	This stage indicates that the SDK has been initialized	call `startActivation` method of OAActivation object
2	fetchDeviceId	This stage indicates that the device id is missing and the client needs to fetch device id to proceed further	call `fetchDeviceId` method of OAActivation object
3	provideUserBasicDetail	This stage indicates that the client needs to provide the basic details.	call `submitBasicDetails` method of OAActivation object
4	sendUploadLink	This stage indicates that the client should send upload link and continue to document upload from another device.	call `sendUploadLink` method of OAActivation object
5	sendUploadLinkOrOpenSecureScreen	This stage indicates that the client can open the secure-screen and capture it or the client should send upload link and continue to document upload from another device.	call `sendUploadLink` method or call `openSecureScreen` method of OAActivation object
6	secureScreenDismissed	This stage indicates that the secure-screen is dismissed, the client can make use of This stage to update the membership status.	the client may wish to refresh the membership status after receiving this action to show the updated status.
7	completed	This stage indicates that activation flow is completed	NA

2.2 OAError is the enum that will represent any error that will occur in the activation process.

	Error	Code	Description
1	ERROR_EMPTY_TOKEN	1001	Token Empty
2	ERROR_EMPTY_ORDER_UUID	1002	Order UUID Empty
3	ERROR_EMPTY_MOBILE_NUMBER	1003	Mobile number is empty
4	ERROR_EMPTY_USER_DETAILS	1004	User details not provided
5	ERROR_EMPTY_PINCODE	1005	Pincode is empty
6	ERROR_EMPTY_ADDRESS	1006	Address is empty
7	ERROR_IMEI_MISSING	1007	IMEI is missing
8	ERROR_SMS_NOT_SENT	1008	Send message on secondary device first
9	ERROR_MEMBERSHIP_IN_PROGRESS	2001	Activation In-Progress
10	ERROR_MEMBERSHIP_PENDING	2002	Activation In-Progress
11	ERROR_MEMBERSHIP_APPROVED	2003	Membership Approved
12	ERROR_MEMBERSHIP_QUEUED	2004	Membership Queued
13	ERROR_MEMBERSHIP_REJECTED	2005	Membership Rejected
14	ERROR_MEMBERSHIP_CANCELLED	2006	Membership Cancelled
15	ERROR_GET_MEMBERSHIP_DETAILS	3001	Unable to get membership detail
16	ERROR_GET_REQ_DOC_DETAILS	3002	Unable to get membership detail
17	ERROR_UPDATE_USER_DETAILS	3003	Unable to update user detail
18	ERROR_SEND_UPLOAD_LINK	3004	Unable to send SMS
19	ERROR_UNABLE_TO_DO_OCR	4001	IMEI not recognized
20	ERROR_PHOTO_PERMISSION	4002	Photo Library permission needed

21	ERROR_SCREEN_RECORDING	4003	Screen recording
22	ERROR_SCREENSHOT_FETCH	4004	Unable to fetch screenshot
23	ERROR_SCREENSHOT_LOAD	4005	Unable to load screenshot
24	ERROR_UNABLE_TO_PROCESS_DESC	4006	Unable to process. try again!
25	ERROR_UNKNOW	5000	Some error occurred. try again!
26	ERROR_API(code: String, description: String)	code	description

2.3 OAActivation class will consist of methods that the client should execute depending on the OAAction as state above in OAAction table.

2.3.1 OAActivationCompletion

Each method of this class will have a completion block in the format below

(_ success: Bool, _ stage: OAAction, _ error: OAError?) -> ()

1. success Bool value that represents whether the method has processed successfully.
2. stage OAAction value that represents the next stage in the activation flow after the successful execution of the method.
3. error OAError value that represents any error which has occurred in the method execution.

2.3.2 OAActivation methods and activation steps

Step 1. Initialization

```
/// This method will set up the SDK with the token generated from
generateToken API.
/// - Parameter token: non-empty token string needed to setup the SDK.
/// - Parameter isDebugMode: optional param, (default false) it
represent is the SDK is to be run in debug mode or not.
/// - Parameter apiEndPoint: optional param, (default nil) client can
provide the custom endpoint here to work in debug mode
/// to redirect all the activation related API calls to that endpoint.
If this is nil,
/// then default endpoint for debug mode will be used.
func initializeSDK(token: String, isDebugMode: Bool = false,
apiEndPoint: String? = nil, completion: OAActivationCompletion)
```

Step 2. Start Activation

```
/// This method will start the activation flow
/// - Parameter orderUUID: order UUID of the pending membership.
func startActivation(for orderUUID: String, completion: @escaping
OAActivationCompletion)
```

Step 3. Fetching Device Id

```
/// This method will fetch the device id from the device and move onto
the next stage of activation.
/// If pincode and address are also provided then user doesn't have to
submit these detail again.
/// - Parameter pincode: optional param, pincode of the user
/// - Parameter address: optional param, address of the user
func fetchDeviceId(pincode: String? = nil, address: String? = nil,
completion: @escaping OAActivationCompletion)
```

Step 4. Submitting Detail

```
/// This method will submit the basic details of the user.
/// - Parameter pincode: pincode of the user
/// - Parameter address: address of the user
func submitBasicDetails(pincode: String, address: String, completion:
@escaping OAActivationCompletion)
```

Step 5: Sending Upload Link

```
/// This method will send the document upload link to secondary device.
/// - Parameters mobileNumber: mobile number of the secondary device,
should not be same as the mobile number with which the membership is
purchased.
func sendUploadLink(_ mobileNumber: String, completion: @escaping
OAActivationCompletion)
```

Step 6: Opening Secure Screen

```
/// This method will open the secure screen.
/// - Parameter viewController: view controller reference on which the
screen will be opened.
func openSecureScreen(onViewController viewController:
UIViewController, completion: @escaping OAActivationCompletion)
```

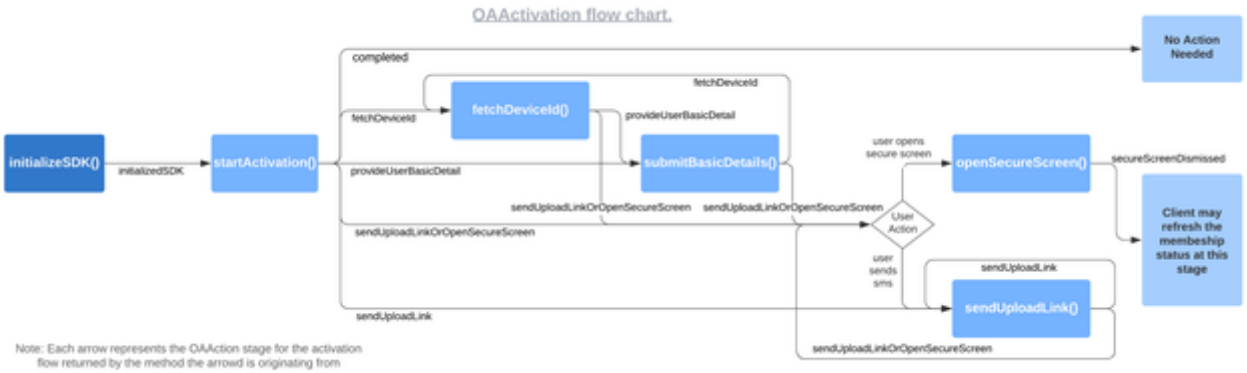
Example:

```
var activationHelper = OAActivation()

// Intialization
activationHelper.initializeSDK(token: AUTH_TOKEN, isDebugMode: true,
```

```
completion: handleSDKCallback)
activationHelper.initializeSDK(token: AUTH_TOKEN, completion:
handleSDKCallback)

// Activation
activationHelper.startActivation(for: ORDER_UUID, completion:
handleSDKCallback)

// Fetch Device Id
activationHelper.fetchDeviceId(pincode: PINCODE, address: ADDRESS,
completion: handleSDKCallback)
activationHelper.fetchDeviceId(completion: handleSDKCallback)

// Submit Basic Detail
activationHelper.submitBasicDetails(pincode: PINCODE, address: ADDRESS,
completion: handleSDKCallback)

// Send Upload link
activationHelper.sendUploadLink(SECONDARY_DEVICE_MOBILE_NUMBER,
completion: handleSDKCallback)

// Open Secure Screen
activationHelper.openSecureScreen(onViewController:
VIEW_CONTROLLER_REF, completion: handleSDKCallback)

// Callback handler
func handleSDKCallback(status: Bool, stage: OAAction, error: OAError?) {
    if status {
        switch stage {
        case .initializedSDK:
            // Client can show UI that will call startActivation or may
call startActivation here if ORDER_UUID exists
        case .fetchDeviceId:
            // Client can show UI that will call fetchDeviceId or call
fetchDeviceId here
        case .provideUserBasicDetail:
            // Client can show UI that will call submitBasicDetails or
call submitBasicDetails here if PINCODE and ADDRESS is already taken
from user.
        case .sendUploadLink:
            // Client will show UI that will call sendUploadLink
        case .sendUploadLinkOrOpenSecureScreen:
            // Client will show UI that will call openSecureScreen and
sendUploadLink
        case .secureScreenDismissed:
            // Client can call membership api to update the membership
status
        case .completed:
            // Activation Flow Completed
        }
```

```
    } else if let error = error {
        let alert = UIAlertController(title: "Error \(error.
errorCode)", message: error.localizedDescription, preferredStyle: .
alert)
        alert.addAction(UIAlertAction(title: "ok", style: .default,
handler: nil))
        present(alert, animated: true, completion: nil)
    }
}
```

Flow Chart:



Extra info:

API endpoint setup in the SDK.

API Endpoint	Environment
https://uat1.1atesting.in/apigateway	Staging
https://api.oneassist.in/apigateway	Production(Default)

Normal UI Flow to use OneAssistSDK:

1. Show membership info to the user with the following actions using membership details:

Membership Status	Possible actions for SDK	UI/UX
`POSTDTLPENDING`, `POSTDTLCOMPLETE`, `REUPLOAD`	Call `startActivation` method after setting up the SDK to check for the state, and perform the action as per the OAAction table.	Show appropriate UI to show the stages at which the membership activation flow is, Show UI to take input for the pincode, address and secondary device mobile number for the corresponding methods. Show UI to show the client some information as to how the device id (IMEI) will be read.
`PENDING`, `CANCELLED`, `REJECTED`, `APPROVED`, `QUEUED`	No action required from the user	Show the current status on the membership card, for `PENDING` case the membership is processing at OneAssist's end.