iOS Lens Receipts

Adding Lens Receipts SDK to your project

Note: The below steps assume that you already have Cocoapods installed and initialized in your project. If you're new to Cocoapods, visit the official site to get started.

1. Open your project's *Podfile*
2. Add the Veryfi private Cocoapods repository as a source at the top of the *Podfile*

```
source 'git@bitbucket.org:veryfi/veryfi-lens-podspec.git'
```

3. Add the VeryfiLens pod to your target

```
pod 'VeryfiLens-Receipts'
```

4. A minimal version of your *Podfile* should look similar to this:

```
source 'git@bitbucket.org:veryfi/veryfi-lens-podspec.git'
source 'https://github.com/CocoaPods/Specs.git'

target 'VeryfiLensExample' do
    use_frameworks!

    # Pods for VeryfiLensExample
    pod 'VeryfiLens-Receipts'
end
```

5. Make sure your SSH key has been granted access to Veryfi's private Cocoapods repository here.

   Also make sure your SSH key has been added to ssh-agent by running this command in the Terminal:

```
# Replace /path/to/private_key with the actual path to your SSH
private key
ssh-add -K /path/to/private_key
```

6. Install the pod by running this command in the Terminal, in the root folder of your project:

```
pod install
```

Configuring your project to use Lens SDK

Add the following permissions to your app's plist:

```
<key>NSCameraUsageDescription</key>
<string>Scan documents</string>
<key>NSPhotoLibraryAddUsageDescription</key>
<string>Access photo gallery for document backups</string>
<key>NSPhotoLibraryUsageDescription</key>
<string>Access photo gallery for document uploads</string>
<key>NSPhotoLibraryAddUsageDescription</key>
<string>Access photo gallery for document backups</string>
<key>NSPhotoLibraryUsageDescription</key>
<string>Access photo gallery for document uploads</string>
<key>NSSpeechRecognitionUsageDescription</key>
<string>Quickly add transactions via voice dictation</string>
<key>NSMicrophoneUsageDescription</key>
<string>Quickly add transactions via voice dictation</string>
<key>NSLocationAlwaysAndWhenInUseUsageDescription</key>
<string>Helps to identify places around you</string>
<key>NSLocationWhenInUseUsageDescription</key>
<string>Helps to identify places around you</string>
<key>NSContactsUsageDescription</key>
<string>Add your ___@veryfi.cc assigned email address for reference<
/string>
<key>NSCalendarsUsageDescription</key>
<string>Enrich your data with business meetings and events from your
Calendar</string>
```

Initializing Lens

1. Import required symbols from Lens SDK:

```
import VeryfiLens
```

2. Configure your authentication credentials:

```
let CLIENT_ID = "XXX" // replace XXX with your assigned Client Id
let AUTH_USERNAME = "XXX" // replace XXX with your assigned
Username
let AUTH_APIKEY = "XXX" // replace XXX with your assigned API Key
let URL = "XXX" // replace XXX with your assigned Endpoint URL

let veryfiLensCredentials = VeryfiLensCredentials(clientId:
CLIENT_ID,
                                                  username:
AUTH_USERNAME,
                                                  apiKey:
AUTH_APIKEY,
                                                  url: URL)
```

3. Configure your Lens settings. Refer to the full list of available settings later in this section.

```
let veryfiLensSettings = VeryfiLensSettings()
veryfiLensSettings.documentTypes = ["receipt", "bill"]
veryfiLensSettings.showDocumentTypes = true
```

4. Enable background upload support. Add the below to your AppDelegate:

```
import AWSS3

func application(_ application: UIApplication,
handleEventsForBackgroundURLSession identifier: String,
completionHandler: @escaping () -> Void) {
    // Store the completion handler
    AWSS3TransferUtility.interceptApplication(application,
handleEventsForBackgroundURLSession: identifier,
completionHandler: completionHandler)
}
```

5. Initialize Lens:

```
VeryfiLens.shared().configure(with: veryfiLensCredentials,
settings: veryfiLensSettings)
```

Available settings:

`autoCropGalleryIsOn`: forces document detection and auto cropping on documents imported from the image gallery (default: *false*)

`autoDeleteAfterProcessing`: if on, scanned files will be deleted once processing has completed (default: *false*)

`autoDocDetectionAndCropIsOn`: detects, highlights and crops documents automatically during camera image capture (default: *true*)

`autoLightDetectionIsOn`: if on the room ambience controls light to illuminate the document. Turn OFF for manual controls (default: *true*)

`autoRotateIsOn`: automatically rotates image so the contained document is correctly oriented (default: *false*)

`autoSubmitDocumentOnCapture`: auto submit document on capture, skipping the preview screen (default: *false*)

`backupDocsToGallery`: uses photo gallery to backup each scans -- NOTE: must ask user for permission (default: *true*)

`blurDetectionIsOn`: checks if a picture captured has 20% or more blur - blurred receipts don't process well (default: *true*)

`brandImage`: specifies a custom image to use in the long receipt preview guide icon. Simply add your image bundle to your app and specify it in this setting e.g. *veryfiLensSettings.brandImage = UIImage(named: "Veryfi-Logo")* (default: *nil*)

`categories`: optional list of custom categories for Veryfi to use in categorizing submitted documents (default: *null*)

`closeCameraOnSubmit`: after submitting an image, the Lens camera view will be closed and user returned to the host app (default: *true*)

`dictateIsOn`: enables/disables the "Add by voice" option (default: *true*)

`docDetectFillUIColor`: document detection rectangle fill color (default: *"#9653BF8A"*)

`docDetectStrokeUIColor`: document detection rectangle stroke color (default: *null*)

`documentTypes`: only available when **showDocumentTypes** is set to *true*. List of document types to collect, with a user selection to switch between specified types. Allowed values are: "long_receipt", "receipt", "bill", "other" (default: *null*):

- *long_receipt* - allows user to capture a long receipt by scanning it from top to bottom. Multiple frames are captured and stitched together, similar to a panorama photo, except in vertical orientation
- *receipt* - standard camera experience, whereby documents will be detected as either a receipt or an invoice
- *bill* - standard camera experience, documents will be marked as bill

`emailCCDomain`: the domain name used to power emailed documents (default: *"*veryfi.cc*"*)

`emailCCIsOn`: enables/disables the email cc view inside settings (default: *true*)

`externalId`: a pass-through field to add a unique reference identifier for a scan which can be used to map back to your system (default: *""*)

`galleryIsOn`: enables/disables the photo gallery feature (default: *true*)

`locationServicesIsOn`: enables/disables location services to grab user's lat & lng (default: *true*)

`manualCropIsOn`: toggles the option to manually crop an image before submitting it for processing (default: *true*)

`moreMenuIsOn`: enables/disables the showing of the more menu (default: *true*)

`moreSettingsMenuIsOn`: enables/disables the showing of the More > Settings option. NOTE: When this is FALSE all Settings come from the app, not the user (default: *true*)

`originalImageMaxSizeInMB`: maximum size in MB applied when producing images. Valid range is: 0.2 to 2.5 (default: *2.5*)

`returnStitchedPDF`: provides path of stitched PDF (when multiple images are stitched for a single document) in the **veryfiLensUpdate** delegate function (default: *false*)

`rotateDocIsOn`: enables option to rotate (on each press) a document by 90 degree clockwise (default: *true*)

`saveLogsIsOn`: stores logs on device. Recommended to be enabled to aid with debugging if required (default: *true*)

`shareLogsIsOn`: enables option on preview screen to share logs for debugging. Recommended to be disabled in production (default: *false*)

`shieldProtectionIsOn`: adds shield icon to capture button and adds an menu option inside More > What is Shield? (default: *true*)

`showDocumentTypes`: enables/disables the documentTypes setting. When disabled, the default camera experience will be used and all documents will be treated as either a receipt, invoice or bill (auto-detected) (default: *false*)

`stitchIsOn`: enables/disables the option to combine multiple receipts together into a PDF (default: *true*)

`stitchedPDFPixelDensityMultiplier`: multiplier for the image resolution being drawn on the PDF. Valid range is: 1.0 to 5.0 (default: *2.0*)

`submitButtonBackgroundColor`: overrides the background color for the document submit button

`submitButtonBorderColor`: overrides the border color for the document submit button

`submitButtonCornerRadius`: overrides the corner radius for the document submit button

`submitButtonFontColor`: overrides the font color for the document submit button

NOTE: In the case when the settings menu is disabled for the user (`moreSettingsMenuIsOn` is set to *false*), Lens will use the settings that it is initialized with. If the settings menu is enabled, the user will by default be presented with the configured values, but will be able to change these within the settings menu.

Launching Lens

1. Launch the Lens camera:

```
VeryfiLens.shared().showCamera(in: self)
```

2. Optionally, instead of launching the Lens camera experience, you may perform one of the following:
   - Launch the Gallery screen to submit images from the Photo Gallery:

```
VeryfiLens.shared().showGallery(in: self)
```

   - Launch the Dictation screen to submit images either typed or dictated by voice:

```
VeryfiLens.shared().showDictation(in: self)
```

   - Open email collection instructions screen, including the email address the user should forward documents to:

```
VeryfiLens.shared().showEmail(in: self)
```

   TIP: Collection email addresses can either exist on the veryfi.cc domain, or your own whitelisted domain. Contact support@veryfi.com for details on whitelabeling.
3. For all of the above features, with the exception of *showEmail()*, your app will need to communicate with Lens to handle user actions, various status changes and extraction results from Veryfi. See the Communicating with Lens section below for details

Communicating with Lens

1. Set your delegate:

```
VeryfiLens.shared().delegate = self
```

2. Implement the delegate methods:

```
extension ViewController: VeryfiLensDelegate {
    func veryfiLensClose(_ json: [String : Any]!) {
        let jsonData = try? JSONSerialization.data(withJSONObject:
json as Any, options: .prettyPrinted)
        let jsonString = String(data: jsonData!, encoding: .utf8)
        print(String("veryfiLensClose: " + jsonString!))   // do
something with the JSON here
    }

    func veryfiLensUpdate(_ json: [String : Any]!) {
        let jsonData = try? JSONSerialization.data(withJSONObject:
json as Any, options: .prettyPrinted)
        let jsonString = String(data: jsonData!, encoding: .utf8)
        print(String("veryfiLensUpdate: " + jsonString!))   // do
something with the JSON here
    }

    func veryfiLensSuccess(_ json: [String : Any]!) {
        let jsonData = try? JSONSerialization.data(withJSONObject:
json as Any, options: .prettyPrinted)
        let jsonString = String(data: jsonData!, encoding: .utf8)
        print(String("veryfiLensSuccess: " + jsonString!))   // do
something with the JSON here
    }

    func veryfiLensError(_ json: [String : Any]!) {
        let jsonData = try? JSONSerialization.data(withJSONObject:
json as Any, options: .prettyPrinted)
        let jsonString = String(data: jsonData!, encoding: .utf8)
        print(String("veryfiLensError: " + jsonString!))   // do
something with the error JSON here
    }
}
```

Delegate Definitions

- `veryfiLensClose` - the Veryfi Lens camera has been closed, either as a result of submitting an image for processing, or the user closed the camera without submitting an image.

  Sample data:

```
{
    "status": "close",
    "queue_count": 1,
    "framework-version": "1.4.0",
    "session_scan_count": 1,
    "framework-build": "1"
}
```

NOTE: In the object above, `queue_count` refers to the number of submitted documents that are currently in the processing queue. `session_scan_count` refers to the number of documents that were submitted in the most recent Lens camera session - if this is equal to 0 (zero) then the camera session was canceled without submitting anything.

- `veryfiLensUpdate` - during the processing of a document, this delegate will be fired multiple times. One time it will contain the thumbnail path for the submitted document and one time it will contain a full-size image path. In addition, multiple instances of this delegate will be fired containing the current upload progress percentage.

Sample *package created* notification:

```
{
    "status": "start",
    "package_id": "edc8653e4c2b4ef1"
}
```

Sample *thumbnail path* notification:

```
{
    "status": "inprogress",
    "msg": "img_thumbnail_path",
    "data": "/path/to/thumbnail.jpg",
    "package_id": "edc8653e4c2b4ef1"
}
```

Sample *full-size image path* data:

```
{
    "status": "inprogress",
    "msg": "img_original_path",
    "data": "/path/to/image.jpg",
    "package_id": "edc8653e4c2b4ef1",
    "document_type": "receipt"
}
```

Sample *upload progress percentage* data:

```
{
    "status": "inprogress",
    "msg": "progress",
    "data": 68,
    "package_id": "edc8653e4c2b4ef1"
}
```

Sample *package removed* notification data:

```
{
    "status": "removed",
    "msg": "clear_package",
    "package_id": "edc8653e4c2b4ef1"
}
```

- `veryfiLensError` - if an error occurs during uploading or processing a submitted or a general exception or crash is caught in Veryfi Lens, this notification contains the error details.

  Sample *error* data:

  ```
  {
      "status": "error",
      "package_id": "edc8653e4c2b4ef1",
      "error": "[Wombat].Reachability.noInternetConnection"
  }
  ```

- `veryfiLensSuccess` - this delegate fires once a document has finished processing, whether it was submitted via the camera, the gallery, or it was dictated or entered/typed manually. This delegate provides the response from the Veryfi API.

  Sample data:

```
{
    "status": "done",
    "package_id": "edc8653e4c2b4ef1",
    "data": {"abn_number": "", "account_number": "",
"bill_to_address": "", "bill_to_name": "", "card_number": "",
"category": "Meals & Entertainment", "created": "2021-01-14 05:
19: 51", "currency_code": "USD", "date": "2021-01-14 05: 19: 51",
"delivery_date": "", "discount": 0, "due_date": "", "external_id":
"", "id": 31428417, "img_file_name": "xxxxxxxxx.png",
"img_thumbnail_url": "https: \/\/cdn.veryfi.com\/partner-receipts\
/xxxxxxxxx_t.png", "img_url": "https: \/\/cdn.veryfi.com\/partner-
receipts\/xxxxxxxxx.png", "incoterms": "", "insurance": "",
"invoice_number": "", "line_items": [], "notes": "", "ocr_text":
"starbucks 23.4", "order_date": "", "payment_display_name": "",
"payment_terms": "", "payment_type": "", "phone_number": "",
"purchase_order_number": "", "reference_number": "VBDEC-0000",
"service_end_date": "", "service_start_date": "", "ship_date": "",
"shipping": 0, "subtotal": 23.4, "tags": [], "tax": 0, "tip": 0,
"total": 23.4, "total_weight": "", "vat_number": "", "vendor":
{"address": "", "fax_number": "", "name": "Starbucks",
"phone_number": "", "raw_name": "Starbucks", "vendor_logo":
"https: \/\/cdn.veryfi.com\/logos\/us\/910419611.png",
"vendor_type": ""}}
}
```


Sample *dictated expense* data:

```
{
    "status": "done",
    "package_id": "edc8653e4c2b4ef1",
    "data": {"abn_number": "", "account_number": "",
"bill_to_address": "", "bill_to_name": "", "card_number": "",
"category": "Meals & Entertainment", "created": "2021-01-14 05:
19: 51", "currency_code": "USD", "date": "2021-01-14 05: 19: 51",
"delivery_date": "", "discount": 0, "due_date": "", "external_id":
"", "id": 31428417, "img_file_name": "xxxxxxxxx.png",
"img_thumbnail_url": "https: \/\/cdn.veryfi.com\/partner-receipts\
/xxxxxxxxx_t.png", "img_url": "https: \/\/cdn.veryfi.com\/partner-
receipts\/xxxxxxxxx.png", "incoterms": "", "insurance": "",
"invoice_number": "", "line_items": [], "notes": "", "ocr_text":
"starbucks 23.4", "order_date": "", "payment_display_name": "",
"payment_terms": "", "payment_type": "", "phone_number": "",
"purchase_order_number": "", "reference_number": "VBDEC-0000",
"service_end_date": "", "service_start_date": "", "ship_date": "",
"shipping": 0, "subtotal": 23.4, "tags": [], "tax": 0, "tip": 0,
"total": 23.4, "total_weight": "", "vat_number": "", "vendor":
{"address": "", "fax_number": "", "name": "Starbucks",
"phone_number": "", "raw_name": "Starbucks", "vendor_logo":
"https: \/\/cdn.veryfi.com\/logos\/us\/910419611.png",
"vendor_type": ""}},
    "type":"dictation"
}
```


Prepare your app for the App Store

1. Go to your app's *Target* > *Build Phases*

   Make sure VeryfiLens.xcframework is included in the *Embedded Frameworks* section

Please note: adding the iOS Lens Receipts SDK to your app will increase your final app size by up to 17 MB. This is due to machine learning models, support libraries, etc included in the SDK.