Web Solution

Version: V4

Release Notes

We're constantly working to improve our OCR AI products.

Release notes list all of the changes that are introduced by each version. Use them to check product enhancements and what changes you might need to make before you migrate your app to any given SDK version.

SDK v.4.2.1-b01

October 9th, 2025

Fixed issues

Resolved a CUI validation bug that prevented certain Peruvian ID cards from being properly scanned during camera capture.

Known issues

None

Breaking changes

None

Upgrade procedure

Two upgrade procedures are available, depending on your architecture. Please follow the one that matches your specific design:

1. OCR Web SDK + IDENTY Web Server, secure decryption on server

Only the OCR SDK component requires an update. Please follow these step-by-step instructions:

Web OCR SDK

Set the new SDK packages in your Web Application

"@identy/identy-common": "5.0.0"

"@identy/identy-ocr": "4.2.1-b01E"

Kindly be aware that certain adjustments are required in the .npmrc file prior to upgrading to this SDK:

Eliminate the initial entry, registry=https://identy.jfrog.io/identy/api/npm/identy-npm/, as it is no longer necessary.

Remove any yarn.lock file present in your project to ensure the upgrade takes effect.

Run *"yarn install --update-checksums"* to update packages.

Please note that each time a license is downloaded from the License Manager, it may differ due to encryption. When deploying the SDK and IWS, ensure that the same BASE64 string is used on both sides. Otherwise, you may encounter a license error.

IDENTY Web Server

- Open a console and navigate to the folder where the *docker-compose.yml* lives.
- Stop container via docker-compose down.
- Edit docker-compose.yml file to use latest version:

```
identy-docker.jfrog.io/identy_biometrics_tomcat:620.510.131.311.1
```

2. OCR Web SDK, local browser capture and processing

This upgrade affects only the OCR SDK component. Please follow these step-by-step instructions:

Set the new SDK packages in your Web Application

"@identy/identy-common": "5.0.0"

"@identy/identy-ocr": "4.2.1-b01"

Kindly be aware that certain adjustments are required in the .npmrc file prior to upgrading to this SDK:

Eliminate the initial entry, registry=https://identy.jfrog.io/identy/api/npm/identy-npm/, as it is no longer necessary.

Remove any yarn.lock file present in your project to ensure the upgrade takes effect.

Run *"yarn install --update-checksums"* to update packages.

IWS 4.2.0

August 14th, 2025

What's new

1. IDENTY WEB SERVER NOW SUPPORTS OCR PROCESSING
A new OCR architecture is now available, introducing server endpoints that accept images from external systems, perform ocr processing, and return extracted data in structured JSON format.
This approach works without the OCR Web SDK, allowing images from internal apps, third-party platforms, or legacy systems to be sent directly to the server.
Ideal for backend automation and smooth integration into existing workflows.

Refer to the documentation for the new `ProcessCard` and `ProcessPoa` endpoints for detailed technical information.

Fixed issues

None

Known issues

None

Breaking changes.

- Open a console and navigate to the folder where the *docker-compose.yml* lives.
- Stop container via docker-compose down.
- Edit docker-compose.yml file to use latest version:

```
identy-docker.jfrog.io/identy_biometrics_tomcat:710.611.131.420.1
```

SDK v.4.2.0-b01

August 1st, 2025

What's new

1. SUPPORT FOR THE NEW PERU 2025 DNI
Handled using the existing `PERU_DNI` Enum.
2. ADDITIONAL MANDATORY FIELDS FOR PERU DNI SCANS
Includes the CUI extracted from backside barcode and well-formed MRZ from the backside.
3. SUPPORT FOR CAMEROON ID CARD
Handled using the existing `CAMEROON_ID` Enum.

4. A4.CSF NOW SUPPORTS PROCESSING INCOMPLETE PDFs
Only the first page is mandatory.
5. COLOMBIA: SUPPORT FOR PERMISO DE PROTECCIÓN TEMPORAL (PPT)
Supported by a dedicated ID type enum named `COLOMBIA_PPT` .

6. ENHANCED ACCURACY IN EXTRACTING THE "CLAVE DE ELECTOR" FIELD FROM INE DOCUMENTS
7. SUPPORT FOR COSTA RICA ID CARDS
Via the new `COSTA_RICA.ID` Enum.
8. SUPPORT FOR PANAMA ID CARDS
Via the new `PANAMA_ID` Enum.
9. SUPPORT FOR PANAMA ELECTORAL ID SIM
Via the new `PANAMA_ELECTORAL_ID` Enum.
10. SUPPORT FOR PANAMA DRIVING LICENSE ID SIM
Via the new `PANAMA_DRIVING_LICENSE` Enum.
11. SUPPORT FOR BUTHAN ID CARDS
Via the new `BUTHAN.ID` Enum.
12. SUPPORT FOR AIRTED SIM
Via the new `MALAYSIA_PASSPORT` Enum.
13. IMPROVED BARCODE RECOGNITION FOR KSA NATIONAL ID
14. PAKISTAN CNIC NOW INCLUDE EITHER FATHER´S NAME OR HUSBAND´S NAME + "LIFETIME" DATE OF EXPIRY
Since these two fields are mutually exclusive, only one will be populated while the other remains empty.
15. IMPROVED COLOMBIA ID LAYOUT VALIDATION
Scanning now checks that the ID card matches the expected format for better accuracy.
16. THE SDK NOW EXPOSES A NEW FIELD INDICATING WHETHER THE DETECTED FACE IMAGE IS COLOR OR BACK & WHITE
This is available throught the new `faceIsGrey` field

Fixed issues

QR code is now correctly extracted from the back side of Malawi ID cards.

Correction applied to "Country of Stay" fixed on PAK cards.

Known issues

None

Breaking changes.

Colombia cards: The attributes `Fecha Y Lugar De Expedicion` , and `Fecha De Nacimiento` are now cammel case as part of standardization efforts.

MEX_PASSPORT: The attributes `"Attribute 1"` and `"Attribute 2"` are no longer included in the JSON response for camera captures.

Upgrade procedure

Two upgrade procedures are available, depending on your architecture. Please follow the one that matches your specific design:

1. OCR Web SDK + IDENTY Web Server, secure decryption on server

Only the OCR SDK component requires an update. Please follow these step-by-step instructions:

Web OCR SDK

Set the new SDK packages in your Web Application

"@identy/identy-common": "4.0.1"

"@identy/identy-ocr": "4.2.0-b01E"

Kindly be aware that certain adjustments are required in the .npmrc file prior to upgrading to this SDK:

Eliminate the initial entry, registry=https://identy.jfrog.io/identy/api/npm/identy-npm/, as it is no longer necessary.

Remove any yarn.lock file present in your project to ensure the upgrade takes effect.

Run *"yarn install --update-checksums"* to update packages.

Please note that each time a license is downloaded from the License Manager, it may differ due to encryption. When deploying the SDK and IWS, ensure that the same BASE64 string is used on both sides. Otherwise, you may encounter a license error.

IDENTY Web Server

- Open a console and navigate to the folder where the *docker-compose.yml* lives.
- Stop container via docker-compose down.
- Edit docker-compose.yml file to use latest version:

```
identy-docker.jfrog.io/identy_biometrics_tomcat:620.510.131.311.1
```

2. OCR Web SDK, local browser capture and processing

This upgrade affects only the OCR SDK component. Please follow these step-by-step instructions:

Set the new SDK packages in your Web Application

Eliminate the initial entry, registry=https://identy.jfrog.io/identy/api/npm/identy-npm/, as it is no longer necessary.

Remove any yarn.lock file present in your project to ensure the upgrade takes effect.

Run *"yarn install --update-checksums"* to update packages.

SDK v.4.02-b01

February 27th, 2025

What's new

1. IMPROVED EFFICIENCY IN BLACK AND WHITE CATEGORIZATION
For both, front and back sides.
2. SUPPORT FOR GUATEMALA ID
Via the new `GUATEMALA.ID` Enum.
3. ADDITIONAL FIELDS NOW EXTRACTED FROM SENEGAL ID CARDS
Including NIN, region, and commune from the back side as mandatory fields
4. IMPROVED OCR EXTRACTION FOR RWANDAN IDs

Fixed issues

None

Known issues

None

Breaking changes.

None

Upgrade procedure

Two upgrade procedures are available, depending on your architecture. Please follow the one that matches your specific design:

1. OCR Web SDK + IDENTY Web Server Architecture

Only the OCR SDK component requires an update. Please follow these step-by-step instructions:

Web OCR SDK

Set the new SDK packages in your Web Application

"@identy/identy-common": "4.0.1"

"@identy/identy-ocr": "4.0.2-b01E"

Kindly be aware that certain adjustments are required in the .npmrc file prior to upgrading to this SDK:

Eliminate the initial entry, registry=https://identy.jfrog.io/identy/api/npm/identy-npm/, as it is no longer necessary.

Remove any yarn.lock file present in your project to ensure the upgrade takes effect.

Run *"yarn install --update-checksums"* to update packages.

Please note that each time a license is downloaded from the License Manager, it may differ due to encryption. When deploying the SDK and IWS, ensure that the same BASE64 string is used on both sides. Otherwise, you may encounter a license error.

2. OCR Web SDK Architecture

This upgrade affects only the OCR SDK component. Please follow these step-by-step instructions:

Set the new SDK packages in your Web Application

"@identy/identy-common": "4.0.1"

"@identy/identy-ocr": "4.0.2-b01"

Kindly be aware that certain adjustments are required in the .npmrc file prior to upgrading to this SDK:

Eliminate the initial entry, registry=https://identy.jfrog.io/identy/api/npm/identy-npm/, as it is no longer necessary.

Remove any yarn.lock file present in your project to ensure the upgrade takes effect.

Run *"yarn install --update-checksums"* to update packages.

SDK v.4.0.0-b01

December 6th, 2024

What's new

1. OCR EXTRACTION ACCURACY SIGNIFICANTLY IMPROVED
Across all supported card models and POA documents in both, Camera and Gallery captures.
We strongly recommend that all customers upgrade to the latest version at their earliest convenience to begin benefiting from this enhancement.
2. ENHANCED PDF PROCESSING
Improved data extraction accuracy from PDF documents during Gallery processing.

3. ENHANCED EXTRACTION FOR POA DOCUMENTS WITH ANNOTATIONS
Improved accuracy for POA documents that have been scanned with manual annotations, ensuring reliable data capture even in annotated or altered documents.

4. MANDATORY FIELDS VALIDATION IN JSON
JSON responses now include a `mandatoryFields` section with Boolean values indicating whether each mandatory field was extracted correctly. This complements the existing hasQuality field by identifying specific culprits when hasQuality is false.
For more details, please see the Returned JSON Structure section.

5. IMPROVED ACCURACY IN PERUVIAN DOCUMENT CLASSIFICATION
Ensuring accurate model identification and optimized OCR extraction.

6. IMPROVED ACCURACY IN PAKISTAN DOCUMENT CLASSIFICATION
Ensuring accurate model identification and optimized OCR extraction.

9. UPGRADE JQUERY-UI DEPENDENCY TO 1.14.1
10. SUPPORT FOR PERUVIAN CDE: CÉDULA DE EXTRANJERIA
Supporting the three layout variants of the CDE card, via the new `PERU_CDE` Enum.
11. SUPPORT FOR BANK CREDIT CARDS
Allowing OCR extraction of the backside, including a readable QR code, via the new `MEXICO_BCC` Enum.

12. SUPPORT FOR UGANDA ID CARDS
Via the new `UGANDA_ID` Enum.

13. SUPPORT FOR RWANDA ID CARDS
Via the new `RWANDA_ID` Enum.

14. SUPPORT FOR MALAYSIA ID CARDS
Via the new `MALAYSIA_PASSPORT` Enum.

15. SUPPORT FOR ID_MRZ_TYPE1_FRONT ID CARDS
Via the new `UNIVERSAL.ID_MRZ_TYPE1_FRONT` Enum, mirroring `ID_MRZ_TYPE1` , but is designed to expect the MRZ on the front side of the document. It does not require a face image or any data on the back side.
16. SUPPORT FOR PNG AND JPEG_95 TEMPLATE FORMATS
The SDK now supports `PNG` and `JPEG_95` template formats in addition to existing formats.

Fixed issues

Date of birth, issue date, and expiry date are now accurately processed for Pakistan cards.

Barcodes on the front side of KSA Alien documents are now correctly extracted.

The SDK no longer includes postal code fields with fewer than 5 digits or characters.

Known issues

On low-end Android devices, the camera capture of CFE POA may incorrectly crop the A4 document.

Breaking changes.

Mandatory fields for PAK cards are now ID Number and DOB.

The POA Telmex date format is now always returned as "dd mm yyyy," regardless of the format in the document.

Argentina 2009 back “Fecha Y Lugar de Nacimiento” is split into “Fecha de Nacimiento” and “Lugar de Nacimiento” separate fields.

Upgrade procedure

Two upgrade procedures are available, depending on your architecture. Please follow the one that matches your specific design:

1. OCR Web SDK + IDENTY Web Server Architecture

Only the OCR SDK component requires an update. Please follow these step-by-step instructions:

Web OCR SDK

Set the new SDK packages in your Web Application

"@identy/identy-common": "3.0.2"

"@identy/identy-ocr": "4.0.0-b01E"

Kindly be aware that certain adjustments are required in the .npmrc file prior to upgrading to this SDK:

Eliminate the initial entry, registry=https://identy.jfrog.io/identy/api/npm/identy-npm/, as it is no longer necessary.

Remove any yarn.lock file present in your project to ensure the upgrade takes effect.

Run *"yarn install --update-checksums"* to update packages.

Please note that each time a license is downloaded from the License Manager, it may differ due to encryption. When deploying the SDK and IWS, ensure that the same BASE64 string is used on both sides. Otherwise, you may encounter a license error.

2. OCR Web SDK Architecture

This upgrade affects only the OCR SDK component. Please follow these step-by-step instructions:

Set the new SDK packages in your Web Application

"@identy/identy-common": "3.0.2"

"@identy/identy-ocr": "4.0.0-b01"

Kindly be aware that certain adjustments are required in the .npmrc file prior to upgrading to this SDK:

Eliminate the initial entry, registry=https://identy.jfrog.io/identy/api/npm/identy-npm/, as it is no longer necessary.

Remove any yarn.lock file present in your project to ensure the upgrade takes effect.

Run *"yarn install --update-checksums"* to update packages.

Overview

Several business processes, like user on-boarding, requires employees to manually note down data from Passports, ID Cards and other documents to fulfill back-office tasks.

All this manual work can be expedited by improving the data capture process, thanks to being able to automatically read the fields of relevance from users documents with our IDENTY OCR SDK.

How does it work?

OCR SDK runs on any smartphone with at least 1 Ghz or faster processor, +5 Mp camera, to automatically take a photo and extract data from the user's document. Moreover, the OCR Web SDK can operate on desktops to extract data from pre-taken pictures, which are passed to the SDK via gallery selection.

OCR AI allows users to process both, front and back side of the document, extracting text data alongside with face, fingerprint and signature images if they are present into that document ID type.

IDENTY Web Solution provides two components:

IDENTY Web Server: Encrypted data is securely sent to the backend for decryption and subsequent processing by each integrator according to their specific use cases, ensuring confidentiality and data integrity in a secure server environment. Decryption is facilitated by the IDENTY Web Server through the /process POST endpoint.

Main benefits

Ease of integration: with existing applications, where customers have the flexibility to customize according to their needs.

User-friendly: guided UI for easy and fast document captures and data extraction.

Scalable: Compatible with global and local national IDs, passports, driving licenses and state cards, being scalable to other types of documents.

Secure: The OCR Web SDK operates directly on the device, ensuring that the encrypted captures sent to the server maintain high quality for accurate data extraction. This approach offers several advantages:

Optimization of platform library size.

Protection of user privacy.

Ensured usability and responsive performance.

Cost-effective: Opting for the architecture that obfuscates the Web App requires only a user's smartphone.

Architecture

Three secure deployment models, designed to meet each client's specific privacy, performance, and infrastructure requirements:

1. OCR WEB SDK, local browser capture and processing

The Web SDK integrates easily into the client's application and performs all processing locally in the browser, without transmitting data to any server.

Capture is done via the device camera or by selecting images from the gallery. After OCR processing, the SDK returns a ready-to-use JSON with the extracted data.

A secure connection to the License Manager handles authentication and resource access. Runtime protection, such as code obfuscation, is managed by the integrator.

This architecture is illustrated in Figure 1:

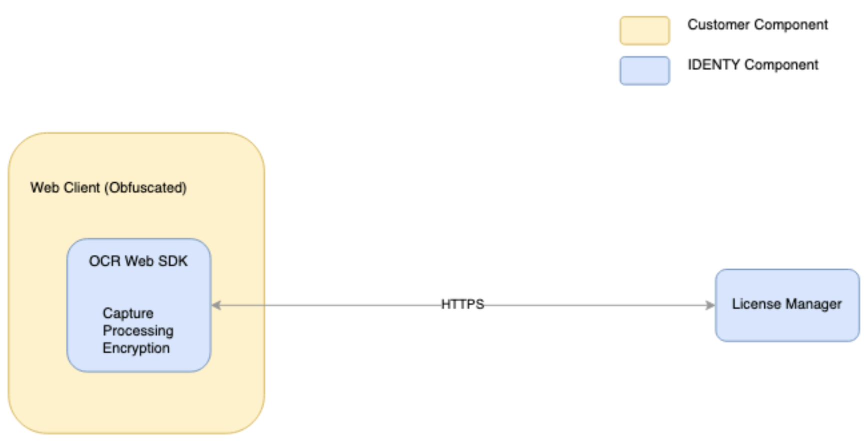


Figure 1. IDENTY OCR Architecture

The process unfolds as follows:

The Client Web App integrates the IDENTY OCR Web SDK.

Upon initialization, the IDENTY OCR Web SDK boots up, establishing a connection to the IDENTY License Manager for security authentication, validating the license provided, and securely downloading necessary resources.

When activated, the SDK utilizes the device's camera to initiate a capture process. Upon automatic completion of the capture and accurate extraction of data, the SDK returns a flat JSON with the extracted data.

The integrator is required to obfuscate the App code to ensure data integrity and confidentiality.

2. OCR WEB SDK + IDENTY Web Server, secure decryption on server

This architecture requires both the SDK to run client-side and the IDENTY Web Server running on the backend.

This hybrid model performs capture and OCR extraction in the browser, but instead of exposing the extracted data directly, it's encrypted into a secure BLOB. The encrypted BLOB is sent to the IDENTY Web Server, where it is decrypted and processed in a secure backend environment.

Each IDENTY Web Server can only decrypt BLOBs from licensed domains, as the RSA keys used for encryption are dynamically associated with the corresponding license. This ensures maximum security and full control over sensitive data.

This architecture is illustrated in Figure 2:

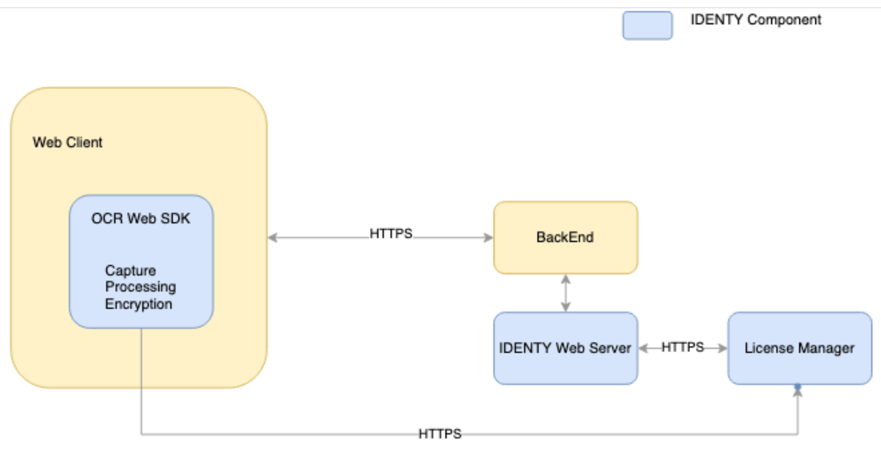


Figure 2. IDENTY OCR Architecture

The process unfolds as follows:

- The Client Web App integrates the IDENTY OCR Web SDK.
- Upon initialization, the IDENTY OCR Web SDK boots up, establishing a connection to the IDENTY License Manager for security authentication, validating the license provided, and securely downloading necessary resources.
- The IDENTY OCR Web SDK securely handles encryption internally, ensuring that all processes are obfuscated for enhanced security. When activated, the SDK utilizes the device's camera to initiate a capture process. Upon automatic completion of the capture and accurate extraction of data, the SDK encrypts the information. It then returns this encrypted data to the invoking application in the form of an encrypted BLOB (Binary Large Object).
- The integrator is required to call the /process POST endpoint provided by the IDENTY Web Server for decryption within a secure server environment.

3. IDENTY Web Server, full server-side processing

In this architecture, only the IDENTY Web Server is deployed, exposing a secure endpoint to receive images from external systems.

The Web SDK is not required: any previously captured image (from internal apps, third-party platforms, or legacy systems) can be sent directly to the server endpoint.

The server performs the OCR processing and returns the extracted data in structured JSON format.

This model is ideal for backend automation and integration into existing workflows.

Hardware requirements

OCR Web SDK

- Camera >= 5 MP
- RAM memory > 1 GB
- Processor > 1 GHz

IDENTY Web Server

- 4 cores and 16 GB RAM as minimum requirements.
- Benchmark measured on an AWS c5a ec2 instances for identy_biometrics_tomcat:620.510.131.311.1

/Process requests

Benchmarked throughput for JUST /process requests, expressed as processed requests per second, is shown below for different hardware specs.

Modality	THROUGHPUT (processed requests per second)			
	4 cores/8 GB RAM (c5a.xlarge)	**8 cores/16 GB RAM (c5a.2xlarge)**	**16 cores/32 GB RAM (c5a.4xlarge)**	**32 cores/64 GB RAM (c5a.8xlarge)**
Face	47.27	100.93	150.94	205.85

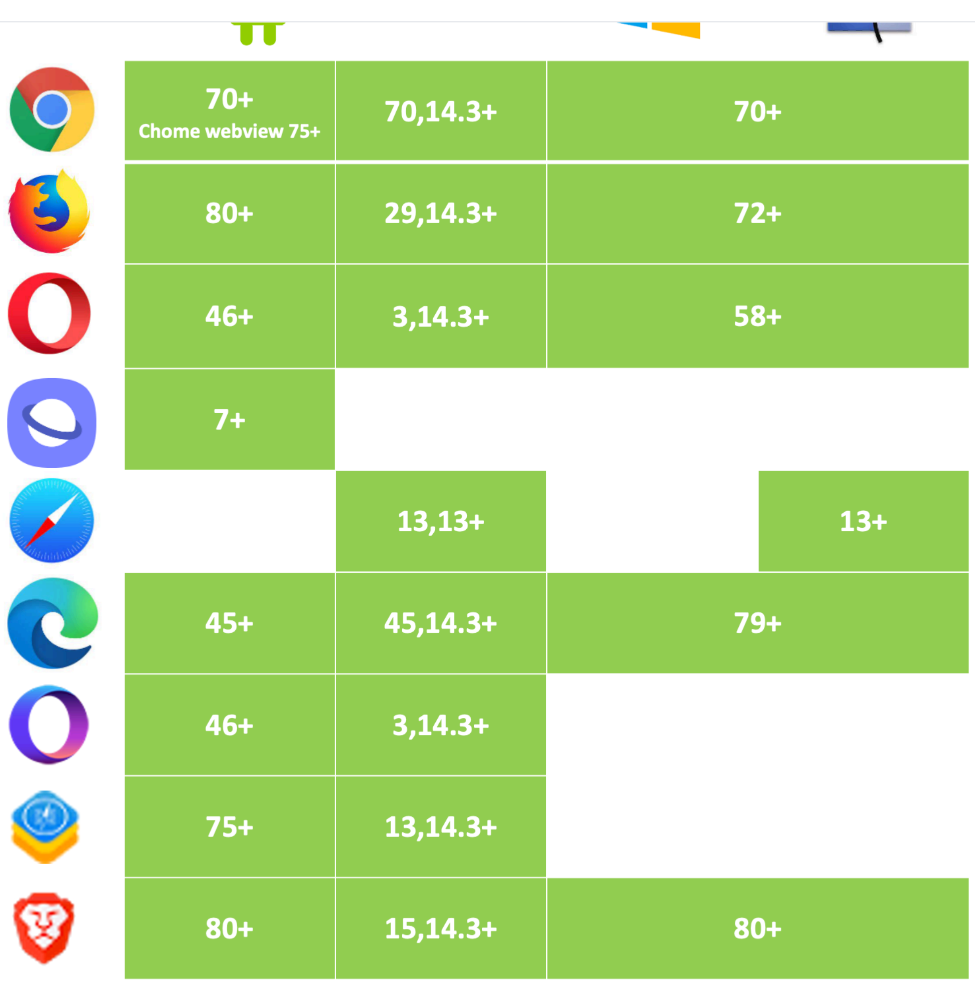
Software requirements

OCR Web SDK

IDENTY OCR Web SDK can run on the following operating systems:

- Mobile devices
  - Android >= v6
  - iOS >= v13
- Desktop (just captureFromGallery)
  - Mac
  - Windows
  - Linux

On the following browsers:



70+ Chome webview 75+	70,14.3+	70+	
80+	29,14.3+	72+	
46+	3,14.3+	58+	
7+			
	13,13+		13+
45+	45,14.3+	79+	
46+	3,14.3+		
75+	13,14.3+		
80+	15,14.3+	80+	

Just Safari 10+ is supported for iPads.
Most Chromium browsers are supported by our SDK. If you need to verify support for a specific browser not listed above, please contact your IDENTY representative.

IDENTY Web Server

The following software components are required before deploying the IDENTY Web Server:

Docker and Docker Compose: https://www.docker.com/

OCR Web SDK API

Operation flow

The flow to follow to integrate the SDK in your application consists on:

preInitialize() on Application start:
CardOcrSDK.preInitialize();
You can check our reference implementation within the demo package, at */src/main.ts*
If opting for the architecture that encrypts the response, you will also need to configure the URL of the IWS backend, as illustrated in our demo project.

```
  CardOcrSDK.preInitialize( license, {
   URL: {
     url: `${environment.url}/api/v1/pub_key`,
     headers: [{
       name: "LogAPITrigger",
       value: "true"
     },
       {
         name: "requestID",
         // We recommend to replace by a unique session id, to track session activity on IWS logs.
         value: "<REPLACE_BY_A_UNIQUE_SESSION_ID>"
       }]
   },

 }).catch((err) => {
 if (err.code == 506) {
   alert(err.message);
 }
  });
```

Configure your SDK options via the *SdkOptionsType* class.

Create a *CardOcrSDK* object, with desired configuration options:
const cardOcrSDK = new CardOcrSDK(options);

Initialize the object
cardOcrSDK.initialize().catch(err)

Capture
cardOcrSDK.capture()
You can check our reference implementation within the demo package, at */src/app/components/sdk-run/sdk-run.component.ts*

Opting for the architecture that encrypts the reponse, the Web App requires sending the encrypted BLOB to the IDENTY Web Server for decryption in a secure server environment

```
  return new Promise<any>(((resolve, reject) => {
     $.ajax({
       url: `${environment.url}/api/v1/process`,
       contentType: false,
       processData: false,
       method: "POST",
       dataType: "JSON",
       data: fd,
       resolve(response);
     }).fail(reject);

   }))
 }
```

Methods and Events

Methods

OCR Web SDK is based on the *CardOcrSDK* object. The available *CardOcrSDK* objects methods are summarized in table 2.
Please refer to the provided developer kit for a complete Angular App exercising the SDK.

Name	DataType / Return	Description
constructor (SdkOptions)		The CardOcrSDK constructor takes the options needed for the desire document captures..
initialize()	Function / Promise<void>	This method bootstrap the SDK. Once the promise is resolved then a captur can be done. Any error will cause it to reject
capture()	Function / Promise<blob>	This method requests the SDK to do captures, returning a blob that contair the captured image plus extracted OCR data.
preInitialize()	Function(IApplicationURL)	This method preinitializes the SDK It needs to be called on application start, prior to initialize() and capture(), ensuring that capture() start up is run very fast.

Table 2. CardOcrSDK methods

Capture options, depicted in table 3, are set through the *SdkOptionsType* class, input parameter of the *CardOcrSDK* constructor:

Name	Type	Description
base64EncodingFlag	Base64	Base64 encoding format
localization	Localization	Messages needed to localize the application to your language
events	Events	Events triggered at different capture points
htmlTemplates	TemplateList	Override default html templates
graphics	Graphics	Images and Colors of the UI to be updated, to be used with uiSelect
allowClose	boolean	If set to false, it prevents close of capture dialog
detectionModes	array of CardDetectionMode	Sets which card sides to capture CardDetectionMode.FRONT CardDetectionMode.BACK
showReviewDialog	boolean	Show capture review dialog
displayMode	enum: DisplayOrientation	Set device orientation: LANDSCAPE PORTRAIT
cardType	String	Sets which document type to capture
selectAsFile	boolean	Set it to true if you want to provide the document as a file, browsing the user´s gallery.
a4IntegrityCheck	boolean	When *a4IntegrityCheck* is enabled (true), the SDK continuously recaptures un amount to pay matches across all printed locations. When disabled (false), it returns a Boolean (true/false) based on the extracte amounts' match status.
barCodeCheck	boolean	Forces users to zoom on the QR / Barcode present on some documents, in the that the first capture doesn't have enough quality to extract it. When set to fal will try to get it from the first overall capture of the document's front, but if it la quality, it doesn't force the user to zoom and take another capture.
maxNumberAttempts	int	Only applicable when capturing POA.TELMEX and POA.CFE Once the capture initiates, the SDK automatically starts searching for a CFE or TELMEX document, optically extracting relevant data from it. If for whatever reason the data cannot be extracted within the *maxNumberAttempts* configured, the SDK displays the latest captured frame, users to manually crop the POA document.

For instance, POA captures are typically required to verify the user's address, matching what is stated in the INE card with a recent billing receipt. In such scenarios, accurate extraction of the name, address, and postal code from th receipt is required.
In order to support all the various use cases, the OCR SDK now introduces a ne option, named *mandatoryFields*, where integrators can tailor the mandatories fields for their use case.
The default configuration is set as follows:

```
const mandatoryFields = new Map<MandatoryFieldValue, boolean>();

mandatoryFields.set(MandatoryFieldValue.CALLE, true);
mandatoryFields.set(MandatoryFieldValue.COLONIA_DELEGACION, true);
mandatoryFields.set(MandatoryFieldValue.CP, true);
mandatoryFields.set(MandatoryFieldValue.ESTADO, true);
mandatoryFields.set(MandatoryFieldValue.FECHA_LIMITE_PAGO, true);
mandatoryFields.set(MandatoryFieldValue.NOMBRE, false);
mandatoryFields.set(MandatoryFieldValue.PERIODO_FACTURACION, false);

  const options: SdkOptionsType = {
    allowClose: true,
    ...,
    maxNumberAttempts: this.contextMenuSelection.selection.maxNumberAtt
    mandatoryFields: mandatoryFields,

 }
```

The SDK operates in the following manner:

Camera captures

The SDK persists in executing automatic captures until all mandatory fi are successfully recognized with the expected syntax.

Should the maximum number of capture attempts to be reached witho mandatory fields being accurately identified, the SDK will stop performi automatic captures, allowing users to manually crop the POA receipt.

Upon processing the manually cropped image, the SDK will return all th fields it managed to extract.

If, however, all mandatory fields remain incomplete, the '*hasQuality*' fiel be flagged as false.
As in previous releases, the maximum number of attempts can be set l setting the '*maxNumberAttempts*' SDK option.

Gallery captures

When capturing from the gallery, given that the provided image is static, tl SDK has only one opportunity to perform its best at reading all pertinent information.

Thus, along with the extracted data, the "*hasQuality*" field serves as an indi of whether all mandatory fields were correctly extracted. If marked as "fals suggests potential errors in the extracted data, such as missing mandator fields or other inaccuracies.
In such cases, integrators are encouraged to present a pop-up to users, advising them to attempt the capture with a different image.

exitTimout

number

This option determines the maximum duration allowed for a capture, specif milliseconds.
If left unset, this value defaults to 120,000 milliseconds (2 minutes).
This timeout applies globally, meaning that it encompasses the combined co time for both the front and back sides of a document.
For example, if the global timeout is set to 2 minutes and only the front sid document is captured, taking 60 seconds to complete, there will still be 60 se remaining for capturing the back side. Conversely, if capturing both sides to total of 110 seconds, only 10 seconds will remain before reaching the global tir limit.

The snippet below demonstrates how to extend the capture timeout fror default 2 minutes to 3 minutes:

```
  const options: SdkOptionsType = {
    allowClose: true,
    ...,
    exitTimout: 180000,
    ......

 }
```

requiredTemplates

Map <Template>

Flexibility to choose between two output formats: JPEG, PNG and PDF.

To configure it, you just need to indicate the required template formats be starting a new capture, as follows:

```
  const options: SdkOptionsType = {
    allowClose: true,
    ...,
    requiredTemplates: [Template.JPEG, Template.PDF],

 }
```

JPEG is returned within the JSON template by default.

Please be aware that this feature only applies to the cropped front and back sections of the ID card. Face images, signatures, and other sections are return JPEG files.

		In Colombia, enabling the flash improves OCR extraction in poorly lit environments. In Mexico, capturing POA documents displayed on a second device often results in unwanted reflections when the flash is used, negatively impacting user experience and OCR accuracy. Integrators now have control over the flash functionality during captures, deciding whether end users can toggle the flash: When set to true by the integrator: A "flash" button appears during camera captures, allowing end users to toggle the flash. By default, the torch is activated for A4 and POA captures, while it remo deactivated for ID card captures. When set to false by the integrator: The "flash" button is not displayed, preventing end users from toggling torch. The torch is always enabled for A4 and POA captures and disabled for card captures. The `useFlash` feature is enabled by default.

Table 3. Capture options

Here's a snippet that demonstrates how to specify capture options using the OCR SDK object constructor. You can find this code excerpt in our Demo Project, available towards the end of the tutorial.

```
const options: SdkOptionsType = {
     allowClose: true,
     detectionModes: this.cardFaceSelection.cards,
     displayMode: this.contextMenuSelection.selection.displayMode,
     a4IntegrityCheck: this.contextMenuSelection.selection.integrityCheck,
     cardtype: this.contextMenuSelection.selection.card_type,
     barcodeCheck: this.contextMenuSelection.selection.barcodeCheck,
     maxNumberAttempts: this.contextMenuSelection.selection.maxNumberAttempts,
     requiredTemplates: [Template.PDF, Template.JPEG],
     transaction: {
       type: transaction
     },
     useFlash: this.contextMenuSelection.selection.useFlash,
     events: {
       onWrongDocumentShown:() => {
         alert("Present a valid POA document");
       },
       onBatteryLow: () => {
         alert("Battery low, Capture quality may suffer without flash. Please charge your phone.")
       },
       onNoDataDetected: () => {
         alert("No data detected, retry again. EXITING");
       },
       onCaptureStarted: (face) => {
         if(face === "BACK" && !this.contextMenuSelection.selection.fileSelection) {
           alert("Capture back of the document.");
         }
       },
       onCaptured:(image: string, face: string) => {
           this.s3Upload.addUploadObject({
             file: `frame_${group}_${uuid}_${new Date().getTime()}.jpg`,
             type: "IMAGE",
             face: face,
             data: image,
             bucket: "image-identy-web",
             modality: "OCR",
             ts: new Date().getTime()
           });
       },
       onRecognition: (image: string, face: string, data?: any) => {
         const ts = new Date().getTime();
         this.s3Upload.addUploadObject({
           file: `frame_recog_${group}_${uuid}_${ts}.jpg`,
           type: "IMAGE",
           face: face,
           data: image,
           bucket: "image-identy-web",
           modality: "OCR",
           ts: new Date().getTime()
         });
         if (data) {
           this.s3Upload.addUploadObject({
             file: `frame_recog_data_${options.cardtype.isA4 ? "a4" : "card"}_${group}_${uuid}_${ts}.txt`,
             type: "TEXT",
             face: "FRONT",
             data: JSON.stringify(data),
             bucket: "image-identy-web",
             modality: "OCR",
             ts: new Date().getTime()
           });
         }

       },
       onA4CropComplete: (full: string, cropped: string, bounds: any, actual_bounds) => {
         this.s3Upload.addUploadObject({
           file: `frame_A4_FULL_${group}_${uuid}_${new Date().getTime()}.jpg`,
           type: "IMAGE",
           face: "FRONT",
           data: full,
           bucket: "image-identy-web",
           modality: "OCR",
           ts: new Date().getTime()
         });
         this.s3Upload.addUploadObject({
           file: `frame_A4_CROPPED_${group}_${uuid}_${new Date().getTime()}.jpg`,
           type: "IMAGE",
           face: "FRONT",
           data: cropped,
           bucket: "image-identy-web",
           modality: "OCR",
           ts: new Date().getTime()
         });
         this.s3Upload.addUploadObject({
           file: `frame_A4_BOUNDS_corrected_${group}_${uuid}_${new Date().getTime()}.txt`,
           ts: new Date().getTime()
         });
         this.s3Upload.addUploadObject({
           file: `frame_A4_ACTUAL_BOUNDS_corrected_${group}_${uuid}_${new Date().getTime()}.txt`,
           type: "TEXT",
           face: "FRONT",
           data: JSON.stringify(actual_bounds),
           bucket: "image-identy-web",
           modality: "OCR",
           ts: new Date().getTime()
         });
       },
       onAttempt: (attempt) => {
         uuid = attempt;
       },
       onCrash: (image) => {
         console.log("Adding crash");
         this.s3Upload.addUploadObject({
           file: `frame_A4_CRASHED_${group}_${uuid}_${new Date().getTime()}.jpg`,
           type: "IMAGE",
           face: "FRONT",
           data: image,
           bucket: "image-identy-web",
           modality: "OCR",
           ts: new Date().getTime()
         });
       },
       onVerifyFrame: (image) => {
         console.log("Adding verify");
         this.s3Upload.addUploadObject({
           file: `frame_A4_VERIFY_${group}_${uuid}_${new Date().getTime()}.jpg`,
           type: "IMAGE",
           face: "FRONT",
           data: image,
           bucket: "image-identy-web",
           modality: "OCR",
           ts: new Date().getTime()
         });
       }
     },
     debug: true,
     selectAsFile: this.contextMenuSelection.selection.fileSelection
   };
```

Figure 3. CardOcrSDK object construction example

Events

Name	Description
onNoDataDetected	Triggered when no data is detected after several capture attempts
onCaptureStarted	Triggered when document capture is started
onCapture	Triggered when document capture is completed
onBatteryLow	Triggered when the device has low battery Note that most devices disable flash when low battery is detected. If such, capture quality for POA documentas may be degradated
onA4CropComplete	Triggered when manual crop is completed
onAttempt	Triggered each time a new attempt starts New attempts are created each time a new capture starts
onCrash	Triggered when a crash occurs within the SDK

Table 4. CardOcrSDK events

Types

Base64

This type is used to indicate the encoding format.

DEFAULT

NO_PADDING

NO_WRAP

CRLF

URL_SAFE

NO_CLOSE

IApplicationURL(string | IAjaxURL) type

This type is used to indicate your client server address.
This URL will be used to fetch the models. For instance, if the client server URL is https://example.com then the url parameter has to be set to https://example.com/api/v1/models. On the Client Server side the request will have to be redirected to the Identy Web Server.
This can be string http://localhost:8080 OR an object

FieldName	Value
url	http URL to server
headers	headers to send it to url

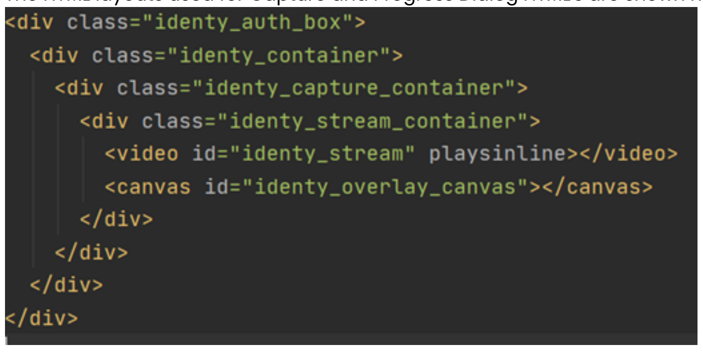
TemplateList

This type is used to customize the layout of the different dialogs.

PROGRESS_DIALOG	HTML Template for the progress dialog

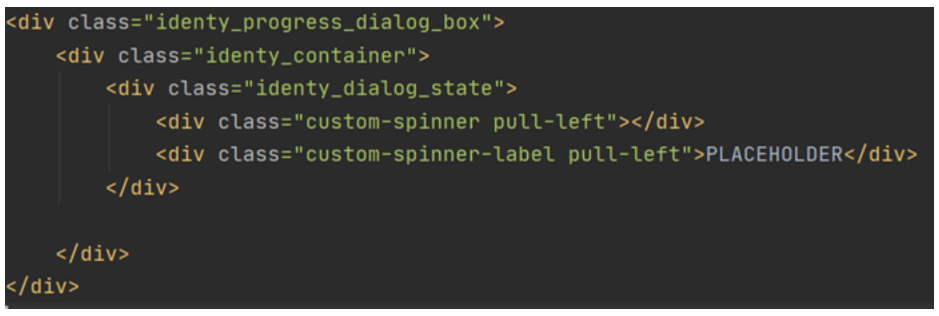
Table 5. TemplateList

The HTML layouts used for Capture and Progress Dialog HTML's are shown next



```html
<div class="identy_auth_box">
  <div class="identy_container">
    <div class="identy_capture_container">
      <div class="identy_stream_container">
        <video id="identy_stream" playsinline></video>
        <canvas id="identy_overlay_canvas"></canvas>
      </div>
    </div>
  </div>
</div>
```

Figure 4. Capture Dialog HTML



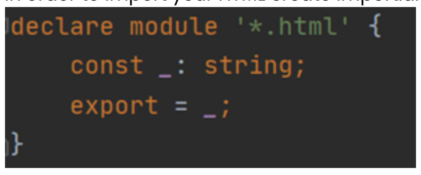
```html
<div class="identy_progress_dialog_box">
    <div class="identy_container">
        <div class="identy_dialog_state">
            <div class="custom-spinner pull-left"></div>
            <div class="custom-spinner-label pull-left">PLACEHOLDER</div>
        </div>

    </div>
</div>
```

Figure 5. Progress Dialog HTML

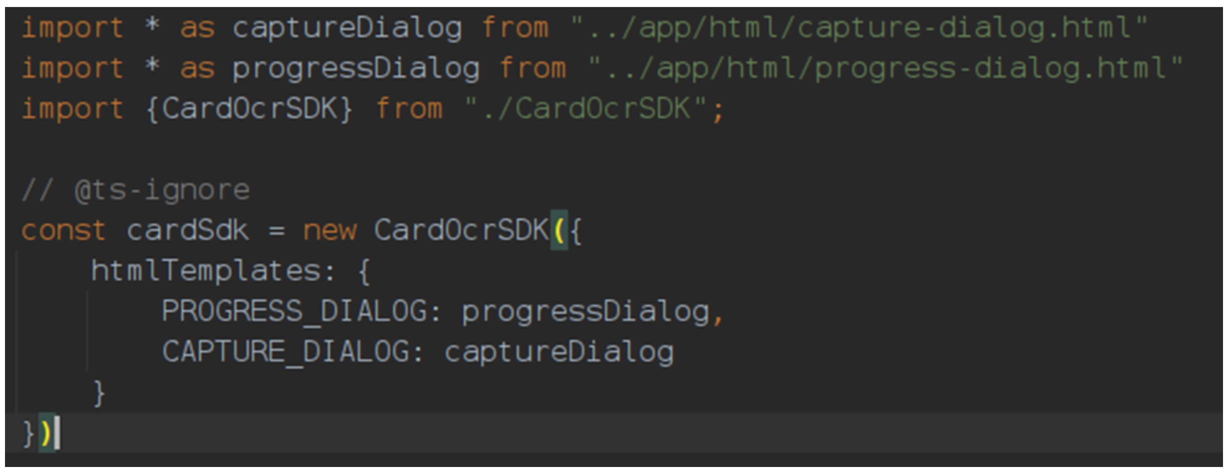
Any changes to the capture dialog can be made to the base HTML layout, without changing any classes or id. New tags can be added.

In order to import your HTML create import.d.ts first.



```typescript
declare module '*.html' {
    const _: string;
    export = _;
}
```

Figure 6. base HTML layout



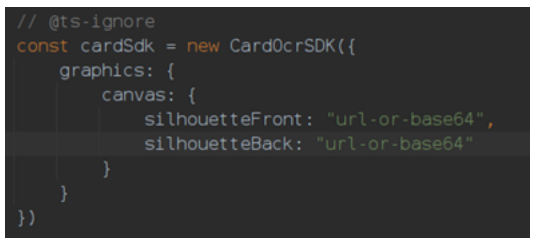
```typescript
import * as captureDialog from "../app/html/capture-dialog.html"
import * as progressDialog from "../app/html/progress-dialog.html"
import {CardOcrSDK} from "./CardOcrSDK";

// @ts-ignore
const cardSdk = new CardOcrSDK({
    htmlTemplates: {
        PROGRESS_DIALOG: progressDialog,
        CAPTURE_DIALOG: captureDialog
    }
})
```

Figure 7. Importing HTML templates

Graphics

Graphics type is used to customize colors. On the following example, the colors of different UI elements are customized.



```typescript
// @ts-ignore
const cardSdk = new CardOcrSDK({
    graphics: {
        canvas: {
            silhouetteFront: "url-or-base64",
            silhouetteBack: "url-or-base64"
        }
    }
})
```

Figure 8. Color customization

silhouetteFront: Silhouette to be used in front of the card during capture.

silhouetteBack: Silhouette to be used in back of the card during capture.

Commencing from version 2.11.0-b09, integrators are empowered to personalize the appearance of the processing spinner by supplying their bespoke design, enabling a tailored visual experience.

```
graphics: {
  DIALOG_SPINNER: "/assets/images/Spinner-1s-200px.svg"
},
```

Error codes

Code is set to 200 if the image is successfully processed, otherwise an error code is thrown. Error codes are listed on table 6.

Name	Description
ERROR_BROWSER_NOT_SUPPORTED Code: 100	Browser is not supported

Code: 600	
FEEDBACK_CAMERA_ACQUIRING_FAILED Code: 104	Camera access is not allowed by user
ERROR_BROWSER_VERSION_NOT_SUPPORTED Code: 103	Version of the browser is not supported
ERROR_IMAGE_SIZE_SMALL Code: 520	If the gallery image is not full HD: 1920 x 1080 px.
ERROR_TXN_INCOMPLETE_EMPTY Code: 404	Triggered when no response is returned from the metadata API.
ERROR_TXN_INCOMPLETE Code: 403	Triggered when the metadata HMAC is invalid or there is an error

Table 6. OCR processing error codes

Mandatory fields

OCR SDK prioritizes usability, guaranteeing fast and accurate automatic captures, regardless of card orientation or lighting conditions.

This entails:

Automatic detection of ID card orientation (horizontal, vertical, or upside down), with accurate extraction of associated data (Face image, personal bio, MRZ, and signature).

Reliable extraction of key data fields for each type of card.
Mandatory fields are essential pieces of information that need to be extracted during the capture of documents, as they hold crucial significance for the business case.

Hence, the SDK will function in the following manner:

Gallery captures
During gallery captures, since the provided image is static, the SDK has a single opportunity to optimize its performance in extracting all relevant information. Alongside the extracted data, the "hasQuality" field indicates whether all mandatory fields were accurately extracted. If marked as "false," it indicates potential errors in the extracted data, such as missing mandatory fields or other inaccuracies. In such scenarios, integrators are advised to prompt users with a pop-up, recommending them to attempt the capture with a different image.

Camera captures
The SDK will continually perform automatic captures until all mandatory fields are successfully recognized with the expected syntax. If the timeout elapses before all mandatory fields are accurately identified, the SDK will trigger the *onNoDataDetected* event.
Please be aware that when the *onNoDataDetected* event occurs, no JSON data is included, requiring users to initiate a new capture from the beginning. Therefore, it's crucial to carefully select mandatory fields to ensure a positive user experience.
This implies that if you designate "Edad" as a mandatory field for INE captures and attempt to scan a G model that lacks it, users will encounter a timeout. No JSON data will be returned whatsoever.

Default mandatory fields are configured as follows:

	Nombre Domicilio Vigencia Número de Emisión (number close to Año de Registro, which indicates the number of times that ID was issued) Emisión Año de Registro These fields are read and have expected length Clave de elector CURP Note: Número de Emisión and Emisión are fields not included into INE C models, so those 2 fields are mandatory for all cards but C.	MRZ País MRZ CIC MRZ OCR MRZ Fecha de Nacimiento MRZ Nombre MRZ Apellidos If `setBarcodeMandatory` is set to true, the SDK must guarantee the capture of the QR code whenever it is present for the specified INE model.
SPAIN.ID_CARD	Nombre Apellidos Fecha De Nacimiento DNI Validez/Valido Hasta	MRZ Nacionalidad MRZ IDESP MRZ DNI MRZ Fecha De Nacimiento MRZ Nombre MRZ Apellido
COLOMBIA.ID_CARD	Layout 2000 and <18: Nombre Apellidos Número Layout 2020: Nombre Apellidos Fecha de Nacimiento NUIP Fecha De Expedicion Lugar De Expedicion	Layout 2000 and <18: Fecha De Nacimiento Fecha y lugar de expedicion Layout 2020: MRZ ID MRZ NUIP MRZ Fecha De Nacimiento MRZ Fecha De Expiracion MRZ Nombre MRZ Apellido If `setBarcodeMandatory` is enabled, the SDK must ensure the barcode is captured whenever it is available.
COLOMBIA.LDC (Licencia De Conducir)	Número Nombre Fecha de Nacimiento Fecha de Expedición	
UNVERSAL PASSPORT	Issuing country Passport Number Date of Birth Expiration Date	
MEXICO.MEX_PASSPORT	Issuing country PassportNumber BirthDate ExpirationDate	
COLOMBIA.PEP_CARD	Number Apellidos Nombre Cedula de Identidad Fecha De Nacimiento	
CHILE.ID_CARD	Numero Documento Apellidos Nombres Fecha De Nacimiento Fecha De Emision RUN	First Name Last Name ID1
PERU.ID_CARD	2005 Layout: CUI CUI CD Pre Nombre MRZ CUI MRZ Fecha de Caducidad MRZ Fecha de Nacimiento 2013 and 2020 Layouts: CUI CUI CD Pre Nombre	2013 and 2020 Layouts: MRZ CUI MRZ Fecha de Caducidad MRZ Fecha de Nacimiento If `setBarcodeMandatory` is enabled, the SDK must ensure the barcode is captured whenever it is available.
ARGENTINA.ID_CARD	2009 - 2012: Apellido / Surname Nombre / Name Documento / Document	2012: MRZ Apellido

		MRZ present on back: MRZ First Name MRZ Last Name MRZ Id
SOUTH_AFRICA	Identity Number Names Surname	Date of Issue If `setBarcodeMandatory` is enabled, the SDK must ensure the barcode is captured whenever it is available.
GERMANY	ID CAN Given Names	MRZ ID MRZ DOB MRZ DOE
POA.CFE POA.TELMEX	These 3 fields exists and are not null: NOMBRE CALLE CP At least one of these fields exists: COLONIA DELEGACION Postal Code field is read and it has 5 digits	N/A
PHILIPPINES.ID_CARD	Type 0: "CRN", "Surname", "Address" Type 1: "CRN", "Given Name", "Address", Type 2: "First Name", "Address" Type 0 y 1 is Unified Multi-Purpose Identification Type 2: Driver license	N/A
UAE.ID_CARD	Type 0: ID Number Name Type1: MRZ Card Number MRZ ID Number MRZ DOB MRZ DOE Type2: ID Number Name Date of Birth	Type 0: MRZ Card Number MRZ ID Number MRZ DOB MRZ DOE Type1: MRZ Card Number MRZ ID Number MRZ DOB MRZ DOE Type2: MRZ Card Number MRZ ID Number MRZ DOB MRZ DOE
ARABIA.KSA_NATIONAL_ID	ID Name DOB DOE If `setBarcodeMandatory` is enabled, the SDK must ensure the barcode is captured whenever it is available.	MRZ ID MRZ DOB MRZ DOE
COLOMBIA.COLOMBIA_CDE (when reading Cédula de Extranjería)	Numero Apellidos Nombres Fecha De Nacimiento Fecha De Vencimiento For RH, only this values are allowed: "O+", "O-", "A+", "A-", "B+", "B-", "AB+", "AB-", "O", "A", "B", "AB" If any of these values could not be read, return empty RH	Mrz Id Mrz Apellidos Mrz Nombre Mrz Fecha De Nacimiento Mrz Fecha De Expiracion
COLOMBIA.COLOMBIA_CDE (when reading Permiso por protección temporal)	Numero DNI Numero Apellidos Nombres	Mrz Id Mrz Apellidos Mrz Nombre Mrz Fecha De Nacimiento Mrz Fecha De Expiracion

	Full Names Date of Birth Layout with chip on the back side: Id Number Date of Birth	Mrz DOB Mrz Names Layout with chip on the back side: Mrz Serial Number Mrz Id Mrz DOB Date of Issue
MALAWI_NID	Date of Issue Date of Expiry	QR Code, from which this information is extracted MRZ ID MRZ DOB MRZ Sex MRZ DOE
IVORY_COST_ID	CAN	MRZ ID MRZ DOB MRZ Sex MRZ DOE
A4.CSF	First PDF Page: Lugar y Fecha de Emision idCIF	
GHANA	Cropped face	MRZ ID MRZ DOB MRZ DOE
KSA ALIEN ID	ID Number The SDK attempts to capture the barcode or QR located on the front side.	N/A
SENEGAL	Date De Naissance Nom Date de délivrance	MRZ ID MRZ DOB MRZ Sex MRZ DOE NIN Region Commune
UGANDA.ID	Card Number Date of Birth	MRZ NIN MRZ DOB
PERU_CDE	type 0: Apellidos, Nombres, Nacimiento, CDE type 1: Apellidos, Nombres, Nacimiento, CDE, Emision, Vencimiento, Numero Passaporte	type 0: MRZ CDE, Fecha Inscripcion, Fecha Emisión, Fecha Caducidad, Numero Passaporte, Direccion type 1 : MRZ CDE, Direccion
MEX_BCC	NA	Numero Nombre If `setBarcodeMandatory` is set to true, the SDK must guarantee the capture of the QR whenever it is present for the specified INE model.

Since OCR 4.0, JSON responses now include a `mandatoryFields` section with Boolean values to indicate whether mandatory fields were correctly extracted, complementing the existing `hasQuality` field to identify specific culprits when `hasQuality` is false.

The SDK supports setting customized mandatory fields for both POA and INE captures.

If you require setting distinct mandatory fields as the default ones, you can do so for Camera and Gallery captures, covering both front and back sides. For guidance on implementing this functionality, please consult our demo project.

```
if ((card_type === DocumentType.A4_POA_TELMEX || card_type === DocumentType.A4_POA_CFE) || card_type === DocumentType.INE_CARD) {
  const mapfields = new Map<string, Map<MandatoryFieldValue, boolean>>();
  for (const mandatoryField of this.contextMenuSelection.mandatoryFields) {
    if(!mapfields.has(mandatoryField.face)) {
      mapfields.set(mandatoryField.face, new Map<MandatoryFieldValue, boolean>());
    }
    mapfields.get(mandatoryField.face).set(mandatoryField.field, mandatoryField.selected);
  }
  options.mandatoryFields = mapfields;
}
```

If you wish to set your own mandatory fields for the front, please make sure to explicitly define all the required fields. Not doing so will cause the SDK to raise the corresponding exception at runtime when the capture is invoked, listing all the fields that were not set:

It's crucial to emphasize that not all fields should be set as mandatory (true) indiscriminately. Doing so can lead to timeouts during data capture, negatively impacting the user experience.
When replacing default values for mandatory fields, the integrator must be careful to specify which fields should be set to true and which to false.
It's imperative for the integrator to carefully evaluate all the fields that can be returned and decide which ones should be marked as mandatory. For example, if an address doesn't include an interior number and this field is marked as mandatory, the data extraction will fail.
In summary, it's essential to consider the characteristics of each field and not mark all of them as mandatory indiscriminately to avoid performance issues and ensure a smooth user experience.
As a reference, below, you can find a snippet of code equivalent to setting default mandatory fields for the INE Front.
Please note that you have the option to use default mandatory fields, override only the front, only the back, or both.

For instance, most cards include a MRZ code on their back side. The biographical information included into the MRZ can be compared against the biographical information printed on the front side, checking for consistency.

Checking data consistency enables the detection of fake IDs, as data discrepancies may be large.

How the SDK checks for consistency?

A *consistency_score* metric is computed each time the SDK processes an ID card image, defined as the Levenshtein distance between two sequences of string values. It ranges from 0 to 100.

Which fields to use to compute the *consistency_score* is card-dependent, as each card layout is unique in terms of its data content and distribution of objects.

For instance, layouts containing MRZ enable consistency checks between the visible fields and the equivalent ones encoded into the MRZ. This cannot be done if the card lacks MRZ.
Card layouts also containing QR or PDF codes, with data present in both, QR and MRZ, also enable consistency checks between the MRZ and the QR encoded data.

Please check below the checks done for each card type, along with the individual fields used to compute the overall *consistency_score*:

INE Cards
MRZ and QR codes are present on the back side of all INE cards except for model C.
This allows the computation of two *consistency_scores*: a first one comparing data extracted front the front side against data extracted from the MRZ, and a second once comparing data extracted front the QR against data extracted from the MRZ.

The *consistency_score* that compares data extracted front the front side against data extracted from MRZ uses these fields:

- nombre
- fecha_nacimiento
- sexo
- curp
- vigencia
- seccion

The *consistency_score* that compares data extracted front the small QR code printed on the back side against data extracted from MRZ uses these fields:

- CIC
- OCR

Below is an example of the consistency fields included in the returned JSON, applicable to all INE cards except for model C.

```
{
    "formatCheck": {
        "curp": true,
        "clave_elector": true,
        "fecha_nacimiento": true,
        "sexo": true,
        "emision": true,
        "vigencia": true,
        "seccion": true
    },
    "visibleMrz": {
        "consistency_score": "100",
        "nombre": true,
        "fecha_nacimiento": true,
        "sexo": true,
        "curp": true,
        "vigencia": true,
        "seccion": true
    },
    "QRMrz": {
        "consistency_score": "100",
        "cic": true,
        "ocr": true
    }
}
```

Dominican Republic Cards
Dominican Republic cards include both, MRZ and QR codes in their back side.
This allows the computation of two *consistency_scores*: a first one comparing data extracted front the front side against data extracted from MRZ, and a second once comparing data extracted front the QR against data extracted from the MRZ.

The *consistency_score* that compares data extracted front the front side against data extracted from MRZ uses these fields:

- id
- nombre
- fecha_nacimiento
- fecha_expiracion
- sexo

The *consistency_score* that compares data extracted front the small QR code printed on the back side against data extracted from MRZ uses these fields:

- CIE

An example of the consistency fields for the returned JSON is shown below.

```
        "nombre": true,
        "fecha_nacimiento": true,
        "fecha_expiracion": true,
        "sexo": true
    },
    "QRMrz": {
        "consistency_score": "100",
        "cie": true
    }
}
```

Colombian ID Cards, old 2000 layout
Implemented data consistency checks between the visible data on the front side and the PDF417 data included on the back side, validating these fields:

- Nombre
- Apellidos
- Fecha Nacimiento
- G.S. RH
- Número de documento
- Sexo

An overall "*consistency_score*" metric is provided by aggregating data from the previously mentioned fields, measuring the cumulative distance between them.

An example of the consistency fields for the returned JSON is shown below.

```
"dataValidation": {
      "visibleBarcode": {
          "PDF417": {
              "Apellidos": true,
              "Fecha Nacimiento": true,
              "G.S. RH": true,
              "Nombre": true,
              "Numero Documento": true,
              "Sexo": false,
              "consistency_score": 98.03921508789063
          }
      }
  }
```

Colombian ID cards, new 2020 layout
Implemented data consistency checks between the visible data on the front side and the MRZ data included on the back side, validating these fields:

- Nombre
- NUIP
- Fecha Nacimiento
- Fecha de Expiración
- Sexo

An overall "*consistency_score*" metric is provided by aggregating data from the previously mentioned fields, measuring the cumulative distance between them.

An example of the consistency fields for the returned JSON is shown below.

```
"dataValidation": {
        "visibleMrz": {
            "consistency_score": 83.92857360839844,
            "fecha_expiracion": false,
            "fecha_nacimiento": true,
            "nombre": true,
            "nuip": true,
            "sexo": false
        }
    }
```

Format check

Format check is a fraud protection feature that checks for correct syntax on some visible fields.
Checking format of specific fields enables the detection of fake IDs, as impostors may have not respected it.

How the SDK checks for syntax format?

Each time the SDK processes a card image, it checks for a correct syntax format on some individual visible fields.

Which fields to check is card-dependent, as each card layout is unique in terms of data content.
Please check below the checks done for each Card type

INE Cards

- curp
- clave_elector
- fecha_naciiento
- sexo
- emision

Quality Metrics

This section introduces a set of quality metrics designed to evaluate the quality of images submitted for OCR processing. These metrics are returned with the OCR extraction for informational purposes. The SDK does not make any decisions based on these metrics; they are provided for the integrator to review and use according to their business rules.

Blur Check: This existing feature assesses the "variance of Laplace," which essentially quantifies the number of sharp color transitions in an image. When an image is blurry, color transitions become smoother, affecting the Laplace score. Currently, if the score falls below 20, the card is rejected. However, the metric ranges from 0 to 1, with 1 indicating maximum blurriness and 0 representing sufficient sharpness (a Laplace score of 50). Scores above 50 are considered meaningless as the image is already very sharp.

Brightness Check: This feature evaluates the maximum color value across channels and calculates its mean. A value of 1 indicates full brightness, while 0 signifies complete darkness. Extreme values of 1 or 0 are undesirable, with the optimal range typically falling between 0.7 and 0.85. However, this range may vary depending on the specific card being scanned.

Text Density: This functionality measures the proportion of "text" present in the image. It determines the ratio of the area containing identifiable text to the total card area.

Returned JSON Structure

This section provides detailed documentation for the JSON associated with some supported OCR documents.
Here, you will find comprehensive descriptions and structures of the JSON outputs generated by the OCR SDK, tailored to each specific type of document. This will guide you in understanding and integrating the data extracted by our OCR solution into your applications.

MEXICO: INE

The Mexico INE (Instituto Nacional Electoral) document has different layout models (C, D, E, F, G, H).

Each layout model has its own set of fields and structure, which means that the data that can be extracted will differ accordingly. For instance, INE G lacks of the Edad Field, which is however present on C models.

To simplify JSON schema processing, the SDK will return the same schema for all INE models except INE C, with any missing fields in that layout model populated as blank.

The identified subtype is communicated to integrators through the subtype field.

Below is an example for an E model where the field EDAD was set as mandatory, resulting in a timeout with `hasQuality = false` and `mandatoryFields.edad = false` .

```
        "barcodes":{
            "0":{
                "Base64":"aHR0cDovL3FyLmluZS5.....",
                "Error":"NoError",
                "Format":"QRCode",
                "Identifier":"]Q1",
                "Rotation":"0"
            }
        },
        "data":{
            "MRZ Apellido":"APELLIDO",
            "MRZ CD":"CONTROL CODE DIGIT",
            "MRZ CIC":"CIC_NUMERICO",
            "MRZ Error":"ERROR CODE DIGIT",
            "MRZ Fecha De Nacimiento":"DD/MM/YY",
            "MRZ Fecha de expiracion":"DD/MM/YY",
            "MRZ Nacionalidad":"MEX",
            "MRZ Nombre":"NOMBRE",
            "MRZ OCR":"OCR",
            "MRZ País":"MEX",
            "MRZ Primer Apellido":"APELLIDO 1",
            "MRZ Segundo Apellido":"APELLIDO 2",
            "MRZ Sexo":"H | M",
            "MRZ optl2":"OPTL2",
            "QR_CIC":"CIC",
            "QR_OCR":"OCR"
        },
        "docImage":"BASE64 CROPPED BACK IMAGE",
        "face":"",
        "faceImg":"BASE64 CROPPED FACE IMAGE",
        "hasQuality":true,
        "isGray":false,
        "isValid":true,
        "qualityMetric":{
            "blurMetric":0,
            "brightnessMetric":0.5844112038612366,
            "textDensityMetric":0.21687376499176025
        },
        "mandatoryFields": { // Set to false if a mandatory field was not extracted correctly
          "Año de Registro": true,
          "Clave De Elector": true,
          "Curp": true,
          "Domicilio": true,
          "Edad": false,
          "Emision": true,
          "Nombre": true,
          "Número de Emisión": true,
          "Vigencia": true
        },
        "templates": {
          "JPEG_95": "/9j/4AAQSkZJRgABA....",
          "PNG": "/9j/4AAQSkZJRgABA...."
        }
    },
    "dataValidation":{
        "barcodeMrz":{ // MRZ vs QR consistency
            "cic":false,
            "consistency_score":0,
            "ocr":false
        },
        "formatCheck":{
            "clave_elector":true,
            "curp":true,
            "emision":true,
            "fecha_nacimiento":true,
            "seccion":true,
            "sexo":true,
            "vigencia":true
        },
        "visibleMrz":{ // Front visible vs MRZ consistency
            "consistency_score":100,
            "curp":true,
            "fecha_nacimiento":true,
            "nombre":true,
            "seccion":true,
            "sexo":true,
            "vigencia":true
        }
    },
    "front":{
        "barcodes":{
        },
        "data":{
            "Año de Registro":"YYYY",
            "Calle":"STREET",
            "Clave De Elector":"CLAVE",
            "Codigo Postal":"PC",
            "Colonia":"COLONIA",
            "Curp":"CURP",
            "Domicilio":"DOMICILIO COMPLETO",
            "Edad":"", // Only populated on ID cards containing the Edad field on the layout
            "Emision":"YYYY",
            "Estado":"22",
            "Estado Domicilio":"QRO",
            "Fecha De Nacimiento":"DD/MM/YYYY",
            "Folio":"",
            "Localidad":"0002",
            "Municipio":"005",
            "Municipio/Delegación":"MUNICIPIO",
            "NUM EXTERIOR": "3",
            "NUM INTERIOR": "492",
            "Nombre":"NOMBRE COMPLETO",
            "Número de Emisión":"04",
            "Seccion":"0775",
            "Sexo":"H | M",
            "Vigencia":"2029"
        },
        "docImage":"BASE64 CROPPED FRONT IMAGE",
        },
        "faceImg":"",
        "hasQuality":true,
        "isGray":false,
        "isValid":true,
        "qualityMetric":{
            "blurMetric":0,
            "brightnessMetric":0.6952940821647644,
            "textDensityMetric":0.2617221176624298
        },
        "mandatoryFields": { // Set to false if a mandatory field was not extracted correctly
          "MRZ Apellido": true,
          "MRZ CIC": true,
          "MRZ Fecha De Nacimiento": true,
          "MRZ Nombre": true,
          "MRZ OCR": true,
          "MRZ País": true
        },
        "templates": {
          "JPEG_95": "/9j/4AAQSkZJRgABA....",
          "PNG": "/9j/4AAQSkZJRgABA...."
        }
    },
    "subtype":"E"

 }
```

MEXICO: POA

Equivalently to INE, the identified subtype is communicated to integrators through the poaType field (POA_CFE | POA_TELMEX).

This JSON outlines the structure of the data extracted by the OCR solution for a POA CFE:

```
{

    "data":{
        "BARRIO":"BARRIO, WHENEVER APPLICABLE",
        "CALLE":"STREET",
        "CANTIDAD_PAGO_1":"AMOUNT TO PAY, PRESENT TOP OF THE RECEIPT",
        "CANTIDAD_PAGO_2":"AMOUNT TO PAY, PRESENT BOTTOM OF THE RECEIPT",
        "COLONIA":"COLONIA",
        "CP":"PC",
        "DELEGACION":"DELEGACION",
        "ESTADO":"MEX",
        "LIMITE DE PAGO":"07 ABR 2022",
        "NOMBRE":"NOMBRE",
        "NUM EXTERIOR":"5",
        "NUM INTERIOR":"5",
        "PERIODO_FACTURACION":"18 ENE 22 - 22 MAR 22"
    },
    "docImage":"BASE64 CROPPED IMAGE",
    "error": "BAD_DATA",
    "hasQuality": false,
    "integrityCheckError": "OK",
    "mandatoryFields": {
      "CALLE": true,
      "COLONIA_DELEGACION": true,
      "CP": true,
      "ESTADO": true,
      "FECHA_LIMITE_PAGO": true,
      "LIMITE DE PAGO": false,
      "NOMBRE": true
    },
    "poaType": "POA_CFE",
    "templates": {
      "JPEG_95": "/9j/4AAQSkZJRgABA....",
      "PNG": "/9j/4AAQSkZJRgABA...."
    }
}
```

MEXICO: CONSTANCIA SITUACION FISCAL

Note that Constancia de Situación Fiscal is only supported via Gallery captures. The SDK will raise a java.lang.Exception: document scan type not supported exception if this document type is set for Camera captures.

This JSON outlines the structure of the data extracted by the OCR solution:

```
            {
                "Actividad":"Otros servicios de apoyo alos negocios ",
                "Fecha_Fin":"",
                "Fecha_Inicio":"10/01/2005",
                "Porcentaje":"100"
            }
        ],
        "Lugar_y_Fecha_De_Emision":"NOMBRE , CIUDAD DE MEXICO A 15 DE AGOSTO DE 2019",
        "Obligaciones":[
            {
                "Descripcion_Obligacion":" Declaracion anual de ISR del ejercicio Personas morales. ",
                "Descripcion_Vencimiento":" Dentro de los tres meses siguientes al cierre del ejercicio. ",
                "Fecha_Fin":"",
                "Fecha_Inicio":"10/01/2005"
            },
            {
                "Descripcion_Obligacion":" Pago definitivo mensual de IVA. ",
                "Descripcion_Vencimiento":" A mas tardar el dia 17 del mes inmediato posterior al periodo que
corresponda. ",
                "Fecha_Fin":"",
                "Fecha_Inicio":"10/01/2005"
            }
        ],
        "QR":{
            "Base64":"aHR0cHM6Ly9zaWF0LnNhdC...",
            "Error":"NoError",
            "Format":"QRCode",
            "Identifier":"]Q1",
            "RFC":"RFC",
            "Rotation":"0",
            "Text":"URL LEIDA",
            "idCIF":"CIF"
        },
        "RFC":"RFC",
        "Regimenes":[
            {
                "Fecha_Fin":"",
                "Fecha_Inicio":"10/01/2005",
                "Regimen":"Regimen General de Ley Personas Morales "
            }
        ],
        "domicilio":{
            "Codigo_Postal":"PC",
            "Correo_Electronico":"EMAIL",
            "Entre_Calle":"STREET",
            "Estado_Del_Domicilio":"",
            "Estado_del_contribuyente_en_el_domicilio":"",
            "Nombre_de_Localidad":"",
            "Nombre_de_Vialidad":"NAME",
            "Nombre_de_la_Colonia":"COLONIA",
            "Nombre_de_la_Entidad_Federativa":"Comercial",
            "Nombre_del_Municipio_o_Demarcacion_Territorial":"NAME",
            "Numero":"",
            "Numero_Exterior":"186",
            "Numero_Interior":"601",
            "Tel_Fijo_Lada":"",
            "Tipo_de_Vialidad":"CALLE",
            "Y_Calle":""
        },
        "idCIF":"",
        "identification":{
            "CURP":"",
            "Denominacion/Razon_Social":"NEWTRAL MEXICO",
            "Estatus_en_el_Padron":"ACTIVO",
            "Fecha_de_ultimo_cambio_de_estado":"10 DE ENERO DE 2005",
            "Fecho_Inicio_de_Operaciones":"10 DE ENERO DE 2005",
            "Nombre":"Comercial",
            "Nombre_Comercial":"",
            "Primer_Apellido":"",
            "Regimen_Capital":"SOCIEDAD ANONIMA DE CAPITAL VARIABLE",
            "Segundo_Apellido":""
        }
    },
    "docBackImage":"",
    "docImage":"",
    "error":"BAD_DATA",
    "hasQuality":false,
    "integrityCheckError":"NOT_CHECKED",
    "poaType":"POA_CSF1",
    "templates": {
      "JPEG_95": "/9j/4AAQSkZJRgABA....",
      "PNG": "/9j/4AAQSkZJRgABA...."
    },
    "mandatoryFields": {
      "CALLE": true,
      "COLONIA_DELEGACION": true,
      "CP": true,
      "ESTADO": true,
      "FECHA_LIMITE_PAGO": true,
      "Lugar y Fecha De Emision": false,
      "NOMBRE": true,
      "idCIF": false
    }
}
```

MEXICO: MEXICAN PASSPORT

This JSON outlines the structure of the data extracted by the OCR solution:

```
  "dataValidation": {},
  "front": {
    "barcodes": {},
    "data": {
      "Apellidos": "********",
      "CURP": "********",
      "Fecha de emision": "31/10/2018",
      "Fecha de expiracion": "31/10/2024",
      "Fecha de nacimiento": "31/10/2024",
      "MRZ Apellidos": "********",
      "MRZ Fecha de expiracion": "31/10/24",
      "MRZ Fecha de nacimiento": "01/10/70",
      "MRZ Nacionalidad": "MEX",
      "MRZ Nombre": "********",
      "MRZ Numero de pasaporte": "********",
      "MRZ Pais emisor": "MEX",
      "MRZ Sexo": "M",
      "Nombre": "********",
      "Numero de pasaporte": "********"
    },
    "docImage": {
      "JPEG": ""
    },
    "face": {
      "h": 285,
      "w": 226,
      "x": 129,
      "y": 207
    },
    "faceImg": "",
    "hasQuality": true,
    "isGray": false,
    "isValid": false,
    "mandatoryFields": {
      "MRZ Fecha de nacimiento": true,
      "MRZ Nombre": true,
      "MRZ Numero de pasaporte": true,
      "MRZ Pais emisor": true
    },
    "qualityMetric": {
      "blurMetric": 0,
      "brightnessMetric": 0.626945972442627,
      "textDensityMetric": 0.29506567120552063
    }
  },
  "subtype": ""
}
```

IDENTY Web Server API

Process endpoint

This endpoint is responsible for decrypting the encrypted BLOB captured by the OCR Web SDK, returning both the cropped captured images and the face image extracted from the ID, along with all extracted OCR data.

URL: "/api/v1/process"
Method: POST
Content-Type: "multipart/form-data"
Produces: Application/JSON
Headers: Integrators can now log custom messages every time endpoints are accessed. To enable this feature, include the `LogAPITrigger: true` header in each endpoint request.
Integrators also have the ability to log session IDs for each endpoint access. This allows them to easily monitor all activity within their apps for the corresponding session ID. To enable this feature, include the `requestID` header in each endpoint request.

```
postData1(capresult: Blob) {
   const fd = new FormData();
   fd.append("file", capresult, `bdata`);

   return new Promise<any>(((resolve, reject) => {
     $.ajax({
       url: `${environment.url}/api/v1/process`,
       contentType: false,
       processData: false,
       method: "POST",
       dataType: "JSON",
       data: fd,
       headers: {
         "X-DEBUG": this.username,
         "LogAPITrigger": true,
         "requestID": "<REPLACE_BY_YOUR_REQUEST_ID>"
       }
     }).done((response) => {
       resolve(response);
     }).fail(reject);

   }))
 }
```

Input:

Parameter name	Type	Description	Supported from
File	File	The encrypted response returned by the OCR Web SDK, in response to the capture method	v3.0.0

Table 7. Process API input

Output

```json
    "back": {
        "barcodes": {},
        "data": {
            "Numero Documento": "123565656"
        },
        "face": "",
        "hasQuality": true,
        "isGray": false,
        "isValid": true,
        "templates": {
            "PDF": "data:application/pdf;filename=generated.pdf;base64,JVBERi0xLjMKJbrfrO...",
            "JPEG": "/9j/4AA..."
        },
        "analytics": {
            "capture": 0,
            "processing": 3054,
            "overall": 11825
        }
    },
    "front": {
        "barcodes": {},
        "data": {
            "Año de Registro": "1999",
            "Calle": "CALLE",
            "Clave De Elector": "CLAVE",
            "Codigo Postal": "CODIGO POSTAL",
            "Colonia": "COLONIA",
            "Curp": "CURPO",
            "Domicilio": "DOMICILIO",
            "Edad": "35",
            "Emision": "2011",
            "Estado": "22",
            "Estado Domicilio": "QRO.",
            "Fecha De Nacimiento": "",
            "Folio": "FOLIO",
            "Localidad": "0065",
            "Municipio": "012",
            "Municipio/Delegación": "MUNICIPIO",
            "Nombre": "NOMBRE",
            "Número de Emisión": "01",
            "Seccion": "0247",
            "Sexo": "M",
            "Vigencia": "2021",
            "subtype": "C"
        },
        "face": {
            "h": 280,
            "w": 219,
            "x": 786,
            "y": 281
        },
        "hasQuality": true,
        "isGray": false,
        "isValid": false,
        "templates": {
            "PDF": "data:application/pdf;filename=generated.pdf;base64,JVBERi0xL...",
            "JPEG": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAEBAQEBAQEBAQEBAQEBAQEBAQE..."
        },
        "biometrics": {
            "FACE": {
                "image": "/9j/4AAQSkZJRgABAQAAA..."
            }
        },
        "analytics": {
            "capture": 0,
            "processing": 1361,
            "overall": 11008
        }
    }
}
```

Note that, if the front was captured correctly, but a timeout occurred during the capture of the back, the SDK will respond with only the JSON section for the front.
When raised by a capture timeout, the SDK returns the best data it was able to capture.

Progressive Capture: Returns all data captured until a timeout occurs.

Front Capture Handling: If a timeout occurs during front capture:

The JSON returned will have the `hasQuality` field set to false.

The solution provides front information even if the back capture times out, leaving the back JSON section empty.

This improvement enables integrators to implement their own post-processing based on their business rules, while ensuring that the SDK provides as much information as possible when the timeout is reached.

ProcessCard

Starting from version 4.2.0, IDENTY Web Server offers an endpoint to receive images from external systems and perform the OCR processing, returning the extracted data in structured JSON format.

The Web SDK is not required: any previously captured image (from internal apps, third-party platforms, or legacy systems) can be sent directly to the server endpoint.

This model is ideal for backend automation and integration into existing workflows.

Method: POST
Produces: "application/json"
Accepts: "application/json"
Headers:
Integrators can now log custom messages every time endpoints are accessed. To enable this feature, include the 'LogAPITrigger: true' header in each endpoint request.
Integrators also have the ability to log session IDs for each endpoint access. This allows them to easily monitor all activity within their apps for the corresponding session ID. To enable this feature, include the 'requestID' header in each endpoint request.

RequestsAttributes:

docType	String	docType to be scanned Uses the same list of supported identity documents as the OCR Web SDK.	n/a	Yes
requiredTemplates	String	Flexibility to choose between two output formats: JPEG and PNG. Please be aware that this feature only applies to the cropped front and back sections of the ID card. Face images, signatures, and other sections are returned as JPEG files.	n/a	Yes

Request example:

```
{
  "frontCapture": "iVBORw0KGgoAAAANSUhEUgAA",
  "backCapture": "iVBORw0KGgoAAAANSUhEUg",
  "docType": "INE_CARD",
  "requiredTemplates": [
    "PNG", "JPEG"
  ]
}
```

Response example:

```
{
    "AS": {},
    "back": {
        "barcodes": {},
        "data": {
            "Numero Documento": "1234567891234"
        },
        "docImage": {
            "JPEG": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAEBAQEBAQEBAQEBAQEBAQEBAQEB.."
        },
        "face": "",
        "faceImg": "",
        "hasQuality": true,
        "isGray": true,
        "isValid": true,
        "mandatoryFields": {}
    },
    "dataValidation": {},
    "front": {
        "barcodes": {},
        "data": {
            "Año de Registro": "1999",
            "Calle": "CALZ ",
            "Clave De Elector": "XXXXXXXXXXXXXXXXXX",
            "Codigo Postal": "02710",
            "Colonia": "COL ",
            "Curp": "XXXXXXXXXXXXXXXXXX",
            "Curp Valido": "SI",
            "Domicilio": "CALZ XXXXXX ,D.F.",
            "Edad": "37",
            "Emision": "2010",
            "Estado": "09",
            "Estado Domicilio": "D.F.",
            "Fecha De Nacimiento": "",
            "Folio": "0000011596361",
            "Localidad": "0001",
            "Municipio": "002",
            "Municipio/Delegación": "",
            "NUM EXTERIOR": "492",
            "NUM INTERIOR": "402",
            "Nombre": "",
            "Número de Emisión": "02",
            "Seccion": "0246",
            "Sexo": "H",
            "Vigencia": "2020"
        },
        "docImage": {
            "JPEG": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAEBAQEBAQEBAQEBAQEBAQEBAQEBAQEBAQ..."
        },
        "face": {
            "h": 312,
            "isGray": false,
            "w": 222,
            "x": 790,
            "y": 251
        },
        "faceImg":"/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAEBAQEBAQEBAQEBAQEB...",
        "hasQuality": true,
        "isGray": false,
        "isValid": false,
        "mandatoryFields": {
            "Año de Registro": true,
            "Clave De Elector": true,
            "Curp": true,
            "Domicilio": true,
            "Emision": true,
            "Nombre": true,
            "Número de Emisión": true,
            "Vigencia": true
        }
    },
    "refid": "W-ocr-processCard-bbe4e7ef-8231-46e5-b31e-ff426cac9fab-1755155120385",
    "subtype": "C"
}
```

The Web SDK is not required: any previously captured image (from internal apps, third-party platforms, or legacy systems) can be sent directly to the server endpoint.

This model is ideal for backend automation and integration into existing workflows.

Method:POST
Produces:"application/json"
Accepts:"application/json"
Headers:
Integrators can now log custom messages every time endpoints are accessed. To enable this feature, include the 'LogAPITrigger: true' header in each endpoint request.
Integrators also have the ability to log session IDs for each endpoint access. This allows them to easily monitor all activity within their apps for the corresponding session ID. To enable this feature, include the 'requestID' header in each endpoint request.

RequestsAttributes:

Parameter	Type	Description	Default	Mandatory
Capture	String	BASE64 of a POA Image.	n/a	Yes
docType	String	POA type to be scanned Uses the same list of supported POA documents as the OCR Web SDK.	n/a	Yes
requiredTemplates	String	Flexibility to choose between two output formats: JPEG and PNG. Please be aware that this feature only applies to the cropped front and back sections of the ID card. Face images, signatures, and other sections are returned as JPEG files.	n/a	Yes

Request example:

```
{
  "Capture": "JVBERi0xLjMKJcTl8uXrp/Og0M",
  "docType": "POA_CFE",
  "requiredTemplates": [
    "PNG", "JPEG"
  ]
}
```

Response example:

```
{
    "data": {
        "BARRIO": "",
        "CALLE": "XXXXXXXXXXXXXXXXXX",
        "CANTIDAD_PAGO_1": "599",
        "CANTIDAD_PAGO_2": "599",
        "COLONIA": "SAN XXXXXXXXXXXXXXXXXX",
        "CP": "52226",
        "DELEGACION": "XXXXXXXXXXXXXXXXXX",
        "ESTADO": "GON",
        "LIMITE DE PAGO": "03-JUN-2022",
        "NOMBRE": "XXXXXXXXXXXXXXXXXX

"

,
        "NUM EXTERIOR": "1",
        "NUM INTERIOR": "1",
        "PERIODO_FACTURACION": "11-MAY-2022"
    },
    "docBackImage": {},
    "docImage": {
        "PNG": "iVBORw0KGgoAAAANSUhEUgAABDgAAAWHC8g3/4...."
    },
    "error": "NO_ERROR",
    "hasQuality": true,
    "integrityCheckError": "OK",
    "mandatoryFields": {
        "CALLE": true,
        "COLONIA_DELEGACION": true,
        "CP": true,
        "ESTADO": true,
        "NOMBRE": true
    },
    "poaType": "POA_TELMEX",
    "refid": "W-ocr-processPOA-bbe4e7ef-8231-46e5-b31e-ff426cac9fab-1753346946683"
}
```

Customizing the UI

What´s customizable?

UX guidelines often advocate for employing diverse visual techniques to reinforce the same message. While repetition might seem redundant, it serves a crucial purpose in aiding user comprehension and retention.

Using varied visual methods—such as icons, colors, animations, or textual cues—to communicate a consistent message across different interfaces or interactions caters to diverse user preferences and cognitive styles. This approach not only reinforces the core information but also accommodates users who might respond better to specific visual cues, ensuring a more inclusive and comprehensible user experience.

On OCR captures, the SDK opens the device camera / Gallery to perform a high-quality document capture, presenting messages to users to guide them for an optimal capture.
UI objects on these screens are customizable as per your requirements:

String messages displayed to guide users while capturing, localized to the supported languages of your App
Which provide online feedback to users with guidelines to capture high quality images (e.g. please hold, please move the

Training Screen

In digital, unassisted workflows, clear graphical explanations of what to do significantly enhance user understanding and confidence, being specially relevant on initial interactions.
The OCR SDK accommodates various use cases such as capturing ID cards or A4 documents via camera or gallery, for both front and back, and more. Having a single generic training screen may not address specific needs or could lead to misunderstanding.
Hence, the SDK allows integrators to craft four distinct training layouts, each tailored for a particular business scenario:

- Camera-based ID card captures.
- Gallery-based ID card captures.
- Camera captures for A4 documents like POA receipts (e.g., CFE or Telmex).
- Gallery captures for A4 documents like POA receipts (e.g., CFE or Telmex).

The training screen is enabled by default, though it can be disabled on SDK instantiation. Additionally, when capturing ID Cards and requiring front and back captures, the SDK displays the training screen only once.

The *"Don't show me again"* option on the screen is stored in local storage. Once checked, the training screen won't reappear unless a new browser or tab is opened

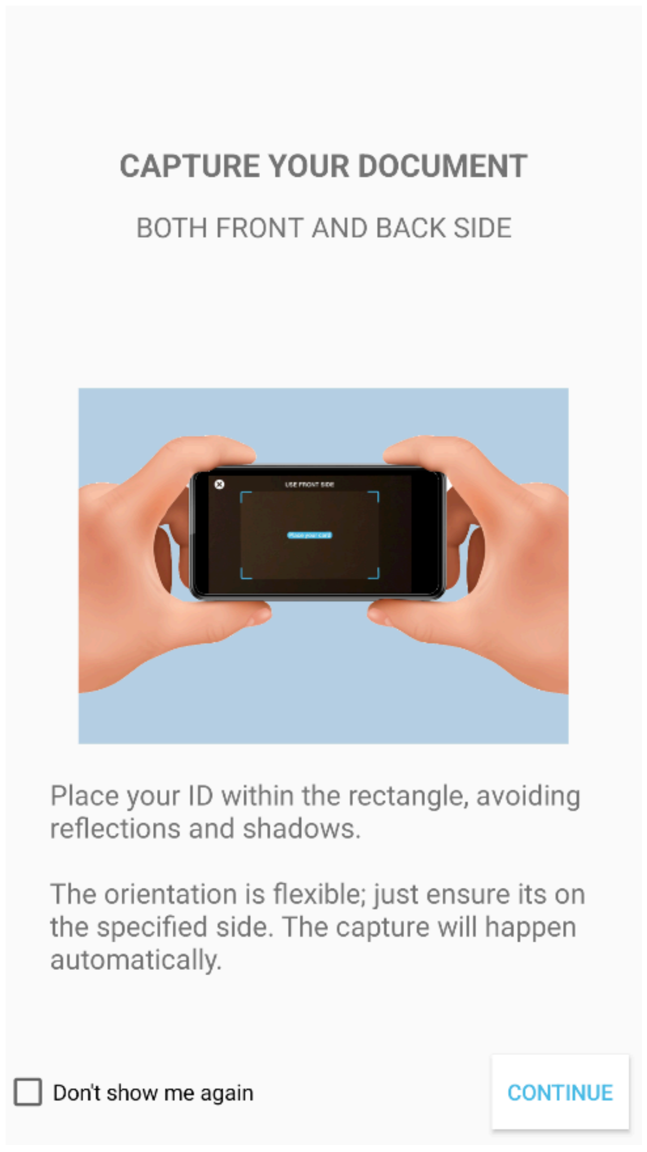
ID Card captures

Achieve complete customization of the ID card training screen on SDK instantiation.
Here's a snippet of code from our
Demo package, which you can download later in this
document.

```
      const options:
SdkOptionsType =
{
        allowClose:
true,
        detectionModes:
this.cardFaceSelection.cards,
        displayMode:
this.contextMenuSelection.selection.displayMode,
        a4IntegrityCheck:
this.contextMenuSelection.selection.integrityCheck,
        cardtype:
this.contextMenuSelection.selection.card_type,
        barcodeCheck:
this.contextMenuSelection.selection.barcodeCheck,
        maxNumberAttempts:
this.contextMenuSelection.selection.maxNumberAttempts,
        requiredTemplates:
[Template.PDF,
Template.JPEG],
        // graphics: {
        //   training: { <<--- THIS TRAINING SECTION
IS THE ONE YOU CAN CUSTOMIZE

        //     show:
true
               DIALOG_CONTENT: template as
any
        //
   }
        //
 },
        transaction:
{
          type:
transaction

},
```



CAPTURE YOUR DOCUMENT

BOTH FRONT AND BACK SIDE

Place your ID within the rectangle, avoiding reflections and shadows.

The orientation is flexible; just ensure its on the specified side. The capture will happen automatically.

☐ Don't show me again

CONTINUE

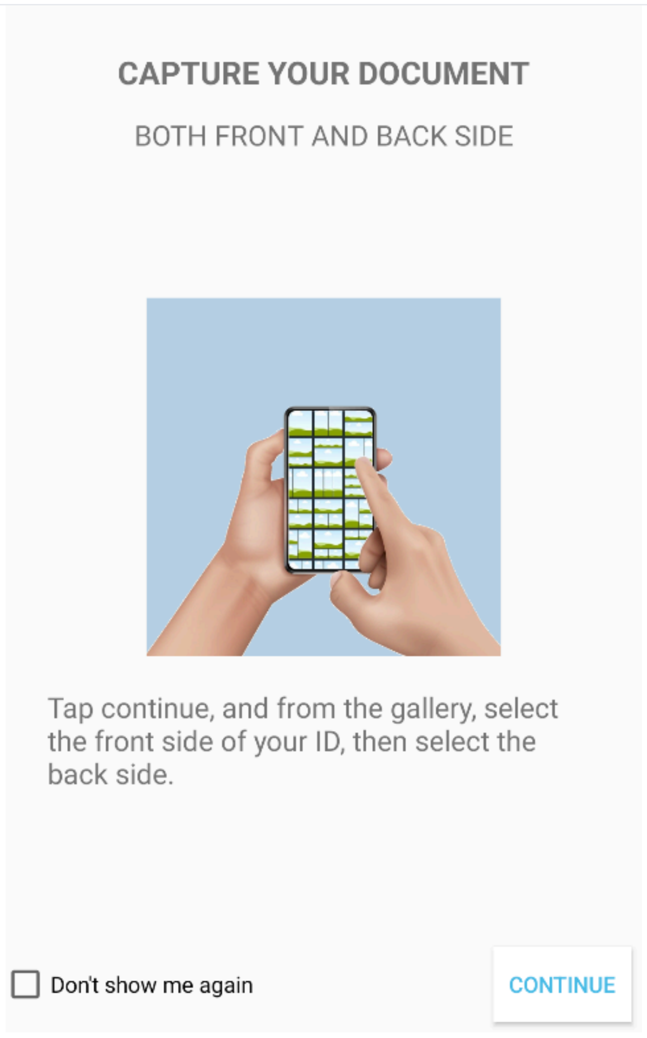
A4 captures

Achieve complete customization of the A4 training screen on SDK instantiation.

```
        allowClose:
true,
        detectionModes:
this.cardFaceSelection.cards,
        displayMode:
this.contextMenuSelection.selection.displayMode,
        a4IntegrityCheck:
this.contextMenuSelection.selection.integrityCheck,
        cardtype:
this.contextMenuSelection.selection.card_type,
        barcodeCheck:
this.contextMenuSelection.selection.barcodeCheck,
        maxNumberAttempts:
this.contextMenuSelection.selection.maxNumberAttempts,
        requiredTemplates:
[Template.PDF,
Template.JPEG],
        // graphics: {
        //   training: { <<--- THIS TRAINING SECTION
IS THE ONE YOU CAN CUSTOMIZE

        //     show:
true
               DIALOG_CONTENT: template as
any
        //
   }
        //
 },
        transaction:
{
          type:
transaction

},
```


CAPTURE YOUR DOCUMENT

BOTH FRONT AND BACK SIDE

Tap continue, and from the gallery, select the front side of your ID, then select the back side.

The DIALOG_CONTENT should direct to an HTML file where the setup for the training screen content is configured.
Below is an example illustrating the default ID capture screen.

```
   export  default
`<div
style="color: gray;background: white;position: absolute;top:
50%;-ms-transform: translateY(-50%);transform:
translateY(-50%);">
       <div style="margin:
15px">

     <div
style="text-align:
center;">

<h4>CAPTURE YOUR
DOCUMENT</h4>

</div>

<div style="text-align:
center;">

<h5>BOTH FRONT AND BACK
SIDES</h5>

</div>

</div>
       <div
class="id_image">

<img src="../../assets/images/ocr_instructions.png" style="width:
 80%" alt="">

</div>

<div style="margin: 15px;padding: 10px;text-align:
left">

  Place your ID within
the rectangle, avoiding reflections and
shadows.

 <br>

The orientation is flexible; just ensure its on the specified side. The
capture will happen
automatically.

</div>
   </div>
   `;
```

Capture Screen

You can change colours of the capture screen when initializing the OCR SDK

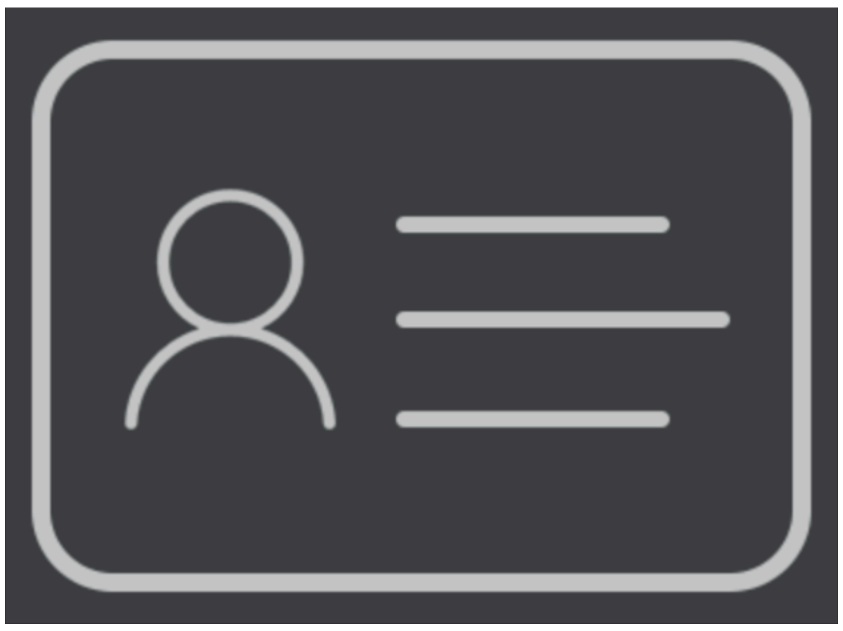
```
        label: '#ffffff', <-- Colour of the real time feedback text
        labelBackground: '#00b7ea', <-- Colour of the real time feedback box background
        guideBorderColor: '#77bae7' <-- Colour of the box capture area
      }
      ...
  })
```

Capture Silhouettes

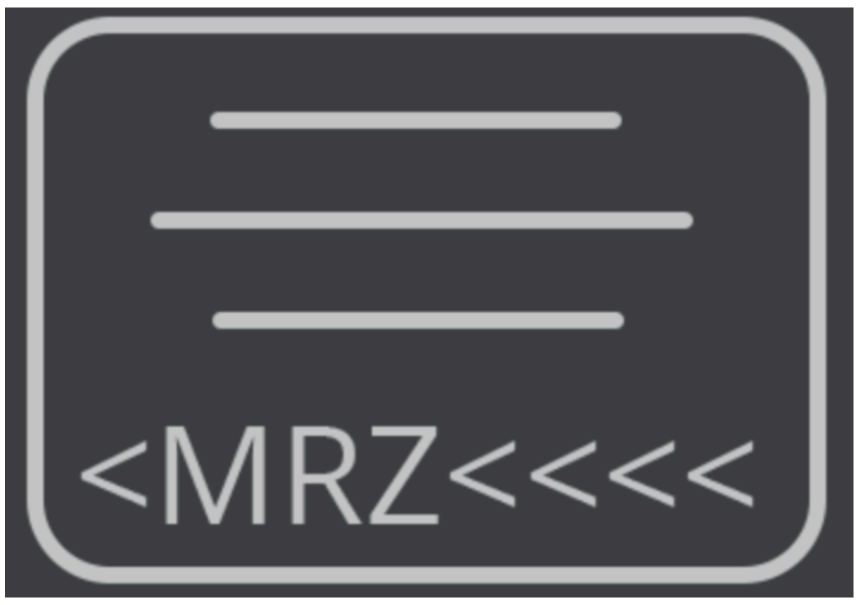
The SDK now assumes the role of a personal guide throughout the entire business process. Upon capture initiation, a blinking guide silhouette appears, signalling users to present their card accurately.
The SDK comes equipped with two default silhouettes—one for the front and another for the back of the card. Additionally, integrators have the flexibility to customize these silhouettes as needed, ensuring they align perfectly with the specific layout of the card being captured. For instance, accommodating ID cards where the facial area is positioned on the right side of the front card.
By default, the presentation of card silhouettes is activated, displaying the front side for 6 seconds and the back side for 2 seconds. The silhouette blinks every 300 milliseconds during display.
Integrators can tailor the duration of the silhouette to align with their UX guidelines. This customization is achieved on SDK initialization.

```
new CardOcrSDK({
  graphics: {
    ...
    silhouette: {
      enable: true,
      CARD_FRONT: Images.CARD_SILO_FRONT, // (URL or Base64)
      CARD_BACK : Images.CARD_SILO_BACK, // (URL or Base64)
      A4_BACK: "",
      A4_FRONT: "",
      ...
    },
    timings: {
      silhouette_front_duration: 6000, // 6 sec
      silhouette_back_duration:  2000, // 2 sec
      silhouette_front_blink_interval: 300, // 0.3 sec
      silhouette_back_blink_interval: 300, // 0.3 sec
      ...
    }
   ...
  }
})
```

Default front side Silhouette



Default back side Silhouette



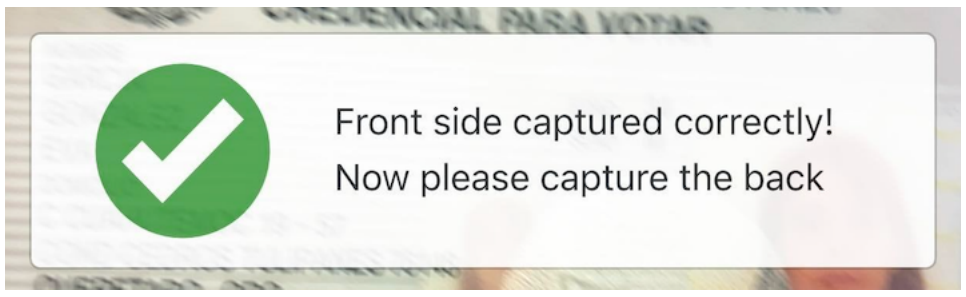
Capture Success feedback

In its role guiding users through the complete business process, the SDK delivers a confirmation message after capturing either the front or back side.

On SDK initialization, the SDK offers the *onCardFaceCaptureSuccess* event when either front or back are captured.

```
const options: SdkOptionsType = {
    allowClose: true,
    detectionModes: this.cardFaceSelection.cards,
    displayMode: this.contextMenuSelection.selection.displayMode,
    a4IntegrityCheck: this.contextMenuSelection.selection.integrityCheck,
    cardtype: this.contextMenuSelection.selection.card_type,
    barcodeCheck: this.contextMenuSelection.selection.barcodeCheck,
    maxNumberAttempts: this.contextMenuSelection.selection.maxNumberAttempts,
    requiredTemplates: [Template.PDF, Template.JPEG],
    transaction: {
      type: transaction
    },
    useFlash: this.contextMenuSelection.selection.useFlash,
    events: {
      onCardFaceCaptureSuccess:(face: string) => {
        return new Promise((resolve, reject) => {
          const dialog = this.modalService.show(CardFaceComponent, {
            class: "modal-dialog-centered modal-sm txn-modal-dialog modal-card-face",
            backdrop:true,
            ignoreBackdropClick: true,
            initialState: {
              card_face: face
            }
          });
          setTimeout(() => {
            $(".modal-card-face").parent("modal-container").css("z-index", 99999999);
            if(window.orientation === 0) {
              $(".modal-card-face").find(".modal-content").css("transform", "rotate(90deg)");
            }
            setTimeout(() => { dialog.hide() ;resolve(null)}, 3000)
          }, 10);

        });
      },
    ....,
    debug: true,
    selectAsFile: this.contextMenuSelection.selection.fileSelection
  };
  if (card_type === DocumentType.A4_POA_TELMEX || card_type === DocumentType.A4_POA_CFE) {
    const map = new Map<MandatoryFieldValue, boolean>();
    for (const mandatoryField of this.contextMenuSelection.selection.mandatoryFields) {
      map.set(mandatoryField.field, mandatoryField.selected)
    }
    options.mandatoryFields = map;
  }
  const cardOcrSDK = new CardOcrSDK(options);
  ...
```



Flipping Animation

When capturing both sides of an ID Card, the SDK now incorporates an animation prompt after capturing the ID card's front side. This animation serves as a clear indication to users to flip the card and capture the back side.

When activated, you can adjust its duration using **silhouette_flip_duration**, measured in milliseconds.
You can disable this animation by setting *animate_flip* to **false** on SDK initialization.

```
new CardOcrSDK({
  graphics: {
    ...
    silhouette: {
      animate_flip: true,
      CARD_FRONT: Images.CARD_SILO_FRONT, // (URL or Base64)
      CARD_BACK : Images.CARD_SILO_BACK   // (URL or Base64)
    },
    timings: {
      silhouette_flip_duration: 1000
    }
    ...
  }
})
```

Language localization

In order to do so, just use the Localization type. The different messages in English used by default are shown below.

```
  get locales(): LocalizedStringsMethods {
    return this._locales;
  }

  set locales(value: LocalizedStringsMethods) {
    this._locales = value;
  }
  get language(): string {
    return this._language;
  }

  set language(value: string) {
    this._language = value;
  }

  private _language = 'en-EN';

  private _locales: LocalizedStringsMethods;

  constructor() {


    this.locales = new LocalizedStrings({
      en : {
        FEEDBACK_CAPTURED: 'Captured Successfully',
        FEEDBACK_SEARCHING_BARCODE: 'Position Barcode in the frame',
        FEEDBACK_SEARCHING: 'Use {0} Side',
        FEEDBACK_SEARCHING_2: 'Place your card',
        FEEDBACK_SEARCHING_A4: 'Please place your document',
        FEEDBACK_SEARCHING_A4_2: 'Searching for form, Move away',
        FEEDBACK_SEARCHING_QRCODE: 'Place your Qrcode',
        FEEDBACK_INSIDE_GUIDE: 'Please be inside the guide',
        FEEDBACK_INSIDE_GUIDE_A4: 'Move further away',
        FEEDBACK_PLEASE_HOLD: 'Please hold.',
        FEEDBACK_STABLE: 'Please be stable',
        FEEDBACK_A4_LEFT_ALIGNED: 'Please center document',
        FEEDBACK_A4_RIGHT_ALIGNED: 'Please center document',
        FEEDBACK_CLOSE: 'Please move closer',
        FEEDBACK_FAR: 'Please move further away',
        FEEDBACK_BLURRY: 'Wait for camera to focus',
        FEEDBACK_TILTED: 'Card too rotated',
        FEEDBACK_NOT_STABLE: 'Please be stable',
        FEEDBACK_OK: 'Please Hold',
        FEEDBACK_ATTEMPT_TIMEOUT_EXCEEDED: 'Timeout exceeded.',
        FEEDBACK_NO_DATA: 'Please be stable and avoid shadows or reflections',
        FEEDBACK_BARCODE_NO_DATA: 'Cannot extract Barcodes from image',
        FEEDBACK_NOT_PARALLEL: 'Please be parallel to doc',
        FEEDBACK_CAMERA_ACQUIRING_FAILED: 'Cannot acquire camera, Allow permission Or Retry Capture',
        FEEDBACK_PROCESSING: 'Processing...',
        FEEDBACK_UPLOADING: 'Uploading debug images...',
        FEEDBACK_UPLOAD_FAILURE: 'Internal error retry',
        FEEDBACK_INITIALIZATION: 'Initializing...',
        FEEDBACK_CROPPING: 'Cropping...',
        FEEDBACK_RECOGNITION: 'Running OCR...',
        FEEDBACK_CHANGE_LIGHT: 'Avoid shadows',
        FEEDBACK_BUTTON_CLOSE: 'Close',
        FEEDBACK_BUTTON_RETRY: 'Retry',
        FEEDBACK_TRAINING_BUTTON_NEXT: 'Next',
        FEEDBACK_TRAINING_LABEL: 'Dont Show again',
        FEEDBACK_LICENCE_INVALID: 'License Invalid!!',
        FEEDBACK_ORIENTATION_NOT_SUPPORTED: 'Only Portrait mode is supported',
        FEEDBACK_CHECK_QUALITY_MESSAGE: 'Please check that image is clear, with no blur or glare.',
        FEEDBACK_CAPTURE_BACK_SIDE: 'Select the backside of card',
        ERROR_FILE_FRONT_WRONG_SIDE: 'Invalid front side selected',
        ERROR_FILE_BACK_WRONG_SIDE: 'Invalid back side selected',
        ERROR_FILE_WRONG_POA_TYPE: 'Wrong document type selected',
        ERROR_BROWSER_NOT_SUPPORTED: 'Browser not supported, Please update to Latest Chrome, Firefox (Android) or
Safari on IOS.',
        ERROR_WEBRTC_NOT_SUPPORTED: 'Webrtc not supported',
        ERROR_MODEL_FAIL: 'Model detection failed',
        ERROR_SERVER_INTERNAL: 'Internal Server Error',
        ERROR_DEVICE_NOT_SUPPORTED: 'WEBGL Not supported.',
        ERROR_SERVER_CONNECTION_FAILURE: 'Server connection failure',
        FEEDBACK_BACK_WRONG_SIDE: 'Please put the back side',
        FEEDBACK_FRONT_WRONG_SIDE: 'Please put the front side',
        FEEDBACK_WRONG_POA_TYPE: 'Present a valid POA document',
        FACE: {
          'FRONT': 'Front',
          'BACK': 'Back',
          'QRCODE': 'QRCODE'
        },
        BUTTON: {
          'CROP': 'Crop',
          'NEXT': 'Continue',
          'CANCEL': 'Cancel',
          'RETAKE': 'Retake',
        },
      }
    });

    this.setLanguage(this.language);
  }

  getString(label: string) {
    return this.locales.getString(label, this.locales.getLanguage(), true);
  }

  setLanguage(language: string) {
    this.locales.setLanguage(language);
  }

}
```

Figure 8. Default English messages

OCR Web SDK

Before installing the OCR Web SDK, please remember that the IDENTY solution supports two mutually exclusive architectures, each utilizing a different OCR Web SDK. Please refer to the architecture section for an in-depth explanation of both architectural designs.

OCR Web SDK
This architecture fully runs client-side, capturing and extracting data from cards and A4 documents using OCR technology. The data is returned as a flat JSON by the SDK, with the integrator responsible for obfuscating their App to ensure data manipulation and integrity.

This SDK is named @identy/identy-ocr, and each released version follows this syntax: @major.minor.patch-fix., e.g. @3.1.0-b02.

OCR Web SDK encrypted + IDENTY Web Server
This architecture requires both the SDK to run client-side and the IDENTY Web Server running on the backend.
The OCR Web SDK still captures and extracts data from cards and A4 documents using OCR technology. However, instead of returning the information as a flat JSON, it's encrypted within the SDK and returned as a encrypted BLOB.

To decrypt the data BLOB, it should be sent to the IDENTY Web Server running in a secure backend environment.

Each IDENTY Web Server can only decrypt BLOBs from licensed domains, as the RSA keys used for encryption are dynamically associated with the corresponding license.

This SDK is named @identy/identy-ocr, and each released version follows this syntax: @major.minor.patch-fix**E**., e.g. 3.1.0-b02E.
Note the capital E at the end of the OCR version number. That E stands for Encrypted, indicating that this SDK returns the encrypted BLOB with the extracted data. This helps you differentiate this version from the one that returns data in clear format.

Please note that each time a license is downloaded from the License Manager, it may differ due to encryption. When deploying the SDK and IWS, ensure that the same BASE64 string is used on both sides. Otherwise, you may encounter a license error.

Prerequisites

Install SW dependencies:

yarn

Architecture design. Which of the supported IDENTY architectures best fulfills my needs?

Request your Jfrog credentials:
IDENTY delivers its SDKs through an artifact management platform named JFrog, giving you the ability to keep your Application up to date our technological improvements with just a few lines of code.
Ask your IDENTY's representative assistance to get your user/password to this platform, or email us to support@identy.io.
IDENTY grants access to JFrog at a company level, not individually.

Request your IDENTY Web OCR license to your IDENTY´s representative.

Request your IDENTY representative to license all the domains where the OCR SDK will be used.

Step-by-Step guide to deploy OCR Web SDK

Setting up the IDENTY Repository

Edit the .npmrc file:

On the server where you will deploy your application, edit the .npmrc file.

If the file does not exist, create an empty one.

File Location:

Linux: Located inside `~/`

Windows: Located inside `C://Users//<YourUserName>//`

Add Credentials:

Add the following lines to the .npmrc file, replacing the placeholder tags with your actual credentials. Make sure to encode the provided password in BASE64.

```
registry=https://registry.npmjs.org/
@tensorflow:registry=https://identy.jfrog.io/identy/api/npm/identy-npm/
@identy:registry=https://identy.jfrog.io/identy/api/npm/identy-npm/
//identy.jfrog.io/identy/api/npm/identy-npm/:_password=<YOUR_BASE_64_PASSWORD>
//identy.jfrog.io/identy/api/npm/identy-npm/:username=<YOUR_JFROG_USERNAME>
//identy.jfrog.io/identy/api/npm/identy-npm/:email=<YOUR_JFROG_EMAIL>
//identy.jfrog.io/identy/api/npm/identy-npm/:always-auth=true
```

In your project, remove any yarn.lock file that may exist from previous compilations.

Install OCR Web SDK by running the following commands for specific versions. Choose between the encrypted or clear version depending on your architecture requirements.

```
yarn add @identy/identy-ocr@x.y.z
yarn add @identy/identy-common@r.s.t
```

As, for instance:

```
yarn add @identy/identy-ocr@3.1.0-b02E
yarn add @identy/identy-common@3.0.0
```

Note the capital E at the end of the OCR version number. That E stands for Encrypted, indicating that this SDK returns the encrypted BLOB with the extracted data. This helps you differentiate this version from the one that returns data in clear format.

IDENTY Web Server

Prerequisites

Before installing IDENTY Web Server, please ensure you have met all prerequisites:

Downloaded the demo project package, available at the end of this tutorial.

Requested an OCR IDENTY Web Server license to your IDENTY´s representative.

Installed Docker and Docker-compose on your host.

Installation time

IDENTY Web Server takes less than 5 minutes to install. Most of that time is required to download the images from the JFrog repository.

This component also verifies the customer license online with the IDENTY Licensing Cloud: https://licensemgr.identy.io. Customers shall make sure that the IDENTY Licensing Cloud URL is reachable by the Web Server.

identy-web will be deployed through the docker-compose file provided in the Demo project.

Step-by-Step guide to deploy IDENTY Web Server

Please follow these step-by-step instructions:

Unzip the demo package, provided at the end of this tutorial. It contains:

OCRDemoApp folder, including an Angular Web Application that integrates and exercises IDENTY OCR Web SDK.

IDENTY Web Server deplyment files.

On your server, open a console and copy all files required for the IDENTY Web Server deployment under the same folder.

Navigate to that folder.

The data and redis-data folders should be owned by user "998" for proper read access. Create them if not already existing. Execute the following *chown* command to achieve this:

```
chown -R 998:998 redis-data/
chown -R 998:998 data/
```

Edit the *lic_config.json* file, living within the same directory where the *docker-compose-ocr.yml* file is located. Populate it with your base64 license content:

```
{
  "ocr" : {
    "license": "<REPLACE_BY_YOUR_VALID_OCR_LICENSE_IN_BASE64>"
  }
}
```

Note that, for deployments intending to also support face or finger functionality, it is needed to include a corresponding "face" and "finger" sections within the *lic_config.json* file.

Edit the *env-sample-ocr* file if you want to edit default ports:

```
TOMCAT_HTTP=8080
TOMCAT_SHUTDOWN=8005
TOMCAT_AJP=8009
LOG_LEVEL=INFO
SERVER_CONFIG_PATH=/opt/data/lic_config.json
JAVA_OPTS="-Dport.http=${TOMCAT_HTTP} -Dport.ajp=${TOMCAT_AJP} -Dport.shutdown=${TOMCAT_SHUTDOWN}"
```

Run *"docker-compose -f docker-compose-ocr.yml --env-file env-sample-ocr up -d"* to start up the solution in the background.

You can check the server is up & running by invoking the ocr_health API to get back the OK response: *http://localhost:8080/api/v1/health_lb*

Monitor docker started correctly by running *docker-compose --env-file env-sample-ocr -f docker-compose-ocr.yml logs -f*

You are done!

To ensure secure communication and protect sensitive data, it is crucial to expose the service over HTTPS. By enabling HTTPS, all data transmitted between the client and the service will be encrypted, preventing unauthorized access or tampering. This added layer of security safeguards user information, authentication credentials, and other critical data from potential threats.

The Demo project

The fastest way to get familiar with IDENTY Web OCR Solution is by playing with our demo project, which includes everything required for a complete end to end test.

Demo project content

We have released two demo packages, one for each supported secure architecture. Each package includes an Angular Web Application that integrates and demonstrates the corresponding IDENTY OCR Web SDK.

OCR Web SDK
This architecture fully runs client-side, capturing and extracting data from cards and A4 documents using OCR technology. The SDK returns the data as a flat JSON, with the integrator responsible for obfuscating their app to prevent data manipulation and ensure integrity.

Download Demo

OCR Web SDK encrypted + IDENTY Web Server
This architecture requires both the SDK to run client-side and the IDENTY Web Server running on the backend.
The OCR Web SDK still captures and extracts data from cards and A4 documents using OCR technology. However, instead of returning the information as a flat JSON, it's encrypted within the SDK and returned as an encrypted BLOB.

To decrypt the data BLOB, it should be sent to the IDENTY Web Server running in a secure backend environment.

Each IDENTY Web Server can only decrypt BLOBs from licensed domains, as the RSA keys used for encryption are dynamically associated with the corresponding license.

This same Web Server package can also be used for the new `processPoa` and `processCard` endpoints introduced in version 4.2.0.

Download Demo

Install Angular CLI.

Install OCR Web SDK as described above. Remember, the OCR Web SDK is licensed, and you must add every single licensed domain to your license. Please contact your IDENTY representative if you need assistance with this.

Update the IDENTY dependencies to the latest OCR SDK release on the yarn file.

Run "yarn install" to install Web OCR SDK and the required App dependencies.

Configure your license
For that, navigate to "*OCRDemoApp/src/main.ts*" and edit the license variable, with your BASE64 encoded license.

If choosing the architecture that encrypts the response, please ensure to use exactly the same BASE64 configuration as set for the IDENTY Web Server.

```
letLICENSE:string;
LICENSE="QUVTAgAAACMcK39Has/osVX9Y...";


//RUN '''export LICENSE="'<Base64_License>'";yarn start'''; If you dont want to replace the LICENSE here.

let license=LICENSE
```

If opting for the architecture that encrypts the response, you will also need to:

Install OCR IDENTY Web Server as described above.

Please note that each time a license is downloaded from the License Manager, it may differ due to encryption. When deploying the SDK and IWS, ensure that the same BASE64 string is used on both sides. Otherwise, you may encounter a license error.

Set up the IDENTY Web Server's listening address for sending POST requests in the environment.ts file.

Run ng serve

You are all set!, the Demo App on your Desktop is running on http://localhost:4200

Don't show again is set to local storage and never shown again otherwise set to sessionstorage which means it won't be shown in that tab