# Common Core Banking System API

The IA-CCBS API (Aka i-Apply Common Core Banking System Application Program Interface) aims to interconnect I-Apply with the core banking system and the card platform if it exists in the Bank's environment. For further information, please refer to paragraph "Architecture" below.

Contents

[Common Core Banking System API 1](#_Toc184042257)

[Version 2](#_Toc184042258)

[Terms & Technologies 2](#_Toc184042259)

[API 2](#_Toc184042260)

[Open API 2](#_Toc184042261)

[RESTful API 2](#_Toc184042262)

[Architecture 3](#_Toc184042263)

[Green layer – Bank's involvement 3](#_Toc184042264)

[iApply light CCBS Integration 5](#_Toc184042265)

[The Common CBS API 5](#_Toc184042266)

[Account Integration 5](#_Toc184042267)

[Collateral Maintenance Integration 6](#_Toc184042268)

[Credit Limit 6](#_Toc184042269)

[Customer Integration 6](#_Toc184042270)

[Customer Position Integration 6](#_Toc184042271)

[Deposit Account Integration 7](#_Toc184042272)

[Fees 7](#_Toc184042273)

[LG Beneficiary 7](#_Toc184042274)

[Mortgage Integration 7](#_Toc184042275)

[API Implementation 8](#_Toc184042276)

[Default values 8](#_Toc184042277)

[The elementary interface 8](#_Toc184042278)

[Common fields 9](#_Toc184042279)

[Request class. 9](#_Toc184042280)

[Response class. 10](#_Toc184042281)

[HTTP Headers 12](#_Toc184042282)

[Example API Call 13](#_Toc184042283)

[*Headers* 13](#_Toc184042284)

[*Request Body* 13](#_Toc184042285)

[*Response Body* 14](#_Toc184042286)

[How to start & resources 15](#_Toc184042287)

[Recommended OpenAPI Tools 17](#_Toc184042288)

[Further concepts 18](#_Toc184042289)

[Absolute necessary API calls 18](#_Toc184042290)

[Position Entities & flow. 18](#_Toc184042291)

[Position flow step 1 19](#_Toc184042292)

[Position flow step 2 19](#_Toc184042293)

[Position flow step 3 19](#_Toc184042294)

## Version

2/12/2024 Chapter: Common Fields

10/5/2024 Additions to the document.

21/9/2023 New paragraph about CCBS

16/8/2023 Initial version of this format. All previous documents are obsolete.

## Terms & Technologies

### API

An application programming interface (API) defines the rules you must follow in order to communicate with other software systems. Developers expose or create APIs so that other applications can communicate with their applications programmatically.

### Open API

The Common-CBS API is an Open API compliant specification. The definition of an Open API does not need further explanation as it is an industry-standard. In brief, Open API can provide a definition of i-Apply API to other organizations (such as the Banking Software providers) who have to implement their services following that specification. Open API provides a standardized method to do this. Our API is described in agnostic terms, decoupling them from any specific programming language. Consumers of our API specification (usually the Banking Software providers) do not need to understand the core of our application. They can understand precisely what they need from our API specification, written in a simple and expressive language. The OpenAPI Specification (OAS) enables this knowledge transfer from the API provider to the API consumer. It is an open standard for describing your APIs, allowing you to provide an API specification encoded in a JSON or YAML document. It provides a comprehensive dictionary of terms that reflects commonly-understood concepts in the world of APIs, thus embedding the fundamentals of HTTP and JSON. Teamed up with supporting tools can provide a rich experience based on a simple document.

### RESTful API

RESTful API is an interface that two computer systems use to exchange information securely over the internet or a controlled network. Most business applications must communicate with other internal and third-party applications to perform various tasks. RESTful APIs support this information exchange because they follow secure, reliable, and efficient software communication standards.

## Architecture

The diagram d1 presents the interfaces layer and the communication between the layers. The i-Apply application server (cyan color on the diagram) sends business requests to the i-Apply CCBS layer (orange). The IA-CCBS performs mappings based on a) the Country, b) the Bank, and c) the actual CBS and performs a sequence of validations. When necessary, it retrieves data from the i-Apply database (IADB) and the i-Apply file system (IAFS). Preparing and using a request invokes a RESTful web service call to the CCBS API (green) layer. The Bank usually develops the green layer with the help of the CBS and Cards provider(s). The typical action is to call the Core Banking Software and do what the method (e.g., SearchCustomer, OpenAccount, etc.) aims to do. Finally, prepare a valid response and return it to the IA-CCBS (orange) layer. At that point, the IA-CCBS evaluates the returned response, creates logs, errors, and data mappings to i-Apply terms and codes, and performs the necessary data persistence. The last step is returning a response to the IA-CBS (orange) and that layer to the initiator (cyan layer).

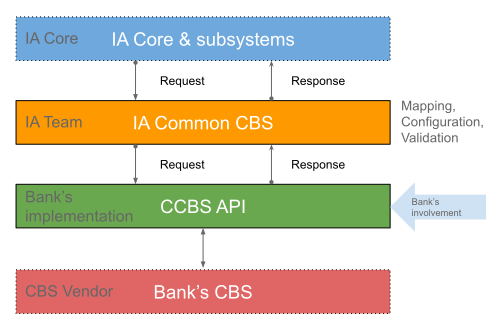


Diagram d1. Layers of CBS Interface

## Green layer – Bank's involvement

The green layer of the diagram is what the provider of the Banking Software must implement for the i-Apply to CBS interfacing. The orange layer of the architectural diagram d1 will do the following process:

1. Retrieve from the database further data if needed.
2. Map the i-Apply codes to Bank's-CBS-specific codes.
3. Invoke the RESTful API method call and handle the result.
4. In case of an error will return the error to the i-Apply
5. in case of a successful invocation, the Bank's-CBS result codes will be mapped to i-Apply codes and stored the results in the database (if necessary).
6. Will return that response back to the i-Apply system.

# iApply light CCBS Integration

The iApply CCBS Integration uses 99 API calls to offer integration between the Core Banking System (CBS) and the iApply platform. The light version uses only 53 integration points (API calls), organized into folders and priorities.

The **folder** indicates where the supplementary information of each call is stored on the file system, on GitHub, and in the OpenAPI interface.

The **Priority** indicates the sequence of the particular deliverables we expect from the side of the provider (in most cases, the vendor of the core banking system)—The Priority aims to offer a “Proof of Concept” and the elementary integrations between the two systems.

# The Common CBS API

The following text contains information regarding the Excel file and lists all the available API calls. Among others, it contains the following columns:

* **Colour:** only the green calls are available in the light iApply version.
* **Group** is the group's name in which this document's current chapter is organized.
* **Folder** is the name of the folder on the file systems (and other places) where the reader can find further information.
* **Priority** is the Priority mentioned above in the text.
* **iApply Interface** is the name of the API call.
* **Invocation** contains an indication of where the call is invocated by the iApply platform.
* **Client** contains some more information than the invocation regarding the API call.
* **Excel File Names** are the names of the Excel files the reader can find in the appropriate folder, containing the input and the output parameters of the API call (request and response)

## Account Integration

The account integration group contains interface API calls used to open and manipulate accounts to the CBS. In total, it contains 12 API calls organized into 3 priorities and the following folders:

1. **Repayment Schedule.** It is the set of calls aiming to create and retrieve repayment schedules as a simulation thus being an official part of the application process.
2. **Drawdown.** Is the Partial disbursement of an amount of an application form.
3. **Account Integration.** Any other call is related to the account.
4. **LG.** Used to manipulate LG family of products. In some setups, it might be used to manipulate other similar products.

All the calls depend on a user action by selecting it on the CBS operations menu or by clicking a button.

The calls, marked as priority 2, are necessary for iApply/Light minimum CBS integration.

## Collateral Maintenance Integration

The collateral maintenance integration group contains interface API calls used to maintain the application form's collaterals. The collaterals can be movable, immovable, or any other type. In total, it contains 7 API calls organized in a single priority, and one folder.

All the calls depend on a user action by selecting it on the CBS operations menu. An exception is the API call "*DeleteCollateralMaintenanceDetails*," which executes automatically in the iApply operation "Delete Collateral Allocation, existing in CBS."

## Credit Limit

The credit limit group contains interface API calls to maintain the application form's collaterals. The collaterals can be movable, immovable, or any other type. In total, it contains 4 API calls organized in a single priority and one folder.

The limits are the same for all the products (e.g., Overdraft limit). Note that when iApply retrieves the limit simultaneously, it retrieves the available remaining amount.

All the calls depend on a user action by selecting it on the CBS operations menu.

## Customer Integration

The Customer Integration group contains interface API calls to retrieve and update or create the customer (applicant) and the other participants to the CBS. The integration can be simple or very complex when the iApply is used to manipulate all the customer details to the CBS. In total, it contains 14 API calls organized into 2 priorities and one folder. The term customer in iApply means any individual or legal entity participating in an application form with any of the roles of applicant, guarantor, representative, administrator, spouse, and many others. However, it is a matter of configuration and, in some cases, implementation of the kind of customer will be exchanged with the CBS.

The API calls under priority 1 are necessary for elementary cooperation between the CBS and the iApply/Light minimum CBS integration. There are mainly calls to search for an existing customer and retrieve back to the i-Apply the current instance of his details.

Most calls depend on a user action by selecting it on the CBS operations menu by clicking a button on a UI area. Only two calls are in-code orchestrated and executed automatically to maintain the customer addresses.

## Customer Position Integration

The Customer Position Integration group contains interface API calls used to retrieve the customer's existing position in the Bank. That information contains the existing exposures and any other obligations to the Bank related to the principal applicant or parties. Also, the information regarding the collaterals is already known to the bank products. Additionally, the loan insurance is expected to affect the results. In total, it contains 4 API calls organized in a single priority and one folder.

The API calls are under priority 1 and are necessary for elementary cooperation between the CBS and the iApply/Light minimum CBS integration. They are only retrieving information and do not perform any data modification.

All the calls depend on a user action by selecting it on the CBS operations menu by clicking a button or on a UI area.

## Deposit Account Integration

The Deposit Account Integration group contains interface API calls used to retrieve and maintain the deposit account on the side of the CBS. Only one call was used to retrieve information, and the others used to modify the accounts. In total, it contains 4 API calls organized in a single priority and one folder.

The same calls are used for similar types of accounts, such as the Service and the Overdraft accounts.

All the calls depend on a user action by selecting it on the CBS operations menu and clicking. The exception is one call to retrieve the existing account information when the user selects the UI as an overdraft account.

## Fees

The Fees group contains a few interface API calls used to maintain the fees on the side of the CBS. In total, it contains 2 API calls organized in a single priority and one folder.

All the calls depend on a user action by selecting it on the CBS operations menu.

## LG Beneficiary

The LG beneficiary group contains a few interface API calls used to retrieve and update the LG type of product beneficiaries on the side of the CBS. In total, it contains 2 API calls organized in a single priority and one folder.

All the calls depend on a user action by selecting it on the CBS operations menu.

## Mortgage Integration

The Mortgage Integration group contains interface API calls used to retrieve and update the information needed by the type of the mortgage family of products on the side of the CBS. In total, it contains 4 API calls organized in a single priority and one folder.

All the calls depend on a user action by selecting it on the CBS operations menu.

# API Implementation

Each API call has one expected response structure and returns one specific result structure. Both are described in two points:

* The Open API specification covers the total technical needs. Using online tools, the developer can convert iApply's request and response structures into programming language-specific data structures.
* The Excel files are organized in folders where the reader will find further information in texts and "last minute" corrections or clarifications.

## Default values

In both responses and results and when a field value is null, the CCBS subsystem expects that the vendor will specify (on request) and provide (on results) the default values coming up from the configuration of the CBS system for the specific Bank (solution) and the product-specific default values when this is applicable. Consider that note as crucial as the iApply does not reproduce the CBS setup or know the entire product factory.

## The elementary interface

With the term elementary interface, i-Apply means the necessary API methods that the Bank must implement to start working with the i-Apply. That crude interface is needed to set up and work in a fundamental way and a startup version of the i-Apply. Does not provide any of the necessary automation available in the entire API version but offers a starting point for the Bank. Later the rest of the API calls must be implemented by the Bank (or the CBS provider).

The list of the supported API calls is organized into groups, and there are the following:

* Group G00, with informative API calls. Used to explain to the i-Apply the CBS system and as a test for the connectivity.
* Group G01. Customer basic data. It is a set of methods to search a customer in the CBS and retrieve the primary data (such as demographics).
* Group G02. Position-related methods. Used to extract the customer's current obligations (position) to the Bank.

All the other API calls to the elementary interface are characterized as Group 99 (other calls).

***For more information see: Further concepts -> Absolute necessary API calls.***

## Common fields

The CCBS OpenAPI specification mandates the inclusion of certain standard fields in all Response and Request classes. While these fields are not explicitly listed in the Excel file documentation, they are consistently present across all defined classes. Furthermore:

### Request class.

A screenshot of a computer program

Description automatically generatedIn the realm of RESTful API development, the Request class serves as a pivotal component, defining the structure and content of data sent to a server. It acts as a blueprint, outlining the specific parameters, headers, and body that will be transmitted in an API request.

JSON example:

{

"mockupKey": "string",

"additionals": "object",

…

…

"surname": "string"

…

}

The request class is used to send data values to the CCBS API interface. These values can either specify the desired process or data structures, or they can be used to update data within the CBS.

**Field: mockupKey:** This field is designated for internal i-Apply purposes. It is not intended for external use and should always be set to a null value.

…

"mockupKey": null,

…

**Field: additionals:**

This field provides a flexible mechanism to include custom data structures that are not explicitly defined in the OpenAPI specification. It can contain any JSON object, regardless of complexity, and is intended to be interpreted as a dynamic type (eg any, dynamic, object) by the CCBS API implementation.

To ensure compatibility and correct processing, the i-Apply team and the Bank's CBS provider must establish a shared understanding of the expected data structure for this field. In the absence of such an agreement, the field should be assigned a null value.

"additionals": null,

Most of the times the shared understanding is reflected in the Business requirement document (BRD).

Here is an example of that field when containing additional to the standard request data:

{

…

"AmortGPInter": 0,

"CustomerCBSCd": 0,

"CustomerCBSId": "CU1",

"MockupKey": "",

**"additionals":** {

"CoinsuredDataList": null,

"cu\_id": "1234",

"CustomerDataList": null,

"la\_appamt": 0,

"la\_branc": null,

"la\_custcat": null

…

…

},

…

…

}

**Important to remember:** The "*additionals*" structure is not explicitly defined in the Open API specification. It is listed as a string type but can actually accommodate complex data structures. For more details on its usage, please consult with the i-Apply team.

### Response class.

A screenshot of a computer program

Description automatically generatedIn the realm of RESTful API development, the Response class serves as a crucial component, encapsulating the data and metadata returned from a server in response to an API request. It acts as a container, holding the server's response, including the status code, headers, and the actual data payload.

JSON example:

{

"message": "string",

"success": true,

"extensions": "object",

"description": "int"

…

…

"cbsCustomerInfos": [

{

"addressLine1": "string",

…

…

"zipCode": "string"

}

]

}

The response class is used to return data values to i-Apply from the CCBS API interface. These values can either specify the desired process or data structures, or they can be used to update data within the i-Apply database.

**Fields: success and message:** This field is designated for to curry the value ***false*** if no errorhappened during the processing on the CBS side, or **true** when an error technical or business happened. At this case the field *“message”* can contains a user-friendly message to explain the problem.

"success": true,

"message": null,

…

**Or**

"success": false,

"message": "The interest rate cannot be less than 1%",

…

**Field: extensions:**

This field providing equivalent functionality as the field “additions” to the Requests. It provides a flexible mechanism to include custom data structures that are not explicitly defined in the OpenAPI specification. It can contain any JSON object, regardless of complexity, and is intended to be interpreted as a dynamic type (eg any, dynamic, object) by the CCBS API implementation.

To ensure compatibility and correct processing, the i-Apply team and the Bank's CBS provider must establish a shared understanding of the expected data structure for this field. In the absence of such an agreement, the field should be assigned a null value.

"extensions": null,

Most of the times the shared understanding is reflected in the Business requirement document (BRD).

Here is an example of that field when containing extensions to the standard response data structure:

{

"**message**": null,

"**success**": true,

"**extensions**": [

{

"accdate": "2024-12-02T11:45:51.890Z",

"addres1": "string",

…

…

"website": "string"

}

],

…

"birthcountry": "string",

"crm\_id": "string",

"cu\_branch": 0,

"cu\_comlang": "string",

…

…

"pr\_profe": 0,

"pr\_totemp": "2024-12-02T11:45:51.890Z"

…

…

}

**Important to remember:** The "*extensions*" structure is not explicitly defined in the Open API specification. It is listed as a string type but can actually accommodate complex data structures. For more details on its usage, please consult with the i-Apply team.

## HTTP Headers

In the world of RESTful API development, HTTP headers serve as a crucial mechanism for conveying additional information about an API request or response. They are key-value pairs that provide context, metadata, and instructions to the server and client. The CCBS module foreach individual API Call to the customer’s implementation of the Open API Specification (CCBS) is transmitting i-Apply specific HTTP Headers.

|  |  |  |
| --- | --- | --- |
| Header name | Value | Description |
| IA-UserName | "*Username*" | The name of the user in i-Apply. Its common to be the LDAP (Active Directory etc) name. |
| IA-Role | "*Role*" | The active role, at the time of interface (API) invocation, the user had. |
| <defined1> | “define value-1” | User defined header & value 1. Both name and value are static. Defined at the global parameters. |
| <defined2> | “define value-2” | User defined header & value 2. Both name and value are static. Defined at the global parameters. |
| <defined3> | “define value-3” | User defined header & value 3. Both name and value are static. Defined at the global parameters. |
| <definedX> | “define value-X” | User defined header & value X. Both name and value are static. Defined at the global parameters. |

Note: The defined headers is a mechanism in i-Apply to add any number of headers with static value to cover CBS specific or Security specific needs. An administrator will specify the header name and the header value.

## Example API Call

Typical REST API Call consists of HTTP Headers, URI Parameters, Request Body, and Response.

**HTTP headers** are key-value pairs that provide additional information about an HTTP request or response. They are sent in the initial line of the request or response, followed by a blank line and the message body. Headers can be used to specify authentication credentials, content type, caching instructions, and other metadata. For example, the Authorization header can be used to authenticate a request using a token or basic authentication, while the Content-Type header specifies the format of the request or response body, such as JSON or XML.

**URI parameters** are used to dynamically specify parts of a resource's URI. They are typically enclosed in curly braces and are replaced with actual values when the request is made. For instance, in the URI /users/{userId}, the {userId} part is a URI parameter that can be replaced with specific user ID. URI parameters are often used to filter, sort, or paginate data.

**The request body** is the content of an HTTP request, typically used to send data to the server. The format of the request body depends on the HTTP method and the Content-Type header. For example, a POST request to create a new resource might have a JSON body containing the resource's properties. The request body can also be used to update existing resources or to perform other actions.

**The HTTP response** is the server's reply to an HTTP request. It consists of a status code, headers, and a response body. The status code indicates the outcome of the request, such as 200 (OK), 404 (Not Found), or 500 (Internal Server Error). The headers provide additional information about the response, such as the content type, content length, and caching instructions. The response body contains the actual data returned by the server, which can be in various formats, including JSON, XML, or plain text. i-Apply **strongly** recommended to return always 200 (OK), and the response to contain the successful or not indication (field: success) and the error message (field: message).

This is an example of the information are received and transmitted during an API method call.

### Headers

IA-UserName: "Username"

IA-Role: "Role"

### Request Body

{

  "mockupKey": "string",

"additionals": null,

  "accountNumber": "string",

  "customerCBSCd": 0,

  "customerCBSId": "string",

  "customerType": "string",

  "dateOfBirth": "2024-09-26T12:15:03.861Z",

  "email": "string",

  "employeeNumber": "string",

  "employerNumber": "string",

  "firstName": "string",

  "idNumber": "string",

  "idType": "string",

  "mobileNumber": "string",

  "phoneNumber": "string",

  "pin": "string",

  "surname": "string"

}

### Response Body

{

  "message": "string",

  "success": true,

"extensions": {

"extdate": "2024-12-02T11:45:51.890Z",

"name2": "string"

}

],

  "cbsCustomerInfos": [

    {

      "addressLine1": "string",

      "addressLine2": "string",

      "businessregistry": "string",

      "certificationId": "string",

      "certificationTypeDescription": "string",

      "certificationTypeId": "string",

      "country": "string",

      "customerCity": "string",

      "memberSubflag": "string",

      "phoneNumber": "string",

      "pin": "string",

      "profitsCD": "string",

      "profitsID": "string",

      "zipCode": "string"

    }

  ]

}

## How to start & resources

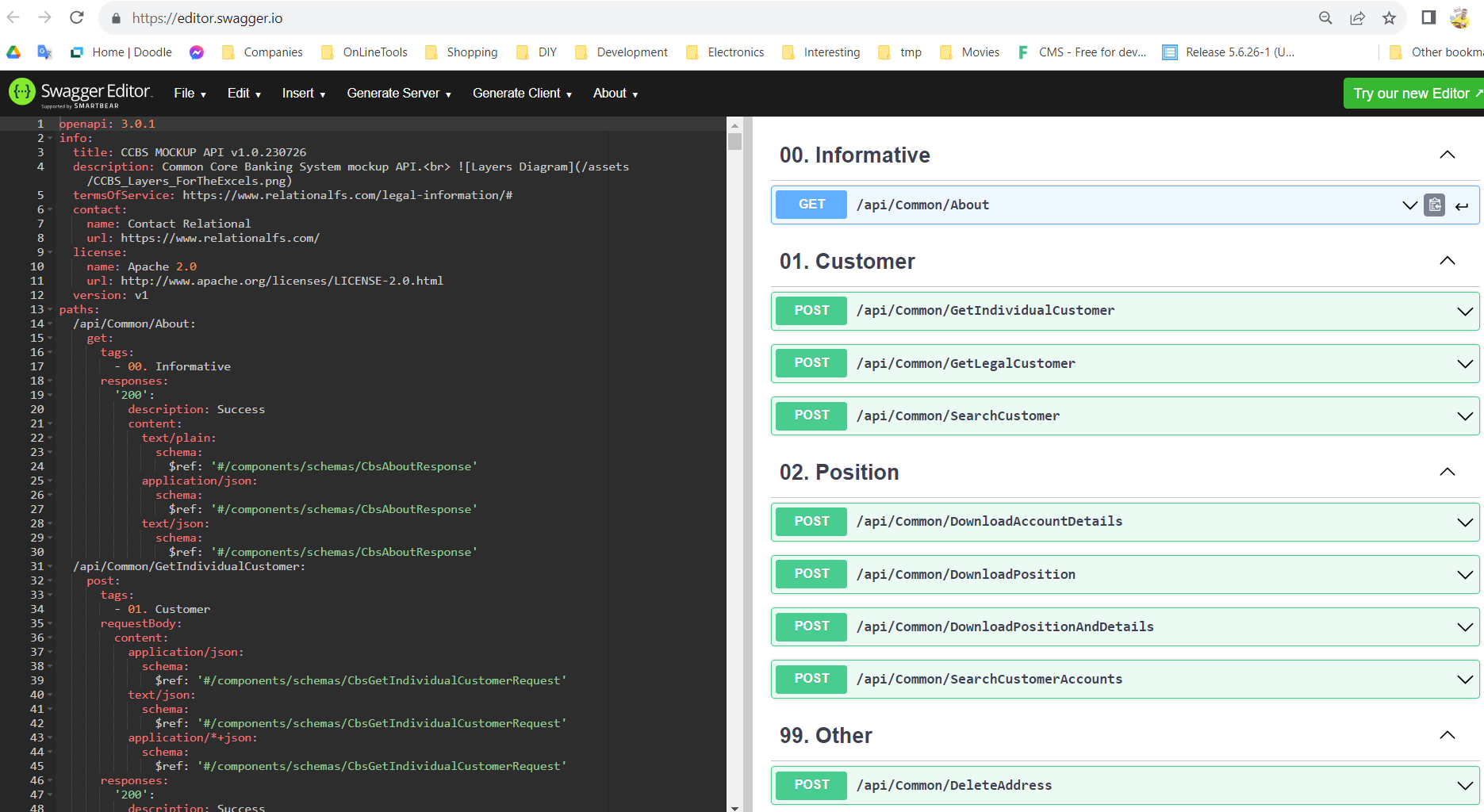
At the GitHub project iApply\_CCBS\_API you will find the OpenAPI JSON file.

Visit link: <https://github.com/aafent/iApply_CCBS_API> and download the file **CCBS\_API.json**

*Notice that we update the repository occasionally, so a git on it is recommended.*

Then we are strongly recommending to use the free service: <https://editor.swagger.io> to import the file or to import the following URL: <https://raw.githubusercontent.com/aafent/iApply_CCBS_API/main/CCBS_API.json>

You will get a picture similar to the following (screen s1):



Screen s1.

As you can see, the API Calls are organized into groups. In that screen shoot, the four (4) groups of the elementary interface are visible.

Actually, there are just 8 (eight) methods to implement for the implementation of the elementary interfaces.

The G00 contains just one method, the "about" The only you must do to return the following response:

**{**

**"title": "CBS Interface,"**

**"apiVersion": "1.0",**

**"code": "ACMEBank.SoftCompany.S1",**

**"message": null,**

**"success": true**

**}**

***Note*** *that in the title field, you can specify whatever you want to name that works, but the code has to be as in the example.*

For Groups G01 and G02, you must implement the described responses based on the given requests. Each method has a specific request and expects either an HTTP Error or a response of the particular structure (class). The responses and the requests are also available in the form of Excel files containing further information.

## Recommended OpenAPI Tools

Open API Tools. An extended list with several tools.

* <https://openapi.tools>

Swagger Tools. The industry standard to the OpenAPI presentation & authoring

* <https://swagger.io/tools/swagger-editor>
* <https://swagger.io/tools/swagger-codegen>

A collection of open-source and commercial tools for creating your APIs with OpenAPI

Sourced from and published for the community.

* <https://tools.openapis.org>

# Further concepts

## Absolute necessary API calls

The absolute necessary API calls to deploy a basic and working i-Apply solution are the API calls that dealing with the Customer inquiry to the CBS and the APIs are used to retrieve the current position of that specific customer, legal entity or individual.

That calls are listed in the document *iApply\_CCBS\_Interfaces.xlsx*, to identify them follow the steps:

1. Rest all the current filters, to show all the available rows.
2. Sort by the sequence listed on row A (green cells). The order is:
   1. First, by the column *Priority*
   2. Second, by the column *API Group*
   3. Then by the column *i-Apply Interface*
3. If you are using the Light version of the i-Apply, on the first column with the label *Color* use the filter and keep only the “Green” value.

The top priorities, between 1 and 2 are indicating the absolute necessary API calls to go at a production without elementary integration with CBS. The i-Apply will retrieve only demographic and position’s information from the CBS.

The next priorities are increasing the level on integration with the CBS.

## Position Entities & flow.

The i-Apply uses the following flow to retrieve the necessary information regarding the position of a customer.



The field names are used to indicate an entity for the position is called (for historical reasons) *“Account Type”* and *“Account Number”* but they are stand for *“Entity Type”* and “Entity Number”. With equivalent logic the fields *“Account Check Digit”* and *“Account Status”* should be considered. For example, the field “Account Number” it stands for the object “Entity Number.”

### Position flow step 1

Is used to retrieve the Customer’s demographic data. The customer can be a) individual or b) legal entity. There are two different API calls one for each type of customer. For both API calls the main request field is the Customer ID in the CBS. The customer ID for the CBS is known to the i-Apply by the API call that is searching customers in the CBS where was invoked before the Position flow.

### Position flow step 2

Is used to retrieve a collection of entities consisting of key values for the customer’s position. Later, within step 3, for each of those values the i-Apply will perform a detailed request to the CBS.

The API calls where is used for the implementation of this step are:

|  |  |
| --- | --- |
| 1 | API Call: **SearchCustomerAccounts**  Document: Exposures\_SearchCustomerAccounts.xlsx |
| Key Fields on Request: Customer CBS Id |
| Key fields on expected Response: A collection of Entity Type, Entity Number,  Product ID, Currency |

|  |  |
| --- | --- |
| 2 | API Call: **DownloadPosition**  Document: Exposures\_DownloadPosition.xlsx |
| Key Fields on Request: Customer CBS Id |
| Key fields on expected Response: A collection of Entity Type, Entity Number, Product ID, Currency |

### Position flow step 3

Is used to retrieve deep details for retrieved entities of step 2. For each item in the response collection of the API call on tables 1 and 2, will be invoked an API call on the table 3 or table 4. API calls 3 and 4 are one alternative to the other. As the API call 4 is contains more extensive information and fields maybe will be more time consumed on resources of the CBS.

|  |  |
| --- | --- |
| 3 | API Call: **DownloadAccountDetails**  Document: Exposures\_DownloadAccountDetails.xlsx |
| Key Fields on Request: Entity Type, Entity Number |
| Key fields on expected Response: a collection of several metrics and informative fields for the specific Entity |

|  |  |
| --- | --- |
| 4 | API Call: **DownloadPositionAndDetails**  Document: Exposures\_DownloadAccountDetails.xlsx |
| Key Fields on Request: Entity Type, Entity Number |
| Key fields on expected Response: a collection of several metrics and informative fields for the specific Entity |

API Calls flow.

