Verification Service - ID HUB

Data wygenerowania: 2025-05-15

Integration Specifications

Document revision - 2.0.3

Definitions

ID HUB (Service, System) - the described tool. An organized set of mechanisms for the Partner to assess the veracity of the Client's data. Verification may include personal data, image compatibility with a document or creditworthiness.

Client - the target user, the person using the service, authenticating one’s data.

Partner - a contractor, an entity integrating with ID HUB (website administrator/technical platform used by the Client)

Paywall - a view in the interface, where a list of available banks/payment channels via which the service can be provided is presented to the Client

Component - a component of the service that provides a verification mechanism within the scope specified for the Partner (e.g. name and surname verification based on bank account details)

Report / Daily report / Monthly report - a document sent to the Partner with data on the number of executed transactions in a specified time

Verification - a technological process consisting in the acceptance by the System of Client's data and providing the feedback specified by a given Component (e.g. photo verification of photographs, bank account details)

Verification result - a document produced by the Verification Component. It may contain a status or additional data. Its content is Component-specific.

ID HUB tool

ID HUB is a tool that allows a Partner to investigate the credibility of its Client or to retrieve Client's personal and Client-related data from external sources (documents, bank accounts, business databases, mObywatel app).

Data verification and acquisition may be performed by different components, each with different characteristics.

The capabilities of the tool are:

1. Account Information Service (AIS) - a mechanism for scanning bank accounts using PSD2 interfaces
2. Verification transfer - execution of a transfer for one zloty and verification of transaction details
3. Photo-verification - OCR of identity document and biometric verification of Client
4. mObywatel - retrieving data from the mObywatel app

ID HUB unifies all components into a single interface and standardizes how to navigate the Client

through the process - regardless of what type of verification/data submission procedure the Partner requires its Clients to undergo.

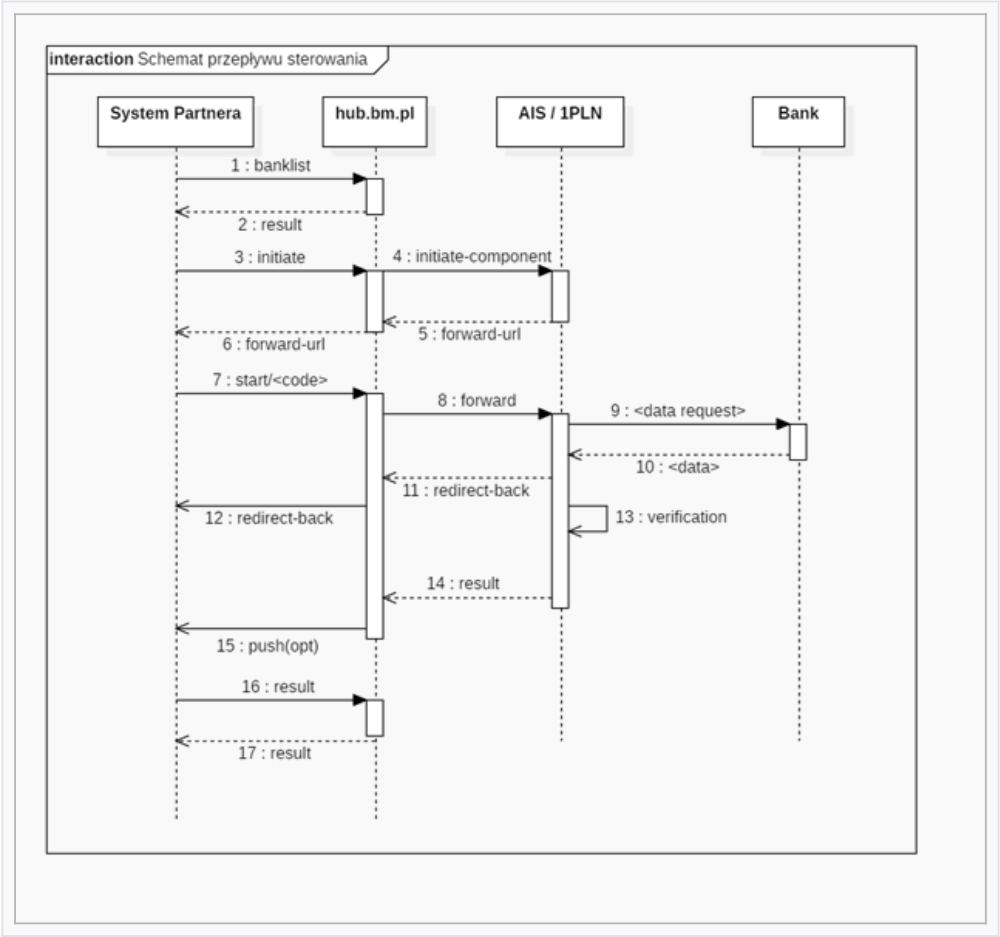
How the ID HUB service works

The Client in the Partner's system performs actions that require verification of its identity. Partner sends to the System a request to create a new Verification. ID HUB, based on the Partner's account configuration, analyzes what kind of Verification should be performed. If the System has a Component, which has the ability to verify the parameters specified by the Partner, it contacts it and builds a unique URL address, to which Client should be redirected.

The URL address is returned to the Partner's system together with the identifier given to the new Verification. Partner's system redirects Client to the received address at the right moment. Client goes to the website in the Autopay domain or directly to the bank and follows the steps set by the selected Verification Component. After completing all the actions requested by the Component, Client is redirected back to the Partner's system. At this point, the verification result should already be available to the Partner's system.

The result of the completed verification is retrieved by Partner by sending a request to the address specified hereinafter. In response, the System will either return the calculated verification result or information that it is under preparation. To the result itself, individual components can attach additional data - so-called reports. Report content is specific to selected components.

It is possible for the System to actively notify the Partner about the prepared verification result. This is called PUSH. It requires the Partner's system to provide a URL, which will be called by the HUB, providing notification of the possibility of downloading the finished result.



Sequence diagram

ID HUB integration procedure

Integration for a new Partner

Integration with the Verification System consists of the following steps:

1. Establish with the Business Supervisor the parameters of system operation. In this step, the Partner specifies his needs, defines the requirements for the ID HUB. The necessary components and the parameters for their use are selected. The result of this step is a completed implementation sheet document.
2. Based on the completed sheet, an account is created in the test environment.
3. The Partner can use the application: https://id-hub-accept.bm.pl/test/start.

TIP: There are many fields on the page, not all of which are required for the test. The values required in the form depend on the needs and order placed in the deployment card and the final configuration determined.

4. Preparation by the Partner of endpoint support (in REST+JSON technology) issued in the System API:

- banklist - downloading a list of available banks with information about the components supporting them (optional step)
- initiate - launching the verification process
- start - redirecting Client to component
- result - downloading the verification result

5. Preparation by the Partner of return addresses, the so-called "landing page" to which the Client should be redirected after leaving the system, where he should wait for the result of his verification.

6. Preparation of an address to handle notifications about a verification result ready to download. [OPTIONAL]

7. Prepare an address to handle change notifications in the list of active banks [OPTIONAL]

Integration for Partner with "Verification Transfer" implemented

Partner may already be integrated with the service provided by Autopay - the Verification Transfer.

Connection to the ID HUB does not preclude the coexistence of the Verification Transfer as a parallel verification mechanism.

If, however, the Partner wishes that the Verification Transfer be covered up by a single interface according to the ID HUB concept and used as a component of a single service, it is unfortunately not

possible to make a hassle-free migration and technical work on the Partner's application will be necessary.

The business configuration (methods of handling refunds, commission settlements, etc.) as well as part of the technological configuration (types of summaries returned by the verification transfer, reports, subsystems ran) will remain unchanged and will be moved to integration in the new model.

The start of verification, Client redirection and the way of receiving the final result will be different.

Environments

Two environments are prepared for the Partner's use: testing and production.

> **NOTE**: When connecting to the Application at the above addresses, we recommend using TLS protocol in version at least 1.2. Support for lower versions will be discontinued soon.

Test environment

- Application: https://id-hub-accept.bm.pl/api
- API Swagger: https://id-hub-accept.bm.pl/swagger/
- Testing application: https://id-hub-accept.bm.pl/test/start

Production environment

- Application: https://id-hub.bm.pl/api
- Test application: https://id-hub.bm.pl/test/start

Integration models

The Partner can integrate with ID HUB using the following models: standard, standard with dedicated frontend, White Label.

Paywall

1. Paywall is available on Autopay end
2. Partner initiates verification
3. Client redirection to Autopay frontend
4. Client carries out the process
5. Client returns to Partner's system

In this model, the look can be personalized in accordance with the Partner’s guidelines. This means that you can add a logotype, change colors, fonts, loaders, etc.

White Label

> **TIP**: The White Label model gives the Partner greater control over the Client's actions and minimizes the time of the Client's stay outside the Partner's domain (reduced to redirections through Autopay websites and staying in the bank).

1. Paywall is available on Partner’s end

2. Partner initiates verification
3. Redirecting Client directly to bank
4. Client carries out the process
5. Client returns to Partner's system

In this manner of integration a new API method appears - BankList. Partner calls this method and in response receives a list of bank accesses configured for him.

Two types of accesses are available: 1PLN and AIS. Within 1PLN, we additionally have a division into the following accesses: PayByLink , PIS (in accordance with PSD2) and Regular Transfer. At the integration stage, Partner is offered the choice of leaving the selection of banks and their respective components to Autopay, or to define a list of banks tailored to exact demand. The downloaded list of banks should be displayed to the Client in order to select the verification channel available to him.

It is Partner's responsibility to collect appropriate consents from the Client. The types of consents and their wording are agreed with the Autopay team before the integration commences.

When the Client chooses the bank in which he holds an account, we proceed to the initiation of verification using the method described in the main part of the documentation, plus the extra bank ID field. The Partner receives the Client's redirection address.

The Client, after getting to the view of his bank, authorizes Autopay systems to access his accounts or execute the transfer. Note, in the case of the AIS component and a history scan longer than 180 days, the Client may be redirected to the Autopay domain to perform additional SCA authorization.

The Client is redirected back, directly to the Partner's website with a temporary "jump" through the Autopay domain.

The report is downloaded in standard way, using the API - Result method.

ID HUB modes of operation

Downloading Client data (DATA HARVEST)

This is a mode that does not perform any verification. It is used to retrieve Client data from external sources. The Partner then has the opportunity to verify the Client's data using its own method.

In download mode, the system redirects the Client to the selected component. There, the Client authorizes the Service's access to his data or enters it himself (Photo Verification).

The HUB takes the data available in the component and transmits it to the Partner in "raw" form or enhanced with additional reports (statistical, categorizing the types of transfers, etc.).

The data download mode is available in every component offered.

Verifications (PERSONAL DATA)

All verifications made by the system consist in comparing the data declared by the Client with the data received from the selected component.

The client data obtained from the components are complicated (in terms of lack of a standardized

structure, unpredictable sequence of elements, variety of addresses, names and surnames); we do not guarantee a 100% correct data classification and consequently 100% correctness of the data comparison result.

To have the success rate of cutting and comparing data as high as possible, we use a number of instruments and algorithms to help with this.

Cutting data

The address data that the System components collect from their sources takes different formats.

Examples of data collected:

```
IZABELA ZIELIŃSKA Warszawska 39/14, 58-400 Kamienna Góra
KOWALSKI MARCIN ul. OSIEK 990, 63-920 OSIEK
WRÓBLEWSKI MARCIN JERZY CEYNOWY 136/15 77-100 BYTÓW
ORGANEK MARTA I ORGANEK WANDA NADWIŚLAŃSKA 82/4 03-349 WARSZAWA
GOSPODARSTWO ROLNE KAMIL MARECZEK BODZIEJOWICE 7B 42-446 IRZĄDZE
JĘDRZEJ NOREK JADWIGA JASKÓŁA-NOREK BRZEŹNICKA 1C32-700 BOCHNIA PL
NIKODEM ARLETA JANA III SOBIESKIEGO 2/6 21-500 BIAŁA PODLASKA
JANUSZ-STOLARCZYK JANINA KOSZARSKO 1 22-335 ŻÓŁKIEW KA
```

The data-cutting tool must recognize in lines like the above (and others) the following components:

1. First name
2. Last name
3. Street
4. House number
5. Staircase number
6. Apartment number
7. Postcode
8. City

The unpredictable and non-deterministic nature of data aggregated from external sources forces the System to resort to special techniques in order to divide lines into specific portions of data as close to ideal as possible.

These techniques include:

1. Use of name and surname dictionaries from the PESEL registry
2. Verifying addresses using address databases and zip codes

The adopted course of action may generate unexpected results (and, as a result, unsuccessful verification processes) for users with address data from outside Poland and first and last names from outside the PESEL database (similarly, when the input data includes the company name).

The Partner may observe unsatisfactory results of the cutting and verification of data when the component used takes data from a bank account that is jointly owned with another person. Such an account makes it impossible to uniquely identify the Client and the user of the System.

Despite making the utmost effort to ensure that the resulting data is correctly cut and classified, one

should be prepared for the possibility that certain verifications are doomed to fail. This is due to the fact that ID HUB has no control over the quality of data received from external sources. This data is sometimes ridden with defects that prevent correct processing (for example, the first name, last name and address from the title of the transfer may be devoid of space characters, forming a cluster of words).

Data comparison

In the process of verifying Client data, a critical and crucial step is to compare the data declared by the Client with the data we obtained about him in the selected Component.

```
The list of fields results from the scope of data that can be obtained from the data of
the verification transfer:
```

- first name
- last name
- street
- house number
- staircase number
- postal code
- bank account number

Parameters of the data comparison mode

1. Co-ownership tolerance

First name and last name are compared on particular terms that make it possible to handle data from shared accounts. A shared account is an account from which we retrieve data of two persons. If the co-owner’s or representative’s data is not included in the transfer sender’s data, the account is considered to be an individual account that belongs to one person. Possible settings: A. Client’s data can be retrieved from the B shared account. Client’s data can be retrieved from the shared account but it must be included in the first place in the data retrieved from the C transfer. Client’s data cannot be retrieved from the shared account.

2. Tolerance of excessive data in the following fields: first name, last name, street.

Possible settings:

a) Mutual tolerance of excessive data

-> Example:

first name from the form	first name from the bank	comparison result
Krystyna	Krystyna Maria	positive
Krystyna Maria	Krystyna	positive

b) Tolerance of excessive data in the form

-> Example:

first name from the form	first name from the bank	comparison result
Krystyna	Krystyna Maria	negative
Krystyna Maria	Krystyna	positive

c) Tolerance of excessive data on the bank account

-> Example:

first name from the form	first name from the bank	comparison result
Krystyna	Krystyna Maria	positive
Krystyna Maria	Krystyna	negative

3. Sensitivity to diacritical marks

By default, diacritical marks matter while comparing data, however, the mode can be configured so that they are ignored.

Personal data

The personal data verification mode consists in the verification of data declared by the Client against the data the System retrieves from, among others, banking systems.

The System verifies personal data using the following Components: AIS, 1PLN, PHOTO.

Data that Components are able to verify

Data	AIS	1PLN	PHOTO	MOBYWATEL
First name	✔	✔	✔	✔
Last name	✔	✔	✔	✔
Company name	✔	✔	✖	✖
Street	✔	✔	✔	✔
House number	✔	✔	✔	✔
Staircase number	✔	✔	✔	✔
Flat/suite/apartment number	✔	✔	✔	✔
Postcode	✔	✔	✔	✔
City	✔	✔	✔	✔
Account number	✔	✔	✖	✖
Identity document number	✖	✖	✔	✔
Place of birth	✖	✖	✔	✔

Data	AIS	1PLN	PHOTO	MOBYWATEL
Identity document expiration dat	✖	✖	✔	✔
PESEL number	✖	✖	✔	✔

Company data

1PLN component

The mode of company data verification in the 1PLN Component consists in verification of the company data declared by the Client with the data that the System collects from banking systems (after successful execution of a PLN transfer) or the Central Statistical Office.

Company data verifiable by the system

Data type	1PLN
Street	✔
House number	✔
Staircase number	✔
Flat/suite/apartment number	✔
Postcode	✔
City	✔
Account number	✔
Company name	✔
REGON number	✔
NIP	✔

Verifying components

AIS verification component

The component referred to as "AIS" is a subsystem of the ID HUB which operates on the basis of the analysis of the Client's bank account history.

The Client, after selecting his bank, either on the list of banks in the Partner's service or Autopay (depending on the mode of integration), is redirected to the login page for his account. There, the Client authenticates and authorizes the System to download the transaction history. The Client returns to the service where it started the process and there waits for the Component to finish retrieving and analyzing the data.

The AIS component, in addition to verifying Client data, has the ability to generate reports and summaries.

Examples of AIS Component report types

Report	Description
Personal data	A report containing the following personal information: 1) First and last name 2) address
Aggregate financial data	A report containing a summary of the activity on the Client's account during a given period. Transactions are grouped according to the following rules: 1) Account number 2) Revenue/expenditure 3) Category of transaction (taxes, purchases, loans etc.) We total the transfer amounts in the cells designated by the above rules. In the report we include the date of the first transfer.
Raw data	A report containing a list of transactions from the Client's account in an agreed format (e.g. CSV, JSON).
Account data	List of bank accounts with detailed description, including: 1) Owner data 2) Bank account number and the type of account
Data with an extended scope (so-called “lambda”)	A report containing a set of various calculated (defined by Partner’s requirements) parameters. These can be i.a. medians/means/numbers/min/max of amounts fulfilling specific requirements (e.g. category type), number of days since a specific event on the account, etc.

Access to reports

The reports generated by the AIS component are not an integral part of the object with the verification result. They are available at the URL that will be included in the verification result. The reason for separating the verification result from related reports is the size of data that the report can generate.

Access to the content of the reports requires authorization by the BasicAuth method. Login and password will be provided to the Partner in the implementation form or at the integration stage.

In the Example of a full report generated by the AIS component there is an example object with selected reports.

In the Example of a category tree for building an aggregate financial report there is an example list-tree of categories that is used in calculating the financial report.It is possible to individually design the category tree according to Partner's needs.

1PLN verification component (Verification Transfer)

The 1PLN Component (Verification Transfer) is a Client verification mechanism in which the user transfers a set amount to Autopay's bank account.

After receiving the transfer, the Component checks if the information about the sender provided by

the bank is consistent with the data received at the moment of initiating the verification. The calculated result is forwarded to the Partner, who takes further steps in the Client's business process.

Configuration personalization is available for the following extensions:

Extension	Description
Predefined transfer title	Each transfer made by the Client will include in its title a fixed description, for example: "Confirmation of contract with XYZ"
Other transfer amount	By default, the user transfers the amount of PLN 1. However, it is possible to configure the component so that the user transfers a different, fixed amount.
Returning the data received in the transfer title	The object with the verification result will be accompanied by a structure with data containing details of the transfer executed by the Client: Returning the data received in the transfer title firstNameFromTransfer - first name lastNameFromTransfer - last name streetFromTransfer - street streetHouseNumberFromTransfer - house number streetFlatNumberFromTransfer - flat number streetStaircaseNumberFromTransfer - staircase number cityFromTransfer - street postCodeFromTransfer - postcode bankAccountNumberFromTransfer - account number unseparatedDataFromTransfer - data before classification companyNameFromTransfer - company name retrieved from CSO database during verification of transfer data (applies to company verification mode)

Methods of executing the transfer

A verification transfer performed at the Autopay Payment Gateway can be made with four methods.

Available payment channels:

Pay By Link - the transfer is carried out through a form generated in the electronic banking of the selected bank. The whole process closes in tens of seconds to a few minutes.

Independent transfer - the transfer is carried out similarly in the Pay By Link method, with the difference that the details for the transfer must be entered manually in their banking, based on the form presented in the Payment Gateway.

PSD2-PIS - a mechanism similar to the Pay By Link method, implemented using PSD2 banking interfaces.

Elixir - in situations where the above payment channels are permanently or temporarily unavailable, the transfer may be carried out using the classic Elixir transfer method. Verification processing time in this case may take up to two business days.

Verifications performed by the 1PLN component are verifications that may take longer. Factors influencing this time are the banking mechanisms that transfer money, along with the procedures that safeguard the process.

In special cases, the transfer processing time may take up to seven days (applies to weeks with consecutive holidays). For this reason, we set the expiration time for verification and waiting for the transfer to 7 days as standard. This parameter may be the cause of notifications about completed verification not arriving in the Partner's system for up to several days.

PHOTO verification component

The PHOTO component is a mechanism in which the System obtains the Client's personal information directly from the Client's identity document (identity document). In addition to acquiring data from the document, the component also allows assessing the compatibility of the Client's biometric parameters at the device he is using with the image found on the presented identity document.

Client verification modes in the PHOTO component

A Client can verify his identity using either a single mobile device or two devices - a desktop computer, where he completes the part of the process responsible for submitting data for verification, and a phone where the process of acquiring and submitting images takes place, after which he returns to the process completed on the computer.

Supported documents

ID card: Polish Passport: Polish, Ukrainian, Belarusian, Georgian

Reports returned in result object for positive verification

Extension	Description
OCR data	The report contains raw text data extracted from the identity document: idDocumentTypeFromOcr: "IDENTITY_CARD" idDocumentIssueCountryFromOcr: "POL" idDocumentIssueStateFromOcr: "" idDocumentExpiryDateFromOcr: "2021-06-30" idDocumentIssuingDateFromOcr: "2011-06-30" idDocumentNumberFromOcr: "ATX012345" peselFromOcr: "74121524371" genderFromOcr: "M streetFromOcr: "GOSPODY" streetHouseNumberFromOcr: "21" streetStircaseNumberFromOcr: "" streetFlaNumberFromOcr: "37" postCodeFromOcr: "80-344" cityFromOcr: "GDAŃSK" fullAddressFromOcr: "80-344 GDAŃSK GOSPODY 21/37" dateOfBirthFromOcr: "1974-12-15" placeOfBirthFromOcr: "ORNETA" firstNameFromOcr: "SZYMON" secondNameFromOcr: "JAN" thirdNameFromOcr: "PAWEŁ" maidenNameFromOcr: "ROGALIK" lastNameFromOcr: "ROGALIK" fullNameFromOcr: "SZYMON JAN PAWEŁ ROGALIK"

Extension	Description
Status of the photo-verification process	These are three indicators: documentStatus - determines the processing status of the identity document bioStatus - determines the processing status of the biometric parameters overallStatus - determines the status of the entire process and compliance of the document with the biometric parameters The possible values that the statuses take are: VERIFIED - positive status NOT_PERFORMED - verification has not taken place at all SUSPICIOUS - verification has been performed and the result suggests a potential fraud attempt In case of problems with positive verification, the result includes “verificationProblems” parameter, where detailed verificators are listed, for example: “expiry_date” means that a document is expired.
Data comparison result	The system compares data selected by the Partner from the document with data shown in the verification initiation in ID HUB. The result may take the following values: POSITIVE - data is consistent NEGATIVE – data is different

The level of data consistency is determined on the basis of the configuration of the comparison mode described in the following part https://developers.autopay.pl/weryfikacje/documentation#data-comparison |

ID HUB compares data received from PHOTO component with data declared by the Client. The data to be verified should be in the input parameters (details are described in the Initiate method).

mObywatel verification component

The "MOBYWATEL" component allows retrieving data from the mObywatel System provided by the Ministry of Digitization.

To use the mObywatel component, it is necessary to cooperate with the Ministry to obtain the appropriate certificate. Submission of the respective application is possible at https://wspolpraca.mobywatel.gov.pl/ The certificate is stored in the Autopay system and is used in the name and on behalf of its owner. The verification process drom the Client's side consists of several simple steps. Clients can submit their data by scanning the QR code in the mObywatel app on their phone. The QR code is returned in response to the initiation of mObywatel verification and displayed on the Partner's website. Once the data is retrieved, it can undergo additional comparison, and after processing, it is sent as a result, analogous to the data from the other components.

Sets of data that can be retrieved from the mObywatel System are defined in the documentation provided by the Ministry od Digitalization, and limited by the permissions granted in the certificate.

Implementation of the service in the Partner's system

The component functions only in the whitelabel model. In order for the Client to use the service, it is necessary to implement the functionality of displaying the data received from the ID HUB system in desktop and mobile versions. Each time, in response to a verification initiation, the ID HUB will return a QR code and an ID consisting of characters that can be copied by the Client. Depending on the Client's device, the Partner should display the appropriate data. The code has a specific expiration date and the default is 60 seconds. After this time, the code expires and there is an opportunity to refresh it. In this case, the Client should be able to generate a new QR code by clicking on the appropriate button, after which the Partner will call the code refresh method.

Programming interface methods

BankList

GET /api/bank/v1.1/list/{partnerUuid}

It is used to download a list of banks associated with a partner. The response contains a list of banks with information which components enable to carry out the subscribed service (verification/retrieval of transaction history) and the current availability status of the bank with the date of the last status change.

BankList - Input parameters in URL

ID	name	type	required	description
n/a	PartnerUuid	uuid	yes	Partner ID

Output parameters

ID	name	type	required	description
5	banks	map<Integer, Bank>	yes	List of banks related to Partner's profile. Map keys are bank identifiers. See table below for Bank object description.
25	status	Assumes the following values: OK, ERROR	no	
30	description	string	no	additional comment related to the action - information message if status=OK, error message if status=ERROR

BankList - Bank object

ID	name	type	required	description
5	name	string	yes	Bank name
10	bic string	no	Bank BIC/SWIFT	code
15	iconUrl	string	no	URL to the bank's logo icon

ID	name	type	required	description
20	additionalConsentsRequired	boolean	no	A flag indicating the need to display additional consents for the client (for the white label model).
25	component	string	yes	The name of the component to be selected in the process. In special cases, the client will be redirected to another component. E.g., in case of sudden unavailability of a pre-selected component during an ongoing process.

BankList - Example of request and response

Request: GET https://id-hub-accept.bm.pl/api/bank/v1.1/list/c455(...)6d89

Response:


```
{
  "status": "OK",
  "description": null,
  "hash": null,
  "banks": {
    "1": {
      "name": "Mbank",
      "bic": "BREXPLPWMBK",
      "iconUrl": "https://platnosci.bm.pl/pomoc/grafika/1800.png",
      "additionalConsentsRequired": true,
      "components": "AIS"
      }
    },
    "24": {
      "name": "Test Mock Bank",
      "bic": "BMMOCKBANK",
      "iconUrl": null,
      "additionalConsentsRequired": true,
      "components": "1PLN"
    }
  }
}
```


Initiate

/api/verification/v1.0/initiate

It is used to initiate the verification process in the system. On the basis of the received input parameters and the project findings, the system selects the appropriate verification Component and prepares the address for redirecting the Client.

Initiating verification in mObywatel component

POST /verification/v1.0/initiate POST /verification/v1.1/initiate/mobywatel The initiate method can be

called using two endpoints. Endpoint version 1.0 requires the component parameter (MOBYWATEL). Endpoint /verification/v1.1/initiate/mobywatel sets this parameter by default. If a component field is additionally completed using endpoint v1.1, it will be ignored.

Initiate - Input parameters

name	type and scope of data	required	description
verificationId	string (~[a-zA-Z0-9-J{1,64}+$)	no	verification identifier given by the Partner
email	string (scope consistent with EmailValidator from Apache Commons ver. 1.6)	no - for 1PLN, PHOTO; yes - for AIS	email address
type	enum: PERSONAL_VERIFICATION and COMPANY_VERIFICATION and DATA_HARVEST	yes	type of verification to be performed in the System
params	map<string, string>	yes	map of input parameters to be verified by a Component of a given type. The list of parameters is defined by the verification type (see "type" field).
PartnerUuid	uuid	yes	Partner's text identifier
bankId	integer (allowed values are keys of the map, returned by the BankList method)	no	id of the selected bank - required in the "White Label" model
component	enum: AIS, 1PLN, PHOTO	no	If there's more components than one, a Partner may choose the component to be used.

Initiate in whitelabel mode in 1 PLN component without bank ID

POST /api/verification/v1.0/initiate-with-iban

There is a possibility to initiate verification in the whitelabel model in the 1 PLN component without providing the bank ID, but instead by providing the customer's IBAN. Based on the 4 or 8 digit of the prefix, the appropriate bank will be assigned, and the system will generate a redirect URL to that bank.

In the case of an unknown prefix, the customer will be redirected to the option 'other bank' ('Mam konto w innym banku'), which leads to a page on the autopay.eu domain, where the data for manual transfer will be provided.

If the account number is to be additionally compared, in PERSONAL_VERIFICATION type, it should be placed in a separate field 'BankAccountNumber' in the 'params' section, in the format of 26 digits, without the PL prefix.

Initiate with iban – Parametry wejściowe

nazwa	typ i zakres danych	wymagany	opis
verificationId	string (^[a-zA-Z0-9-_]{1,64}+$)	nie	identyfikator weryfikacji nadany przez Partnera
email	string (zakres zgodny z EmailValidator z Apache Commons ver. 1.6)	tak – dla 1PLN, nie - dla PHOTO; tak – dla AIS, nie- dla MOBYWATEL	adres email klienta
type	enum: PERSONAL_VERIFICATION i COMPANY_VERIFICATION i DATA_HARVEST	tak	rodzaj weryfikacji do wykonania w Systemie
params	map<string, string>	tak	mapa parametrów wejściowych do weryfikacji przez Komponent danego typu. Lista parametrów jest określona przez typ weryfikacji (patrz pole "type")
partnerUuid	uuid	tak	tekstowy identyfikator Partnera
iban	string (PL + 26cyfrowy numer rachunku bankowego)	tak	IBAN z polskim prefiksem (PL). Prefiksy innych krajów nie są dozwolone.
component	enum: AIS, 1PLN, PHOTO, MOBYWATEL	nie	wybrany przez Partnera rodzaj komponentu do zainicjowania (gdy jest ich więcej niż jeden i Partner nie chce zdać sięna dobór automatyczny)

Initiate - Map key dictionary "params" for the value of the "type" field:PERSONAL_VERIFICATION

Parameter requirements are set individually in the process of creating an account, and depend on the scope of the verification and additional functionality to accompany the process (e.g., securing access to reports in PDF files with a password sent to a phone number).

parameter	allowed values
firstName	^[\p{L}\s]{1,32}+$
lastName	^[\p{L}-'.\s]{1,64}+$
pesel	\d{11}
residenceAddressStreet	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.]{1,64}+$
residenceAddressHouseNumber	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.A]{1,10}+$

parameter	allowed values
residenceAddressStaircaseNumber	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.A]{1,10}+$
residenceAddressFlatNumber	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.A]{1,10}+$
residenceAddressPostalCode	^[09
residenceAddressCity	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.()]{1,64}+$
phoneNumber	^((+
bankAccountNumber	^([0 9]{26})$
idDocumentType	IDENTITY_CARD
idDocumentExpiryDate	Date from the future in the YYYY-MM-DD format

Initiate - Map key dictionary "params" for the value of the "type" field: COMPANY_VERIFICATION

Parameter requirements are set individually in the process of creating an account, and depend on the scope of the verification and additional functionality to accompany the process (e.g., securing access to reports in PDF files with a password sent to a phone number).

parameter	allowed values
companyName	^.{1,150}+$
nip (Tax Identification Number)	^\d{10}$
regon (REGON number)	^(\d{9}|\d{14})$
phoneNumber	^((+
companyAddressStreet	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.]{1,64}+$
companyAddressHouseNumber	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.A]{1,10}+$
companyAddressStaircaseNumber	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.A]{1,10}+$
companyAddressFlatNumber	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.A]{1,10}+$
companyAddressPostalCode	^[09
companyAddressCity	^[A-Za-z0-9ĘęÓóĄąŚśŁłŻżŹźĆćŃń\s-.()]{1,64}+$
bankAccountNumber	^([30 9

Initiate - Map key dictionary "params" for the value of the "type" field: DATA_HARVEST

parameter	allowed values
phoneNumber	^((\+|00)?((?!00)\d{2}))?\d{9}$

The phoneNumber field is required when the Partner does not verify the email addresses of its users.

Initiate - Output parameters

name	type	required	description
redirectUrl	string	yes	The URL to which the Client should be redirected
orderUuid	uuid	yes	Verification identifier given by the System
status	enum	no	Response status. Assumes the following values: OK and ERROR
description	string	no	Additional commentary related to the action. Information message for status=OK, error message for status=ERROR

Initiate - an example of a request and response

Request:

```
{
  "partnerUuid": "cc955e86-…65d2",
  "type": "PERSONAL_VERIFICATION",
  "email": "jan@example.com",
  "params": {
    "firstName": "Jan",
    "lastName": "Niezbędny",
    "residenceAddressStreet": "Ciemna",
    "residenceAddressHouseNumber": "1",
    "residenceAddressPostalCode": "89-999",
    "residenceAddressCity": "Grodkowo",
    "bankAccountNumber": "72249000052663617643733450"
  }
}
```

Response:

```
{
  "status": "OK",
  "description": null,
  "hash": null,
  "redirectUrl": "https://id-hub-accept.bm.pl/api/verification/v1.0/start/U5F9VMY8WV",
  "orderUuid": "2ed3575a-37a6-487a-a993-b753b6e4e607"
}
```

Start

GET /api/verification/v1.0/start/<code>

The method is used to redirect the client to the verification Component in order to continue the identity verification process. The redirection should be done using the http GET method.

The only parameter of the method is the unique identifier given by the System in the initiate method.

The implementation of this method is not obligatory. The Partner's system can rely entirely on the address which appeared in the field "redirectUrl", response to the verification initiation request.

The start method is a one-time method. The client can be redirected to the component only once, after which the code is expired.

Start method for mObywatel component

Start GET /verification/v1.1/start/mobywatel/{code}

POST /verification/v1.0/qr-code/refresh/mobywatel In AIS, PLN 1 AND PHOTO components, the method is used to redirect the Client to the verification Component in order to continue the identity verification process. In the mObywatel component, the method is used to obtain a QR code that should be displayed to the Client on the Partner's website. In addition, a code is provided that can be copied and pasted into the mobile app. This is for the Client path in mobile form, where the QR code should not be displayed. Refreshing an out-of-date code involves not initiating a new verification, but calling the /verification/v1.0/qr-code/refresh/mobywatel method. In response, the ID HUB will return a new code, also with a specific expiration date.

Result

/api/verification/v3.0/result

Result - Input parameters

The method is used to download the verification result. The response is a document with the verification status, information about the Component used and links to reports, if the selected Component had been configured to calculate and provide additional information about the client.

In each verification component, it is possible to opt out of verification. If the client explicitly chooses to do so, or abandons the process for a period of time specified in the configuration, the verification in the System receives ABANDONED status. In this case, the response from endpoint does not contain any additional data about the client and his details.

In DATA_HARVEST mode, the "result" response object field is understood differently. Only the PENDING value is significant here, which indicates that the data harvesting result is not yet ready. When the data is harvested successfully, the "result" field will have POSITIVE value and will mean that the report attached to the response object is downloadable.

Result - Input parameters

name	type	required	description
orderUuid	uuid	yes	Verification identifier
partnerUuid	uuid	yes	Partner identifier

Result - Output parameters

name	type	required	description
result	enum	no	Verification status. Allowed values: ABANDONED - the client has abandoned and not completed the verification process; REJECTED_BY_USER - the client has openly opted out of further verification; POSITIVE - client data verified as consistent; NEGATIVE - client data is inconsistent
verificationId	string	no	Verification identifier given by the Partner
systemsUsed	enum	yes	Component Used. Allowed values: AIS; 1 PLN; PHOTO
resultDetails	map<string, string>	yes	The map of parameters received in the initiate method, in the : format, where "key" matches the keys in the parameter map of the initiate method, and "status" is the enum dictionary field [POSITIVE
data	map<string, object>	yes	Map of objects with a set of data declared in the /initiate method and extracted from the verification component (used to perform verification and additional data transferred without comparison). If no specific fields are defined, all acquired and handled data is returned.
dataComponent	map<string, object>	yes	Object map with set of data obtained from verification component. MOBYWATEL: full set of data obtained from mObywatel System PHOTO: empty object AIS: BANK_ACCOUNT_NUMBER, FULL_ADDRESS and FULL_NAME PLN 1: empty object
addons	map<string, string>	no	Map of additional parameters returned by the verification components. These may be, for example, addresses to the generated reports. Detailed lists of returned parameters can be found in the paragraphs describing the verification components. br/>MOBYWATEL: address for retrieving the full data set in the original structure from the mObywatel System; address of the photo, if included in the retrieved data set PHOTO: verificationProblems, bioStatus, documentStatus, overallStatus AIS: addresses for generated reports PLN 1: unseparatedDataFromTransfer – uncut data of the transfer sender
status	enum	no	Response status. Allowed values: OK - verification is completed PENDING - verification is not yet completed and you should re-query in a few seconds; ERROR - verification error. The cause of the error could have been, for example, a lost connection to the Client's bank

name	type	required	description
description	string	no	Additional commentary related to the action. For status=OK information message. For status=ERROR error message

Example of request and response

Request:

```
{
    "partnerUuid": "cc955e86-f78f-45fd-a6c8-615ae2be65d2",
    "orderUuid": "2bc300b6-ec61-481f-a51f-114f662e9c63"
}
```

Response in AIS component:

```
{
  "status": "OK",
  "description": null,
  "result": "NEGATIVE",
  "verificationId": null,
  "systemsUsed": [
    "AIS"
  ],
  "resultDetails": {
    "firstName": "NEGATIVE",
    "lastName": "NEGATIVE",
    "residenceAddressPostalCode": "NEGATIVE",
    "residenceAddressStreet": "NEGATIVE",
    "residenceAddressHouseNumber": "NEGATIVE",
    "bankAccountNumber": "POSITIVE",
    "residenceAddressCity": "NEGATIVE"
  },
  "data": {
    "provided": {
      "firstName": "Nowak",
      "lastName": "Jan",
      "city": "Gdańsk",
      "street": "Grunwladzka",
      "bankAccountNumber": "(...)",
      "postCode": "80-180",
      "streetHouseNumber": "3"
    },
    "obtained": {
      "city": "warszawa",
      "street": "dobra",
      "bankAccountNumber": [
        "(...)"
      ],
      "postCode": "01-100",
      "individuals": [
        {
          "firstName": "tomek",
          "lastName": "widelec"
        }
      ],
```

```
      "streetHouseNumber": "1"
    }
  },
  "addons": {
    "reportUrl":
"https://id-hub.accept.bm.pl/ais/report/2f9a6e5d-5ac2-4b00-a3c6-cffb9380ae9a"
  }
}
```

Response in 1PLN component:

```
{
  "status": "OK",
  "description": null,
  "result": "POSITIVE",
  "verificationId": null,
  "systemsUsed": [
    "1PLN"
  ],
  "resultDetails": {
    "firstName": "POSITIVE",
    "lastName": "POSITIVE"
  },
  "data": {
    "provided": {
      "lastName": "NOWAK",
      "firstName": "TERESA"
    },
    "obtained": {
      "individuals": [
        {
          "firstName": "teresa",
          "lastName": "nowak"
        }
      ]
    }
  },
  "addons": {
    "streetHouseNumberFromTransfer": "6",
    "firstNameFromTransfer": "teresa",
    "lastNameFromTransfer": "nowak",
    "streetFromTransfer": "długa",
    "bankAccountNumberFromTransfer": "(...)",
    "unseparatedDataFromTransfer": "Iwona Piesiewicz Teresa Nowak Długa 6 80-233
Gdańsk",
    "cityFromTransfer": "gdańsk",
    "postCodeFromTransfer": "80-233"
  }
}
```

Response in PHOTO component:

```
{
  "status": "OK",
  "description": null,
  "result": "NEGATIVE",
  "verificationId": null,
```

```
  "systemsUsed": [
    "PHOTO"
  ],
  "resultDetails": {
    "firstName": "NEGATIVE",
    "lastName": "NEGATIVE",
    "idDocumentExpiryDate": "NEGATIVE",
    "pesel": "NEGATIVE",
    "idDocumentNumber": "NEGATIVE"
  },
  "data": {
    "provided": {
      "firstName": "Janina",
      "lastName": "Miętka",
      "phoneNumber": "123456789",
      "idDocumentExpiryDate": "2030-01-01",
      "pesel": "70060717411",
      "idDocumentNumber": "USZ391914"
    },
    "obtained": {
      "idDocumentExpiryDate": "2021-06-30",
      "pesel": "84031221000",
      "individuals": [
        {
          "firstName": "JAN",
          "lastName": "PAJĄK"
        }
      ],
      "idDocumentNumber": "ATU111111"
    }
  },
  "addons": {
    "verificationProblems": "(...)",
    "genderFromOcr": "M",
    "idDocumentIssuingDateFromOcr": "2011-06-30",
    "bioStatus": "NOT_PERFORMED",
    "mode": "LIVENESS_PASSIVE",
    "idDocumentNumberFromOcr": "ATU111111",
    "idDocumentIssueCountryFromOcr": "POL",
    "streetHouseNumberFromOcr": "15",
    "maidenNameFromOcr": "PAJĄK",
    "placeOfBirthFromOcr": "BYDGOSZCZ",
    "documentStatus": "SUSPICIOUS",
    "idDocumentTypeFromOcr": "IDENTITY_CARD",
    "streetStaircaseNumberFromOcr": "a",
    "dateOfBirthFromOcr": "1984-03-15",
    "cityFromOcr": "gdańsk",
    "fullAddressFromOcr": "80-344 GDAŃSK GOSPODY 15A M.143",
    "streetFromOcr": "gospody",
    "lastNameFromOcr": "PAJĄK",
    "fullNameFromOcr": "JAN PAJĄK",
    "postCodeFromOcr": "80-344",
    "peselFromOcr": "84031221000",
    "firstNameFromOcr": "JAN",
    "streetFlatNumberFromOcr": "143",
    "idDocumentExpiryDateFromOcr": "2021-06-30",
    "overallStatus": "SUSPICIOUS"
  }
}
```

Response in MOBYWATEL component:


```
{
  "status": "OK",
  "description": null,
  "result": "POSITIVE",
  "verificationId": null,
  "systemsUsed": [
    "MOBYWATEL"
  ],
  "resultDetails": {
    "firstName": "POSITIVE",
    "lastName": "POSITIVE"
  },
  "data": {
    "provided": {
      "lastName": "NOWAK",
      "firstName": "TERESA"
    },
    "obtained": {
      "placeOfBirth": "warszawa",
      "idDocumentType": "IDENTITY_CARD",
      "gender": "F",
      "idDocumentIssuingAuthority": "PREZYDENT M. ST. WARSZAWY",
      "idDocumentExpiryDate": "2030-04-15",
      "dateOfBirth": "1985-12-05",
      "pesel": "85120500779",
      "individuals": [
        {
          "firstName": "teresa",
          "lastName": "nowak"
        }
      ]
      "idDocumentNumber": ZOP843564
    }
  },
  "addons": {
    "resultRawDataProviderUrl":
"https://id-hubaccept.bm.pl/api/verification/v1.0/result/raw-dataprovider",
    "resultFaceImageUrl":
"https://id-hubaccept.bm.pl/api/verification/v1.0/result/id-document-faceimage"
},
"dataComponent": {
   "personalDataSet": [
    {
      "key": "LAST_NAME",
      "value": "NOWAK"
      },
      {
      "key": "PLACE_OF_BIRTH",
      "value": "WARSZAWA"
      },
      {
      "key": "GENDER",
      "value": "F"
      },
      {
       "key": "PESEL",
       "value": "85120500779"
      },
      {
```

```
        "key": "FIRST_NAME",
        "value": "TERESA"
        },
        {
        "key": "DATE_OF_BIRTH",
        "value": "1985-12-05"
        }
        ],
        "addressDataSet": [],
        "documentDataSet": [
        {
        "key": "ID_DOCUMENT_NUMBER",
        "value": "ZOP843564"
        },
       {
        "key": "ID_DOCUMENT_FACE_IMAGE",
        "value": "/9j/ +yZnLfTrt7+DSd70QWrRbv8 (…) 6oQnNW1BnFwB1j3bSC+//Z"
        },
        {
        "key": "ID_DOCUMENT_ISSUING_AUTHORITY",
        "value": "PREZYDENT M. ST. WARSZAWY"
        },
        {
        "key": "ID_DOCUMENT_EXPIRY_DATE",
        "value": "2030-04-15"
        },
        {
        "key": "ID_DOCUMENT_TYPE",
        "value": "IDENTITY_CARD"
        }
  ]
  }
  }
```


Health-Check

GET /api/monitoring/health-check

The method is designed to test the network accessibility of the API. The answer to the demand should be the word: OK and http response code: 200. Any other code and any other answer means that there are problems with the operation of ID-HUB.

BankList-notification (PUSH)

The system offers a mechanism for notifying changes in the list of banks. This is effected through notification/push to a specified address.

The address to which the system is to send notifications should be provided by the party integrating with the HUB.

The system sends an empty request (POST) and expects an empty http response with code 204.

By default, the HUB sends one notification and does not try to repeat it even if it has not received the http 204 response code. Repeating is an option that can be enabled on Partner's request.

Result-notification (PUSH)

The HUB can send a notification to the Partner's system, indicating that it is possible to download a ready verification result.

In order to use this functionality, which eliminates the need for cyclic queries about the verification result, the Partner must prepare an endpoint, which will handle the http POST request.

name	type	required	description
orderUuid	uuid	yes	verification identifier
partnerUuid	uuid	yes	partner ID

At the Partner’s request, the PUSH notification can be secured using the HMAC method.

Result-notification (PUSH) - Answer

The response to such a request should be an empty response, with an http code of 200. Once it is received, the System considers the notification as delivered. Otherwise, it repeats the request with decreasing frequency until it receives the expected response.

NOTE: The list of fields notified to the Partner can be extended. Please include in the implementation the possibility of smooth appearance of new fields in the notification object.

Decreasing frequency means repeats in subsequent iterations, where each subsequent attempt takes place after a longer interval than the previous one. The intervals are calculated according to the Fibonacci sequence, as shown in the table below:

Attempt repetition	Interval in minutes since previous attempt
1	1
2	2
3	3
4	5
5	8
6	13
...	t = (t-1) + (t-2)

Client's return from the component to Partner's system

The Client completing the verification process at the HUB is redirected to the Partner's system to the specified return address.

There are two return addresses:

1. Success address - client is redirected after a correctly completed process. Note - negative verification is still treated as a correctly completed process.
2. Failure address - client is redirected in case of a failure situation or system error. The verification was not successful.

The return to partner's system is made to the addresses exactly as specified in the configuration in the implementation sheet.

Enriching the URL with a verification identifier

It is possible to enrich the URL with one of the following dynamic parameters:

parameter	description
orderUuid	Verification identifier
verificationId	Verification identifier assigned by Partner during the verification initiation phase

NOTE: We add only one of the above parameters to the return address, we do not combine them.

Changing verification result manually

In certain situations, the verification result may be disputable. Initiation data crossed with data retrieved from sources may give results that are difficult to be assessed not only for an IT system but also for a human being. Exemplary situations can be as follows: attempts to interpret first names, last names and addresses of clients from outside Poland, co-ownership of bank accounts, erroneous or non-standard address notations introduced into source systems.

The verification result determined by data that has been processed not in line with the Partner’s expectations can be changed to the expected one by a Partner’s operating employee.

The functionality allowing you to change the verification results manually requires activating. The activation occurs after reporting the need to the Autopay support team.

When requesting the activation of the manual verification modification functionality, you should provide a list of persons who will perform these operations. It is suggested that you provide a list of logins used by these users in the system. If you fail to provide the logins, they will be generated. In reply, you will receive a confirmation of the functionality activation and login passwords for the user accounts requested. Once provided to the ordering person, the passwords will not be stored in transparent form and if the passwords are lost, a new password will have to be generated.

Result method response payload with changing verification result functionality enabled

```
{
  "status": "OK",
  (...)
  "addons": {
    "resultChangeUrl": "https://id-hub.bm.pl/v/:uuid1/:uuid2"
   }
}
```

There is a field "resultChangeUrl" in "addons". It contains a unique URL assigned to verification.

When wrong verification result occurs, you have to use the URL from the field above, login to the site and change verification result to expected one.

The change of result may be performed only once and only from POSITIVE to NEGATIVE (and opposite).

After the change, new result notification is sent to the partner's notification URL.

Result method response payload after manual result change

```
{
  "status": "OK",
  (...)
  "addons": {
    "resultSetManually": true,
    "resultSetManuallyAt": "2022-10-10 12:34:22",
    "resultSetManuallyBy": "jkowalski"
   }
}
```

In "addons" object there are new fields: "resultSetManually", "resultSetManuallyAt", "resultSetManuallyBy".

They carry information that the change was performed, when and by who.

Confirmations of verifications performed

The system offers the functionality of providing confirmations of completed 1PLN and AIS verifications.

The confirmation of the verification performed is a digitally signed .pdf document, which is sent by email to the email address set by the Partner (the address handled by the Partner; not the Client's address).

Generation of AIS confirmation is performed automatically, up to 24 hours after verification is completed, after reports are correctly downloaded using the /result method.

The 1 PLN verification confirmation is generated after the order is placed in the Scribe panel. An order can be placed at any time after verification.

Developer recommendations

By implementing transport objects into the System API, we suggest configuring the JSON serialization and deserialization tool in such a way that it tolerates the appearance of new fields or their absence. The development of verification components may result in the appearance of new reports or objects providing detailed client data. The basic answer to such a development is API versioning (visible in URLs), but we allow (without wanting to fragment the interface) for minor modifications without

versioning entire paths.

Security

Network

We provide network security to our Partners by providing the service via HTTPS. Each endpoint or redirection will present itself with a network certificate issued for the domain: *.bm.pl. Apart from handling traffic via HTTPS, we have a default IP address filtering setting for requests incoming to the System’s API.

If these two methods of network security are not sufficient for the Partner, we allow to create a dedicated network tunnel, via which the Partner's application will be able to communicate with the System services. Establishing such tunnel requires separate arrangements and process, because it is not a solution available in the basic configuration.

It is also possible to control traffic between Partner’s systems and the HUB through the so-called "two-way SSL" mechanism. In this integration model, the Partner's system presents itself with a key signed by Autopay.

The implementation of this method of security, similarly to tunnel establishment, requires separate arrangements between the Partner and Autopay.

Personal data and reports

ID-HUB running in AIS mode generates data reports that it sends to the Partner's system and to the verifying user. These reports may contain personal information such as name, address, transaction history from a bank account, etc. Critical to information security is the issue of delivering this information to the correct email address.

At the stage of integration with ID-HUB, the Partner should specify whether the email addresses it will provide at the time of initiation of verification are those confirmed by the user.

If so, ID-HUB will send email messages to the indicated addresses with a link to the site and a password for the reports.

If the Partner is not sure whether the addresses provided by the users are correct, the ID- HUB administrator shall set the appropriate mode in the configuration. As a result of setting it up, ID-HUB will require a phone number during verification initiation. Once the report is generated, the system will send an email to the unverified address and an SMS to the indicated number. In the body of the email will be a link to a website where the reports may be downloaded. The site will be secured with a password, the content of which the user will receive in an SMS.

When a user opens a view where he finds links to reports with his data, he will be able to download them for 20 minutes. After this time, the links will become inactive until the page is reloaded and password authorization made.

Authentication

In addition to the security features in the transport layer, the System attempts to secure

communication also in terms of the integrity of the transmitted data. For this purpose, a method of verifying the correctness of transmitted messages with the conventional name "HMAC" has been prepared.

It involves calculating a checksum from the request body using one of the supported functions:

- HmacSHA256,
- HmacSHA512

The checksum calculating process involves a secret, Partner-specific key, which is passed on together with other integration parameters.

The Partner who in the integration process chooses the HMAC authentication method, is obliged to include two headers in each request performed by a method other than the GET method:

- Hmac-Algorithm - containing the name of the function used to generate the checksum (the value should be included in the above-mentioned list of supported functions)
- Hmac - containing the calculated checksum value

We recommend this way of authorizing requests for the following reasons:

1. Transport objects do not need to be enriched with a field that does not carry business content
2. The order of fields in the transport objects is irrelevant
3. The security data of the request is not part of the request content, so it is slightly more difficult to eavesdrop/view

A missing or incorrect value of the Hmac-Algorithm header results in response code 400.

A missing or incorrect value of the Hmac header results in response code 401.

Example of how to count checksum (JAVA)

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.util.Base64;

class Authenticator {

    public static String calculateSignature(String algorithm, String secretKey, byte[]
requestPayload) {
        Mac mac = Mac.getInstance(algorithm);
        mac.init(new SecretKeySpec(secretKey.getBytes(), algorithm));
        mac.update(requestPayload);
        byte[] signature =  mac.doFinal();

        return Base64.getEncoder().encodeToString(signature);
    }
}
```

Example of how to count checksum (PHP)

```
<?php
    $code = hash_hmac('sha256', $requestPayload, $secretKey, true);
```

```
    echo base64_encode($code);
?>
```

Example of how to count checksum (PHP)

```
using System;
using System.Security.Cryptography;
using System.Text;
public class Program
{
        public static void Main()
        {
                string secretKey = "mdk7G86HJVjhgUGh865434dsr5";
        string payload = "{\n" +
            "  \"partnerUuid\": \"11a122bb-a111-a22a-eeee-22a222a2a2a2\",\n" +
            "  \"email\": \"test@autopay.pl\",\n" +
            "  \"type\": \"PERSONAL_VERIFICATION\",\n" +
            "  \"params\": {\n" +
            "    \"firstName\": \"Jan\",\n" +
            "    \"lastName\": \"Kowalski\",\n" +
            "    \"residenceAddressStreet\": \"Powstańców Warszawy\",\n" +
            "    \"residenceAddressHouseNumber\": \"6\",\n" +
            "    \"residenceAddressStaircaseNumber\": \"A\",\n" +
            "    \"residenceAddressFlatNumber\": \"1\",\n" +
            "    \"residenceAddressPostalCode\": \"81-718\",\n" +
            "    \"residenceAddressCity\": \"Sopot\"\n" +
            "  },\n" +
            "  \"bankId\": \"25\",\n" +
            "  \"verificationId\": \"test\"\n" +
            "}";

        string hmac = CalculateHmacSha512(secretKey, payload);
        Console.WriteLine("HMAC: " + hmac);
        }

    public static string CalculateHmacSha512(string key, string data)
    {
        using (var hmacsha512 = new HMACSHA512(Encoding.UTF8.GetBytes(key)))
        {
            byte[] hashValue = hmacsha512.ComputeHash(Encoding.UTF8.GetBytes(data));
            return Convert.ToBase64String(hashValue);
        }
    }
}
```

TIP: You can find examples in other programming languages at Joe Kampschmidt's blog.

NOTE: When implementing HMAC authorization, pay attention to invisible newline characters, which may lead to different results of calculated signatures on client and server end. The safest way to do this is to make sure that the serialized object being sent is free of these characters at all.

BasicAuth

Authentication of http requests by means of BasicAuth mechanism is used in selected elements of the system, which the System wants to protect against accidental downloading of data by unauthorized users.

In the current version of the documentation and the system, the report download addresses in the AIS component have been secured using the BasicAuth method.

The login and password needed to build the header "Authorization" is given to the Partner at the moment of starting integration with the System, in the implementation sheet.

NONE

In the initial integration phase, in the test environment, we have the option to disable the data integrity control mechanism. We recommend using this configuration only in the test environment, in the initial integration phase.