Use cases

This chapter contains the description of the processes taking place in the IBAN Management Service from the point of view of the Customer and the end user. Information on how to integrate with the solution has also been added.

IBAN transfer receiving

This functionality allows user to receive transfers. User which is using Customer's application is able to share his IBAN to potential payer. This type of transfer is a transaction that top ups user's balance. As receiving an incoming transfer requires only to provide IBAN to his potential payer, the only functionality which the Customer must handle is IBAN displaying. For integration details please visit Technical documentation chapter.

Display IBAN

The user who logs in the Customer application sees his balance and the IBAN that is assigned to this balance. This allows him to share such IBAN to a potential payer so that the payer can make a transfer to the user's account. Having a visible IBAN, the user can also top up his balance by making a bank transfer from his external account. The IBAN can be displayed by using the `getIban` and `getIbanMobile` methods depending on the implementation model used by the Customer. The implementation model also defines the method of Customer authorization in the IMS service. In the case of the Mobile SDK implementation model, the Customer authenticates using a session token received from Mobile DC. In the case of the REST API implementation model, the Customer authenticates through signed x509 certificate.

Mobile SDK integration

The Mobile SDK implementation model was created for Customers who want to share the IBAN Management Service solution through their mobile application. In order for the Customer to be able to provide its users with access to their IBAN's, it is necessary to make a request to the IMS service. The functionality that returns the user's IBANs is secured - Customer authorization is required. This process consists in transferring a Mobile DC session token during the request to get the list of IBANs. To integrate with the getIbanMobile method, please visit the Technical documentation chapter.

@startuml
skinparam ParticipantPadding 30

```
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #002060
skinparam noteBorderColor #002060
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #002060
ArrowFontColor #002060
ActorBorderColor #002060
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #002060
ParticipantBackgroundColor #002060
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #002060
LifeLineBorderColor #002060
}
participant "User" as user
participant "Customer App" as issuer
participant "Mobile DC" as mdc
participant "IBAN Management Service" as ims
note right of user: User opens Customer mobile app
user->issuer: Check my IBAN
issuer->mdc: Get session token
issuer<-mdc: Return session token
issuer->ims: Get IBAN by balance ID
issuer-->ims: Provide Mobile DC session token
ims->ims: Validate session token
issuer<-ims: Return user's IBAN
user<-issuer: Display IBAN to the user
@enduml
```

REST API integration

The REST API implementation model was created for Customers who want to connect with the IBAN Management Service through their backend. In order for the Customer to be able to provide its users with access to their IBANs, it is necessary to make a request to the IMS service. The access to the IMS is secured and requires the Customer to authenticate with x509 certificate. After successful authentication it is possible to get the list of IBANs. To integrate with the getIban method, please visit the Technical documentation chapter.

```
@startuml
skinparam ParticipantPadding 30
```

```
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #002060
skinparam noteBorderColor #002060
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #002060
ArrowFontColor #002060
ActorBorderColor #002060
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #002060
ParticipantBackgroundColor #002060
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #002060
LifeLineBorderColor #002060
}
participant "User" as user
participant "Customer App" as issuer
participant "IBAN Management Service" as ims
note right of user: User opens Customer web app
user->issuer: Check my IBAN
issuer->ims: Get IBAN by balance ID
issuer-->ims: Provide x509 certificate
ims->ims: Validate certificate signature
issuer<-ims: Return user's IBAN
user<-issuer: Display IBAN to the user
@enduml
```

IBAN transfer sending

This functionality allows user to order outgoing transfers. User which is using Customer's application is able to sends money on a given IBAN. The outgoing transfer is a transaction that charges user's balance.

> Implementation work in progress...

Admin Panel

One of the additional functionalities that Verestro provides is the creation of an administration panel that allows the Customer to track the history of users' transactions - individual banking operations such as debiting or topping up the account with the transfer IBAN are visible. The Admin Panel also provides information such what IBAN is assigned to the secified user and and which balance this IBAN refers to. The Admin Panel is configured mainly on the Verestro side. Each instance of the Admin Panel is issued per Customer.

The below diagram shows the processes of filtering users and their IBAN numbers:

> Process available for Admin Panel operetor with every type of Admin Panel operator roles.

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #002060
skinparam noteBorderColor #002060
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #002060
ArrowFontColor #002060
ActorBorderColor #002060
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #002060
ParticipantBackgroundColor #002060
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #002060
LifeLineBorderColor #002060
}
participant "AP Operator" as issuer
participant "Admin Panel" as ap
participant "IBAN Management Service" as ims
participant "Transaction History Core" as thc
note right of issuer: Customer logs in Admin Panel
issuer->ap: Log in
issuer<-ap: Success
note right of issuer: Customer filter IBANs list
issuer->ap: Get IBANs
ap->ims: Get IBANs by specified query
ap<-ims: Return list of found IBANs
issuer<-ap: Show list of found IBANs
note right of issuer: Customer filter transaction history list
issuer->ap: Get transaction history
```

ap->thc: Get transaction history by specified query
ap<-thc: Return list of found transactions
issuer<-ap: Show list of found transactions
@enduml

The below diagram shows the processes of the uploading transaction file and the execution of the IBAN transfers:

> Process available for Admin Panel operetor with Admin Panel `ADMIN` role.

@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #002060
skinparam noteBorderColor #002060
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #002060
ArrowFontColor #002060
ActorBorderColor #002060
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #002060
ParticipantBackgroundColor #002060
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #002060
LifeLineBorderColor #002060
}
participant "AP Operator" as issuer
participant "Admin Panel" as ap
participant "IBAN Management Service" as ims
participant "Antaca" as ant
participant "Transaction History Core" as thc
note right of issuer: AP Operator uploads transaction file
issuer->ap: Upload transaction file
ap->ims: Send transaction file
note left of ims: The authorization token representing a given Customer in PA is validated
ims->ims: Validate Admin Panel authorization token
ims->ims: Validate transaction file
ap<-ims: Validation success
issuer<-ap: Show validation status
issuer->ap: Execute transaction file
ap->ims: Order transaction file execution

ims->ant: Credit/debit balances based on IBANs from the transaction file
note left of ant: The x509 certificate of a given Customer context is validated
ant->ant: Validate CN
ims<-ant: Credit/debit result
ant->thc: Report transaction
ims->ims: Generated transaction report
ims->thc: Report transaction
@enduml

Views

This chapter contains views and actions to be performed available to the Admin Panel operator regarding the IBAN Management solution. Depending on the role of the operator's account, various actions are available in the Admin Panel. The roles available in the Admin Panel are `EMPLOYEE`, `MANAGER` and `ADMIN`. More information about roles and their permissions are included further in Views chapter subsections.

> Depending on the Customer's requirements, the permissions in the Admin Panel can be configurable. For example, it is possible that only `ADMIN` and `MANAGER` can add a new balance from the Admin Panel. `EMPLOYEE`, on the other hand, would only be able to perform `read` actions. Detailed information about roles and permissions configurations are available in the Admin Panel Service documentation.

Enduser details

Each role is entitled to `read` actions. This means that `EMPLOYEE`, `MANAGER` and `ADMIN` can filter their endusers and check their account details such as assigned balances and IBANs. The option of viewing end users is available in the `CUSTOMERS` / `End Users` tab and has been presented on the picture below:

image-1677147920306.png

Using to the available filters in `Endusers` tab, the Admin Panel operator regardless of his Admin Panel role can find a proper user by various filtering criteria - including the IBAN number. The entered filtering criteria can be cleared using the `Clear all` button in the upper right corner. Available filtering options have been presented on the picture below:

image-1676554736454.png

Such the Admin Panel operator is also able to view the details of a given end user by selecting end user's record from the list of all end users. In the end user details section, there are several tabs dedicated to various functionalities - including `Balances` tab with balances details and the same IBANs assigned to these balances. By selecting the `Reveal` option, the Admin Panel operator can view the end user's IBAN number. From the same view, it is also possible to add a new balance to

the end user, which will also generate a new IBAN number. Generating a new balance is possible with the `+Add new balance` button in the lower left corner. The action of generating a new balance is also available for each role of the Admin Panel operator. The described posibilities available in `Balances` tab have been presented on the picture below:

image-1676555543719.png

Payment history

Each of the Admin Panel operators also has the ability to view the transaction history of all endusers. The option of viewing end users is available in the `CUSTOMERS` / `Payment History` tab and has been presented one the picture below:

image-1677150640402.png

Using to the available filters in `Payment history` tab, the Admin Panel operator regardless of his Admin Panel role can find a proper end user's transaction by various filtering criteria. The entered filtering criteria can be cleared using the `Clear all` button in the upper right corner. After specifying the filtering criteria, the operator can also generate a transaction report using the `Generate transaction report` button in the upper right corner.

image-1677151269746.png

Such the Admin Panel operator is also able to view the payment history of a given end user by selecting end user's record from the list of all end users. In the end user details section, there are several tabs dedicated to various functionalities - including `Payment history` tab with transaction details performed by selected enduser. The `Payment history` tab have been presented on the picture below:

image-1677149382343.png

Upload transaction file

> The following section describes actions allowed only for Admin Panel operators with `ADMIN` role.

This chapter is devoted to views and processes related to uploading transaction files and their execution. These processes can only be performed by operators with ADMIN role rights and persons legally authorized to perform such operations as ordering and executing transactions. The functionality responsible for uploading transaction files is available in the `IBAN MANAGEMENT` / `Upload transaction file` tab and has been presented on the picture below:

image-1677152467676.png

After selecting the `Upload transaction file` tab, a dropbox intended for uploading transaction file

appears. The Admin Panel operator with the `ADMIN` role has permission to upload such file and confirms the upload with the `Submit` button visible in the lower right corner. The dropbox view has been presented on the picture below:

image-1677157756676.png

After confirming the upload of the transaction file by the `Submit` button, the validation process begins. During the validation every transaction record contained in the file is checked. The validation result is displayed in the list of uploaded files under the transaction file dropbox.

> Each file should have a unique name otherwise there is a risk of overwriting a previously uploaded file with the same name.

After the validation process, information about whether the file is valid or not valid is displayed. The information about the transaction file validation status is presented in the form of a status in the `Status` column. The list of validation statuses and their meanings are available in the Transaction file validation statuses section. If the file validation was successful, then it is possible to execute transaction file. File execution is performed after clicking the `Execute` button in the `Send file` column. Each transaction file can be sent for execution only once to avoid accidental additionally top up or debit of the given balance. If the validation result of the transaction file is negative, then a proper status in the `Status` column is presented and the validation report file is generated, available for download by the Admin Panel operator. The Admin Panel operator can download such validation report by clicking on the `arrow button` in the `Report link` column. Appearance of the list of uploaded transaction file has been presented on the picture below:

image-1677157946455.png

Transaction file validation statuses

Status	Description
Validated	Transaction file upload completed successfully. Validation returned positive status. Ready for execution.
Executed	Transaction file already executed.
Executing	Transaction file execution is progress.
Pending	Transaction file validation in progress. Execution is not possible.
Invalid	Transaction file validation returned nagative status. Execution is not possible.

Execute files reports

> The following section describes actions allowed only for Admin Panel operators with `ADMIN` role.

This chapter is devoted to views and process related to downloading executed transaction reports. These processes can only be performed by operators with ADMIN role rights and persons legally authorized to perform such operations as ordering and executing transactions. The functionality responsible for uploading transaction files is available in the `IBAN MANAGEMENT` / `Upload transaction file` tab and has been presented on the picture below:

image-1677161299855.png

After selecting the `Executed files reports` tab, a list of the all transaction reports appears. List contains information about each of the generated reports: date of the report generation, name of the report and it's identifier. There is also posibility to download such report by clicking on the `arrow button` in the `Report link` column. The list of the executed transaction reports view has been presented on the picture below:

image-1677235485592.png

Notification after transaction

One of the additional functionalities offered in the IMS solution is sending transaction notifications to the Customer. The notification is sent in the form of an event that should be handled by the Customer. The Customer should create `POST` type endpoint to allow us to send such an event.

```
POST https://example-customer-site.com/transaction-event


{
  "id": 20,
  "timestamp": 1665557034
}
```

Transaction event contains the transaction identifier as has been shown in example above. Using this identifier, the Customer can get the details of a given transaction using the appropriate method. It is also possible to get a list of all transactions. Implementation details are available in the technical documentation of the Transaction History Core service.

@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #002060
skinparam noteBorderColor #002060
skinparam noteBorderThickness 1
skinparam sequence {

```
ArrowColor #002060
ArrowFontColor #002060
ActorBorderColor #002060
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #002060
ParticipantBackgroundColor #002060
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #002060
LifeLineBorderColor #002060
}
participant "Transaction History Core" as THC
participant "Customer's Server" as issuer
THC->issuer: Send notification event
issuer->THC: Get transaction details by received transaction id
issuer<-THC: Return transaction details
@enduml
```

> This is an optional feature, but we recommend using it.

Revision #2
Created 21 March 2023 10:54:12
Updated 27 March 2023 08:12:40