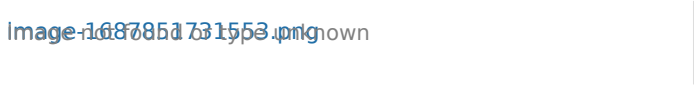
Use cases

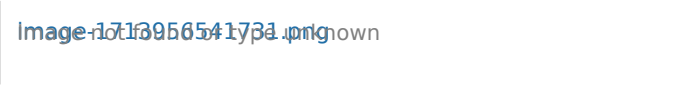
This chapter is mainly dedicated to the appearance of the application and a description of the individual paths that the end user will follow depending on the payment method he choosed. Each of the payment method available in Paytool application is described in a separate section. Additionally, every chapter contains diagrams depicting the entire transaction process per payment method.

> **Tip:** Check it now. Verestro Paytool demo application is available here.

Transaction initialization

The first step of the each transaction always takes place in your application. Your application integrated with the Verestro Paytool solution should have exposed option allowing to pay using Paytool available in your application. Your backend should deliver the initial transaction data to the Paytool API to open new transaction session when your payer selects the Paytool as payment option. Our API will return you the transaction identifier and the payer's redirection address. The view of transaction initiatialization from the end user's perspective has been shown below. This view refers to the Redirect your payer integration path:





> **Tip:** We are fully responsible for the execution of the transaction and threeDs authentication process. You only initialize transaction by redirecting the end user to the Paytool payment form and providing basic transaction metadata via the your backend.

The sequence diagram below illustrates the transaction initialization process step by step

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
```

```
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
actor payer
participant "Customer front" as cfront
participant "Customer backend" as cback
participant "Paytool front" as pfront
participant "Paytool backend" as pback
payer->cfront: Pay with Paytool
cfront->cback: Payer choosed Paytool
cback->pback: Customer authorization + transaction metadata
pback->pback: Store transaction metadata + open payment session
cback<-pback: OK + transaction id + redirect url
cfront<-cback: OK + transaction id + redirect url
cfront->pfront: Redirect + transaction id
pfront->pback: Get transaction data + merchant payment methods
pfront<-pback: OK response
payer<-pfront: Display transaction + payment methods
@enduml
```

Server to server

An alternative path to carry out a transaction is a Payment process via API integration path. Paytool Frontend does not appear in this path. You make all requests directly from your server to the Paytool API methods we described in technical integration guide.

The sequence diagram below illustrates the transaction initialization process step by step

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
actor payer
participant "Customer front" as cfront
participant "Customer backend" as cback
participant "Paytool backend" as pback
payer->cfront: Pay with Paytool
cfront->cback: Payer choosed Paytool
cback->pback: Customer authorization + transaction metadata
pback->pback: Store transaction metadata + open payment session
cback<-pback: OK + transaction id
note right of cfront: At this point, the Customer decides how to present the transaction to the payer
cback->pback: Request proper payment method / executing 3D Secure
cback<-pback: Transaction result
note right of cfront: At this point, the Customer decides how to present the transaction result to the payer
@enduml
```

Payment methods

Verestro as a PSP provides several different payment methods in Paytool solution. You decide which payment method should be available in Paytool payment form. You decide this during the onboarding process but it is editable whenever necessary. The end user, seeing the available

payment methods, can choose the option that is most convenient for him. This chapter describes all payment methods supported by our application.

Payment methods in Paytool
Google Pay™
Apple Pay
Blik
Plain card number

> **Note:** Each of the payment methods available from the Paytool frontend is also available with an API to API connection.

> **Note:** It is required that you have created account in the Acquirer's system which will settle your transactions. Verestro Paytool solution has been implemented so that it is possible to process transactions with the participation of various Acquirers. If you require the settlement of the transaction by a new Acquirer – to which we are not yet integrated – there will be required new integration between Verestro and the new Acquirer. You should provide a specification of the Acquirer API.

Plain card number

The most common payment method available in our application is payment using plain card details. This payment method consists in providing a payment form in which the end user can enter his payment card details. The option to pay with plain card details is available at the button shown below:

image-1685428055821.png
Image not found or type unknown

After clicking on the payment option using plain card details, the end user is redirected to the specially prepared payment form. Now he is able to provide the appropriate card details and clicks the **Pay** button at the bottom of the screen. At this point, the card details are encrypted and sent to the Paytool backend which requests Acquirer to process transaction. At this moment threeDs authentication process may be required. The payment process using plain card details from the end user's perpective has been shown below:

image-1713956617464.png
Image not found or type unknown

The sequence diagram below illustrates the transaction with plain card details process step by step

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
actor "End User" as user
participant "Paytool Frontend" as pfront
participant "Paytool Backend" as pback
participant "Customer Server" as custback
participant "Acquirer" as acq
user->pfront: User chooses payment by plain card details
user<-pfront: Redirect end user to the payment form
user->pfront: User filling data (CN, CVC, EXP)
pfront->pfront: Encrypt data
pfront->pback: Perform transaction providing encrypted payment request body
pback->pback: Decrypt encrypted payment request body
pback->acq: Order transaction
note left of acq: At this point 3D Secure process may occur
acq->pback: Response
pback->pfront: Transaction Result
pback-->pback: Store transaction result
pfront->pfront: Redirect end user (success/failure)
pback->custback: Send transaction postback to provided URL (optional - configurable)
note right of pback: Verestro can also send the notification after transaction to the end
user(optional)
custback->pback: Get transaction status (optional)
pback->custback: Return transaction status
@enduml
```

Google Pay

Paytool as a registered hosted checkout in the Google Pay™ allows to perform a payment with the card from the Google Pay™ Wallet. Using our application your payer can choose the Google Pay™ as the payment method from the Verestro Paytool payment methods list. The option to pay with Google Pay™ card token is available at the button shown below:

image-1685428002241.png
Image not found or type unknown

After clicking on the Google Pay™ payment option, the Google Pay™ Wallet popup is displayed. The payer should select the appropriate card from the Google Pay™ Wallet which is then encrypted on the Google Pay™ side and transferred to our application. We will decrypt obtained data and transfer them to the Acquirer in the transaction request. At this moment threeDs authentication process may be required but it will be handled on our side. The payment process using Google Pay™ card payment token from the end user's perpective has been shown below:

image-1689246683126.png
Image not found or type unknown

The sequence diagram below illustrates the payment process using Google Pay™ card payment token step by step

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
actor "End User" as user
participant "Paytool Frontend" as pfront
participant "Paytool Backend" as pback
```

```
participant "Google Pay" as gpay
participant "Customer Server" as custback
participant "Acquirer" as acq
user->pfront: End user chooses payment by Google Pay
pfront->gpay: Open end user Google Wallet
user<--gpay: Opens end user's Google Wallet
user-->gpay: Choose card from Google Wallet
gpay->gpay: Encrypt chosen card token data
pfront<-gpay: Provide encrypted card token data
pfront->pback: Order transaction
pfront-->pback: Provide encrypted card token data
pback->pback: Decrypt encrypted card token data
pback->acq: Order transaction
note left of acq: At this point 3D Secure process may occur
acq->pback: Response
pback->pfront: Transaction Result
pback-->pback: Store transaction result
pfront->pfront: Redirect end user (success/failure)
pback->custback: Send transaction postback to provided URL (optional - configurable)
note right of pback: Verestro can also send the notification after transaction to the end user (optional)
custback->pback: Get transaction status (optional)
pback->custback: Return transaction status
@enduml
```

Google Pay payment via API

Alternatively Verestro as a registered PSP in the Google Pay™ allows to perform a payment with card token generated by Google. This payment option provides a backend to backend oriented solution. In this payment model you must be registered merchant in Google Pay™ to be able to get a card payment token from Google Pay™.

To use this option your API should be integrated with the `googlePayTransaction` method provided by Paytool API. By using this solution you need to display Google Pay™ Wallet in your application by your own. After selecting the Google Pay™ as the payment option a payer chooses proper card from Google Pay™ Wallet. After that action you get encrypted card payment token from Google Pay™. Card payment token is encrypted on the Google Pay™ side with Verestro's public key. You have to provide this encrypted token to Paytool API by using `googlePayTransaction` method. If you do this, we will handle transaction process on our side. At this moment 3D Secure process may be required and you need to handle it on your side using threeDs authentication methods provided by Paytool API.

The sequence diagram below illustrates the payment process using Google Pay™ card payment token API to API step by step

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
participant "End user" as user
participant "Customer Application" as app
participant "Google Pay" as gp
participant "Paytool Backend" as tps
participant "Acquirer" as acq
note right of user: User wants to pay with Google Pay
user->app: Pay with Google Pay and choose card
app->gp: Requests for card token
gp->gp: Encrypts card token with Verestro pub key
app<-gp: Returns encrypted card token
app->tps: Requests token payment
tps->tps: Decrypts card token
tps->acq: Orders transaction
tps<-acq: Transaction status
note left of acq: At this point 3D Secure process may occur
app<-tps: Transaction status
user<-app: Transaction status
user<--tps: Sends email notification - optional
@enduml
```

> **Important:** If you use Paytool with Payment process via API integration path, make sure you are a registered merchant in the Google Pay™ system. This does not apply to Customers using the Redirect your payer route.

> **Note:** Google Pay™ provides a Google Pay Web integration checklist that will help you with the integration step by step. The documentation is available after whitelisting in Google Pay™ system. The whitelisting process is performed by Google Pay™ during the Customer's merchant account registration process.

> **Note:** Google Pay™ provides Google Pay Web Brand Guidelines that presents branding requirements for web merchants registered in Google Pay™. You must meet these requirements so that you can allow his payers to pay via the Google Pay™ solution.

Apple Pay

Verestro as a PSP registered in the Apple Pay™ allows to perform a payment with card token generated by Apple. Using our application your payer can choose the Apple Pay™ as the payment method from the Paytool payment methods list. The option to pay with Apple Pay™ card token is available at the button shown below:

image-1699524383207.png
Image not found or type unknown

> **Warning!** Apple Pay payment method is available in Safari web browser only.

After clicking on the Apple Pay™ payment option, the Apple Pay™ Wallet popup is displayed. End user should select the appropriate card from the Apple Pay™ Wallet popup which is then encrypted on the Apple Pay™ side and transferred to our application. We decrypt obtained data and transfer them to the Acquirer in the transaction request. At this moment threeDs authentication process may be required. The payment process using Apple Pay™ card payment token from the end user's perpective has been shown below:

image-1699526152020.18.03.png
Image not found or type unknown

The sequence diagram below illustrates the payment process using Apple Pay™ card payment token step by step

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
```

```
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
actor "End User" as user
participant "Paytool Frontend" as pfront
participant "Paytool Backend" as pback
participant "Apple Pay" as gpay
participant "Customer Server" as custback
participant "Acquirer" as acq
user->pfront: End user chooses payment by Apple Pay
pfront->gpay: Open end user's Apple Pay Wallet
user<--gpay: Opens end user's Apple Pay Wallet
user-->gpay: Choose card from Apple Pay Wallet
note left of user: At this point Apple Pay may require user authorization
gpay->gpay: Encrypt chosen card token data
pfront<-gpay: Provide encrypted card token data
pfront->pback: Order transaction
pfront-->pback: Provide encrypted card token data
pback->pback: Decrypt encrypted card token data
pback->acq: Order transaction
note left of acq: At this point 3D Secure process may occur
acq->pback: Response
pback->pfront: Transaction Result
pback-->pback: Store transaction result
pfront->pfront: Redirect end user (success/failure)
pback->custback: Send transaction postback to provided URL (optional - configurable)
note right of pback: Verestro can also send the notification after transaction to the end user
(optional)
custback->pback: Get transaction status (optional)
pback->custback: Return transaction status
@enduml
```

> **Note:** At this moment Apple Pay biometric authorization may be required.

> **Note:** Payment using the Apple Pay Wallet is only possible if the payer uses MacOS.

Blik

> **Warning!** Implemenation is work in progress...

Paytool supports cardless payments using the Blik code. The end user can choose the Blik as the payment method from the Verestro Paytool payment methods list. The option to pay with Blik code is available at the button shown below:

image-1685523665104.png Image not found or type unknown After selecting this payment method, the end user must provide his e-mail address in the proper field in displayed popup and then confirm the willingness to make the transaction. Blik as an external service, in response opens a view that allows end user to enter the 6 digits code. This step of the process takes place outside the Verestro. The 6 digits Blik code is generated in the end user's bank application. The payment process using Blik code from the end user's perpective has been shown below:

image-1687851945379.png Image not found or type unknown

image-1687852047098.png Image not found or type unknown

The sequence diagram below illustrates the payment process using Blik step by step

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
```

```
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
actor "End User" as user
participant "Paytool Frontend" as pfront
participant "Paytool Backend" as pback
participant "Acquirer" as acq
participant "Blik" as blik
participant "Customer Server" as custback
user->pfront: End user chooses payment by Blik (1)
pfront->pback: Requests for payment by Blik (2)
pback->acq: Orders Blik transaction (3)
acq->blik: Internal request to Blik system (4)
acq<-blik: Response from Blik system with HTML payment Base64 encoded form (5)
pback<-acq: Provides Blik's response (5.1)
pfront<-pback: Provides Blik's response (5.2)
pfront->pfront: Decode Blik HTML form (6)
user<-pfront: Displays Blik HTML form (7)
user->user: Enters Blik's 6 digits code (8)
user-->blik: After form submitting request to Blik system is triggered (9)
pfront<--blik: Redirects to Paytool Frontend (10)
user<--pfront: Shows progress bar because the user has to wait for transaction status (11)
acq<--blik: Webhook with information about Blik's transaction status (12)
pback<--acq: Webhook with information about Blik's transaction status (12.1)
pfront->pback: Get status of the Blik transaction (13)
pfront<-pback: Provides Blik transaction status (14)
user<-pfront: Displays transaction status (15)
custback<--pback: Send transaction postback to provided URL (16) (optional - configurable)
@enduml
```

ThreeDs authentication

Transactions using payment cards usually require additional authentication of the end user (card holder). In accordance with the Europenian Union directive PSD2, the Paytool solution uses the threeDs protocol version 2.0 for the authentication process. The threeDs process consists in asking the card issuer whether additional authentication of the card holder is necessary or not. It depends on the bank's response whether the threeDs process will be necessary to perform. This is determined by one of two statuses that the bank returns during the transaction process: `FRICTIONLESS` or `CHALLENGE`. The meaning of each status is described in the section ThreeDs modes. We take responsibility for performing the threeDs authentication process. The sequence

diagrams below illustrates the threeDs authentication process step by step separatly for both cases: `FRICTIONLESS` or `CHALLENGE`.

ThreeDs modes

ThreeDs mode	Description
FRICTIONLESS	The response containing the `FRICTIONLESS` mode denotes that payment is successfully finnished. Additional 3DS authentication is not required.
CHALLENGE	The response containing this mode denotes that the card holder must be additionally authenticated. Along with the `CHALLENGE` mode, the Bank also returns an encoded HTML template, which is displayed to the end user in the Verestro Paytool Frontend, thus allowing the end user to perform 3DS authentication.

FRICTIONLESS

In this threeDs mode there is not any additional action from the end user side required.

The sequence diagram below illustrates the threeDs process for frictionless mode step by step

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
LifeLineBorderColor #1C1E3F
}
```

```
actor "End User" as user
participant "Paytool Frontend" as pfront
participant "Paytool Backend" as pback
participant "Acquirer" as acq
participant "Bank" as bank
user->pfront: End user confirmed transaction (1)
note right of user: For example the end user choosed payment with plain card number (1.1)
pfront->pback: Provide transaction request to perform (2)
pback->acq: Provide transaction request to perform (3)
acq->bank: contact with end user's bank (4)
bank->acq: Return FRICTIONLESS status - transaction succeed (5)
acq->pback: Provide bank's response (6)
pback->pfront: Provide bank's response (6.1)
pfront->user: Display transaction status (7)
@enduml
```

CHALLENGE

In this threeDs mode the end user authentication is required. Authentication takes place on the threeDs bank template displayed by the Verestro Paytool application to the end user during the transaction process.

The sequence diagram below illustrates the threeDs process for challenge mode step by step

```
@startuml
skinparam ParticipantPadding 30
skinparam BoxPadding 30
skinparam noteFontColor #FFFFFF
skinparam noteBackgroundColor #1C1E3F
skinparam noteBorderColor #1C1E3F
skinparam noteBorderThickness 1
skinparam sequence {
ArrowColor #1C1E3F
ArrowFontColor #1C1E3F
ActorBorderColor #1C1E3F
ActorBackgroundColor #FFFFFF
ActorFontStyle bold
ParticipantBorderColor #1C1E3F
ParticipantBackgroundColor #1C1E3F
ParticipantFontColor #FFFFFF
ParticipantFontStyle bold
LifeLineBackgroundColor #1C1E3F
```

```
LifeLineBorderColor #1C1E3F
}
actor "End User" as user
participant "Paytool Frontend" as pfront
participant "Paytool Backend" as pback
participant "Acquirer" as acq
participant "Bank" as bank
user->pfront: End user confirmed transaction (1)
note right of user: For example the end user choosed payment with plain card number (1.1)
pfront->pback: Provide transaction request to perform (2)
pback->acq: Provide transaction request to perform (3)
acq->bank: contact with end user's bank (4)
bank->acq: Return CHALLENGE status along with bank HTML template (5)
acq->pback: Provide bank's response (6)
pback->pfront: Provide bank's response (6.1)
pfront->pfront: Decode bank HTML template (7)
pfront->user: Display bank content to the end user (8)
user-->bank: Perform 3DS authentication (9)
user<--bank: Authentication succeed (10)
acq<-bank: Provide success status and authentication ID (11)
pback<-acq: Provide success status and authentication ID (11.1)
pfront<-pback: Provide success status and authentication ID (11.2)
pfront->pback: Finalize transaction and provide authentication ID (12)
pback->acq:  Finalize transaction and provide authentication ID (12.1)
acq->bank: Finalize transaction (12.2)
bank->acq: Transaction succeed (13)
acq->pback: Provide transaction status (14)
pback->pfront: Provide transaction status (15)
pfront->user: Display transaction status (16)
@enduml
```

Transaction result

Depending on whether the transaction was successful or not, the end user will see the appropriate status on the screen. From the status screen, he is able to return to the store's website manually - by clicking the **Return to the shop** button or wait for an automatic redirection. At the same time, we will send you post-transaction notification containing the status and details of the transaction. Depending on the configuration of your merchant account, an e-mail notification may be sent to the end user. Transaction status views from the end user's perspective are presented below:

Image-1684330553924.png
Image not found or type unknown

Image-1684330621724.png
Image not found or type unknown

Notification after transaction

As mentioned in the Transaction result chapter, Paytool solution supports the sending of post-transaction notifications informing you about the status of a given transaction made in your context. This functionality is optional and it is up to you to decide whether you want to receive post-transaction notifications. This decision is made during the onboarding process and always can be edited. We recommend using this functionality, as this allow you to receive information on each of the transactions made using Verestro Paytool.

> **Important:** In order for you to receive notification after transaction, you must define your own endpoint to which notification will be directed. This is necessary so that we know where to send an HTTP POST request with transaction details. During the onboarding process, the you can provide a valid and accessible URL address that will serve as the endpoint for the post-transaction notifications.

Post-transaction notification example sent by Verestro Paytool to the Customer


```
{
 "transactionId": "string",
 "orderNumber": "9afds032dad",
 "transactionState": "FAILED",
 "amount": 0,
 "message": "string",
 "currency": "string",
 "externalTransactionId": "string",
 "transactionType": "ONE_TIME_PAYMENT",
 "cardDetails": {
  "cardProvider": "MASTERCARD",
  "cardNumber": "512485******4875"
 },
 "failedDetails": {
  "code": "SYSTEM_ERROR",
  "message": "E0200: Transaction rejected, card is blocked"
 }
}
```


End user notifications

As mentioned in the Transaction result chapter, Paytool solution supports an e-mail notification sending. Such notification contains the status and details of the transaction. This message is purely informative so that the payer knows that a transaction has been made in the context of his chosen payment instrument. This functionality is optional and it is up to you to decide whether you us to send end users notifications after the transaction. This decision is made during the onboarding process and always can be edited.

> **Note:** The appearance of the e-mail notification is configurable.

Example e-mail notification after successful transaction	Example e-mail notification after failed transaction
Image not found or type unknown	Image not found or type unknown

Revision #196
Created 1 March 2023 09:23:17 by Jakub Kotyński
Updated 26 April 2024 05:29:53 by Jakub Kotyński