Overview

This document provides a description of functionalities offered by Verestro Paytool. Our solution supports various payment methods such as Google Pay™, Apple Pay, Blik and Debit/Credit Card in the form of the payment gateway. In addition, you can decide which payment methods should be enabled. Simply put, you may decide that, for example, you want Verestro Paytool to provide payment via Google Pay but payment via Debit/Credit Card should be disabled. In such a situation, the end user will see the Google Pay as the availabe payment method in the Verestro Paytool payment form, but the Debit/Credit Card payment method will not appear at all.

Transaction process mainly takes place on Verestro's side. This means that you are completely relieved of responsibility for processing the transaction and/or performing 3D Secure authentication. The only action that the you must do is to provide metadata of the transaction, which includes order number, description, amount, currency and some optional parameters.

> **Important!** Note that if you require the settlement of the transaction by an Acquirer to which Verestro is not integrated there will be required new integration between Verestro and the new Acquirer. You should provide the specification of the new Acquirer which will allow us to perform integration.

Abbreviations

In this chapter there are abbreviations and acronyms used in the document listed in below table.

Abbreviation	Description
ACQ	Acquiring Institution / Acquirer
ACS	Access Control Server
SDK	Software Development Kit
PSP	Payment Service Provider
OS	Operative System
Mid	Merchant Identifier
PCI DSS	Payment Card Industry Data Security Standard
PAN	Permanent Account Number
CVC	Card Verification Code

3DS	3-D Secure
DCC	Dynamic currency conversion

Terminology

This section explains a meaning of key terms and concepts used in the document.

Name	Description
Customer/Merchant	Institution which uses Verestro products. This institution decides which payment method should be available in the solution and how transaction should be processed.
End user/payer	The entity which uses Paytool solution to pay for ordered good from Customer. It is root of entity tree. End user is an owner of the wallet/card and he decides to pay for the purchase using Paytool solution, selecting it from the list of payment methods available in the Customer application.
Payment service provider	The entity which provides a payment services for external Customers who do not have direct integration with acquirers or are not PSI DSS compilent. From the perspective of he Paytool application, Verestro is the PSP.
Card	Card belongs to the user. If user intends to pay with the Paytool solution using plain card payment method, then has to insert required card's data to the appropriate fields shared by the Paytool solution payment form. Card data will not be stored in the Verestro system. They will be provided to Acquirer.
Card payment token	It is a numerical value in the form of a PAN number. It shows a given card from Google Pay or Apple Pay wallet. The card payment token replaces the card number and is delivered by Google Pay/Apple Pay to Verestro if the end user selects one of the two above mentioned payment options. Verestro passes this value to Acquirer for the payment to be made.
Authorization Method	The way of the authentication of the Google Pay™ card transaction. Verestro supports followed authorization methods: `PAN_ONLY` and `CRYPTOGRAM_3DS` if Customer's country belongs to the European Union. Authorization method is always provided in the Google Pay™ encrypted payload as `authMethod` parameter.

Gateway Id	Phrase/value that identifies a given Payment Service Provider in the Google Pay™ system. The Merchant provides gateway Id to Google Pay™ to obtain a card payment token. By provided gateway Id, Google Pay™ encrypts the card payment token with the appropriate public key. Verestro is defined by a gateway Id with the value `verestro` in Google Pay™ server.
Gateway Merchant Id	Unique Customer identifier assigned by Verestro during the onboarding process. This identifier is in the form of a `UUID`. Verestro understands and uses this to verify that the message was for the Customer that made the request. Customer passes it to Google Pay™. More information about the Gateway Merchant Id can be found in Google Pay™ documentation.
Acquirer	Institution that settles payments. Paytool communicates with the Acquirer to order the transaction and authenticate the card holder using the 3D Secure protocol.
MID	Merchant identifier. This entity is represending Customer in the Acquirer's system. Customer has to provide the mid information to enable mid configuration in the Verestro system. Required to process transactions and 3DS process via Verestro system.
Bank/Issuer	Card issuing institution. In the case of an e-commerce transaction, this entity is responsible for checking whether the cardholder's balance has the appropriate amount of funds to perform a given transaction, determining whether 3D secure authentication is necessary or simply checking whether the card is active.
Cardholder	This is the end user who pays for his purchases using one of the available payment options in Verestro Paytool.
PAN	It is 7-16 digits of the credit/debit card number. These digits contain the Permanent Account Number assigned by the bank to uniquely identify the account holder. It is necessary to provide it when end user wants to pay with a card for purchases via Verestro Paytool solution.
CVC	Card Verification Code. It is a type of security code protecting against fraud in remote payments. CVC is necessary to provide it when end user wants to pay with a card for purchases via Paytool solution.
Expiration date	It is a date of the card validity ending and contains two values – month/year - for example 01/28. Card will be valid to the last day of the month of the year showed on it. It is necessary to provide it when end user wants to pay with a card for purchases via Verestro Paytool solution.

3DS	3-D Secure is a method of authorization of transaction made without the physical use of a card, used by payment organization. The 3DS process in the Verestro Paytool solution is performed internally in the Verestro system which means the Customer is not responsible for end user authentication.
PCI DSS	It is a security standard used in environments where the data of payment cardholders is processed. The standard covers meticulous data processing control and protection of users against violations.
Dynamic currency conversion	Dynamic currency conversion is a process whereby the amount of a credit card transaction is converted to the currency of the card's country of issue.
Tokenization	A security measure where sensitive payment data, like credit card numbers, is replaced with a unique string of characters called a "token".
Card on file	Verestro PCI DSS cards storage. It allows for storing the card data of a given payer and using it in subsequent transactions without having to re-enter this data.

Implementation models

Integration with Paytool should be performed by API call. To initiate a payment, you must request an `transaction initialization` method, which in response opens a payment session with a unique identifier. Once you have an active payment session and its identifier, you can choose one of three available payment processing methods described in Redirect your payer (Web integration), Payment process via API (API integration) and Embed - Payment in iframe chapters.

> **Note:** The `X-509 certificate` data will be signed during the onboarding process.

Redirect your payer

The first integration path is to redirect your payer to the payment web view. This implementation model is more comprehensive because when redirecting the payer, you only need to provide us with transaction metadata and the payment session identifier. We are responsible for the rest of the payment process. Do not forget to initialize the payment session for the transaction before redirecting the payer to the Paytool web view. To initialize payment session use the `transaction initialization` method.

> **Tip:** We highly recommend using Redirect your payer integration model because most of the processes and responsibilities are handled on our side and the integration process is very smooth.

Example request body of the transaction metadata provided by Customer


```
{
  "transactionId": "42be3d06-4577-4a9f-b525-2cfaba244557",
  "currencyCode": "PLN",
  "amount": 100,
  "description": "Test transaction",
  "formLanguage": "en",
  "redirectUrl": {
    "successUrl": "https://paytool.verestro.com/demo/?success=1",
    "failureUrl": "https://paytool.verestro.com/demo/?success=0"
  },
  "sender": {
    "firstName": "Yoshimoto",
    "lastName": "Imagawa",
    "address": {
      "countryCode": "PL",
      "city": "Kyoto",
      "postalCode": "12-345",
      "street": "Ichijo",
      "houseNumber": "1"
    }
  },
  "merchantUrl": "https://paytool.verestro.com/demo/",
  "orderNumber": "1"
}
```


Sequence diagram of the payer redirection process

```
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
note left of 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
```

Payment process via API

The second integration path allowing to continue the payment process is to integrate your application directly to Paytool API. As the first step of this approach you should request `transaction initialization` method to create a payment session in our system. In return you will receive a `transactionId` parameter which can be used in the `pay` method and the rest of the Paytool API methods in the context of the current transaction. In the `pay` method you determine which of the available payment methods you want to use. The payment methods available for this integration

path are listed in the Payment methods section.

> **Info:** In this model we do not provide a frontend view.

Example request body of the transaction metadata provided by Customer


```
{
  "transactionId": "42be3d06-4577-4a9f-b525-2cfaba244557",
  "currencyCode": "PLN",
  "amount": 100,
  "description": "Test transaction",
  "formLanguage": "en",
  "redirectUrl": {
    "successUrl": "https://paytool.verestro.com/demo/?success=1",
    "failureUrl": "https://paytool.verestro.com/demo/?success=0"
  },
  "sender": {
    "firstName": "Yoshimoto",
    "lastName": "Imagawa",
    "address": {
      "countryCode": "PL",
      "city": "Kyoto",
      "postalCode": "12-345",
      "street": "Ichijo",
      "houseNumber": "1"
    }
  },
  "merchantUrl": "https://paytool.verestro.com/demo/",
  "orderNumber": "1"
}
```


Sequence diagram of the payment via API process

```
@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: Payer wants to pay
cfront->cback: Order payment
cback->pback: Customer authorization + transaction metadata
note left of pback: Store transaction metadata + open payment session
cback<-pback: OK + transaction id
cback-->cfront
note right of cfront: At this point, the Customer decides how to present the transaction to the payer
cfront-->cback
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
cfront<--cback
@enduml
```

Embed - Payment in iframe

> **Note:** Unlike the other integration models, the `transaction initialization` method in this flow is executed **only after** the payer's data and payment instrument details have been successfully collected.

The last of the supported integration paths involves embedding Paytool directly into your application, which requires the use of a dedicated SDK. The process begins with the backend requesting a list of the user's saved cards via the `paymentMethods` endpoint. Depending on the response and the availability of the "Card on File" service, the initialized SDK displays either a list of available cards or a form for new card entry within a secure iframe. Once the user confirms their selection by clicking the "Pay" button, the application retrieves the data from the iframe using the `getFormState` method. Based on this information, the backend creates a payment session via the `transactionInitialization` endpoint to obtain a transaction identifier (`transactionId`). Finally, this identifier is passed to the SDK's `pay` method to authorize and finalize the payment process. The payment methods available for this integration path are listed in the Payment methods section.

[Simplified sequence diagram from tech docs]\

> **Note:** If you intend to use Paytool iframe, ensure that strict domain whitelisting (e.g., Content Security Policy) is disabled in your application. This is necessary because the cardholder authentication process often requires loading HTML templates directly from the card-issuing bank. Since each bank operates on a unique domain, restrictive whitelisting may block these resources, preventing the user from completing the transaction.

> **Important!** It is crucial to understand that in this integration model, the "**Pay**" button **must be a component of your application's UI**. The Paytool iframe does not render its own submission button. Your application is fully responsible for controlling the payment flow by listening for user events (like the "**Pay**" button click) and invoking the corresponding Paytool SDK methods at each step.

Example request body of the transaction metadata provided by Customer

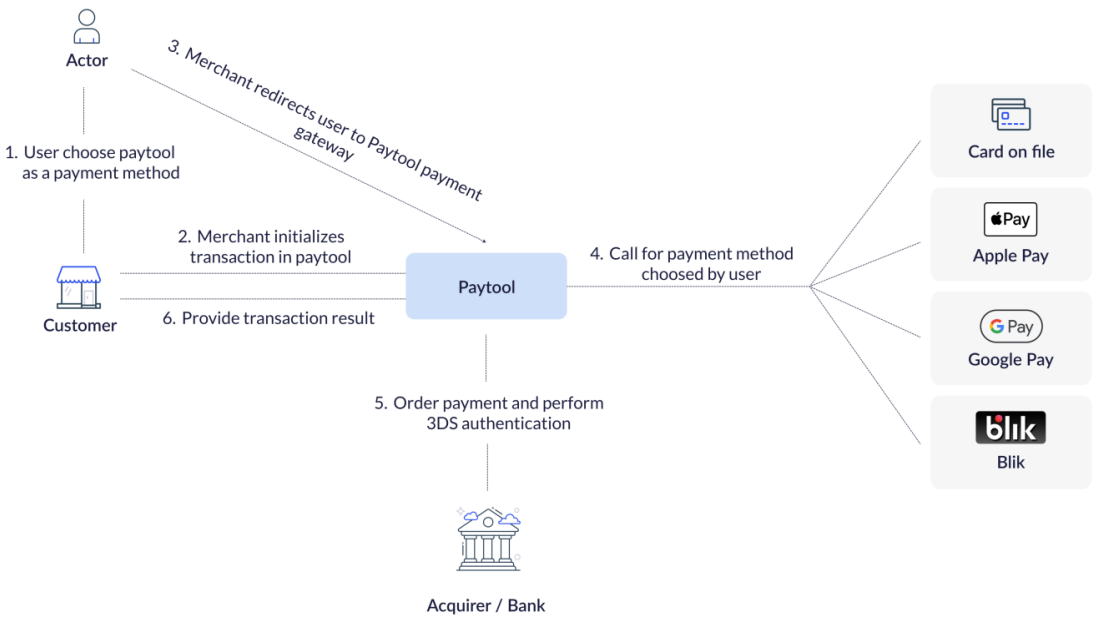
Sequence diagram of the payment via API process

```
@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
@enduml
```

Application components

Application components are all Verestro's internal services which are taking part in the Paytool processes. This chapter describes every component of the Verestro Paytool solution along with it's responsibility.



Verestro Paytool Server

Verestro Paytool Server is the backend component which consists of few internal services responsible for managing cards, card payment tokens and user data, processing transactions and 3D Secure, send notification to the Customer and enduser and storing transaction history. This component is also responsible for connection with Acquirers. Services included in the Verestro Paytool backend can be divided into two groups:

Services that are the part of the Verestro Paytool Solution.
Services supporting the functionalities offered by Verestro Paytool Solution.

Services that are the part of the Verestro Paytool Solution	
Component	**Description**
Paytool API	A service with all methods required to complete the entire transaction process. The methods are called by Paytool Frontend App or by your API in the right order to make the entire payment and 3D Secure process. This service also communicates with the Verestro Acquirer Connector, which orders the execution of the transaction. The last and probably the most important element for which the Paytool API is responsible is opening a payment session and saving the transaction entities in the Verestro system. Another role of this component is to communicate between the Verestro system and the Acquirer's system. This service transfers transaction requests to the Acquirers and also informs if the 3D Secure authentication process is required. This component stores cards in our PCI DSS database in the case of Card on File payment method usage.

Services supporting the functionalities offered by the Verestro Paytool Solution	
Component	**Description**
Notification Service API	A service responsible for sending notifications to end users and Customers. Notifications to end user can be sent via e-mail. The Customer can receive transaction postback via a specific URL he provided.

Verestro Paytool Frontend

Verestro Paytool Frontend is the frontend component consists of two internal services which are responsible for displaying all necessary data comming from Paytool API. Verestro Paytool Frontend can be divided into two services:

Component	Description
Paytool Frontend App	This is a frontend application hosted by Verestro. This is where you redirect the user when you are using the Redirect your payer integration path. This service is intended to display transaction data to the end user, enable him to select a payment method and confirm payment. To perform the above actions, the Paytool Frontend App communicates directly with the Paytool API. This service does not participate in the payment process at all if you use the Payment process via API integration path. Alternatively you can open Paytool in iframe.
Paytool SDK	

Allowed card networks

Listed below are the types of cards supported in transactions using Paytool application:

Card type
`MASTERCARD`
`VISA`
`MAESTRO`

Security

Due to the need to process card data and perform money operations, we had to create security measures that would not allow violations of the transaction process and prevent unauthorized entities from using the solution. In this chapter, we described the main security elements for customers and their transactions.

The sequence diagram below illustrates the application workflow

```
@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 "Customer Frontend" as browser
participant "Customer Backend" as psdk
participant "Paytool Frontend" as pfront
participant "Paytool Backend" as pback
participant "Acquirer" as acq
note right of browser: User chooses "Pay with Verestro Paytool"
browser->psdk: Transaction initialization + metadata
psdk-->pback: Transaction initialization + metadata
pback->pback: Validate transaction metadata
pback->pback: Store transaction session data
pback->psdk: OK + transactionId
psdk->pfront: Redirect end user to Paytool Frontend + transactionId
pfront-->pback: Get transaction metadata + list of the supported payment options
pback-->pfront: Return transaction metadata + list of the supported payment options
pfront->pfront: Display transaction metadata + list of the supported payment options
pfront->pback: Process transaction by chosen payment option
pback->pback: Consisteny validation between current transaction data and provided when
opening the session
pback->acq: Order transaction
@enduml
```

Payment session

In order to start a payment in Paytool, a payment session must first be opened. Opening a payment session involves authorizing your merchant account and sending transaction metadata to Paytool API. Transaction metadata should be encrypted using JWE encryption standard. Based on the obtained transaction data, validation of each data is performed and then a transaction object is created and saved in the internal Verestro database. Thanks to authorization, the context of your merchant account is assigned to the created transaction. Once you have created a payment session, you will receive an identifier for this transaction. Using this identifier you will be able to continue the payment process and we will be able to check whether there has been any interference in the transaction data you provided and interrupt the process if necessary.

> **Note:** We does not store any sensitive data such as PAN or CVC in our system. The obtained data are only required to be transferred to the Acquirer to perform transaction.

Authorization

Authorization of the Customer in the Paytool system is performed using mTLS. Information on how to generate a certificate and the details of the signing process are available in the How to integrate section. Such authorization in Paytool application is intended to check whether the entity trying to execute the request is authorized to do so. If you have a merchant account in the Paytool system, each of your requests should be signed with the certificate x509. This allow us to check whether the action you have taken can be proceeded. We will also check whether your merchant account is associated with a given transaction, and therefore whether it can perform any actions in the context of this transaction

---

Revision #168
Created 1 March 2023 09:22:59
Updated 12 January 2026 08:25:21