COLLECTING SOLUTION

Payment by token - Recurring payment

Implementation Guide

Document version 3.11

Contents

1. HISTORY OF THE DOCUMENT

Version	Author	Date	Comment
3.11	Lyra Collect	3/1/2021	• Addition of the **vads_occurrence_type** field in the *Lifecycle of a recurring payment* chapter. • Addition of fields describing the recurrence in the chapters related to the recurring payment creation notification. • Addition of the *Installment payment* chapter. • Update of the *Payment by token* chapter: the CVV entry and cardholder authentication are required. • Update of the chapters related to token creation: the cardholder must undergo strong authentication.
3.10	Lyra Collect	1/18/2021	• Update of the *Presentation of the service* chapter. • Addition of the *Uniqueness of registered payment methods* chapter. • Clarification on daily recurrences added to chapters related to recurring payment creation. • Update of the chapter *Compatible payment methods*. • Update of the chapter *Automatically canceling a recurring payment via Web Services*. • Addition of the *Test cards* chapter in the appendix.
3.9	Lyra Collect	5/5/2020	• Additional information on the attempts made by the payment gateway in case of activation of anticipated authorizations. • Additional details on payment creation in case of an expired payment method or purged payment data. • Update of notification rule configuration.
3.8	Lyra Collect	10/21/2019	• Update of compatible payment methods. • Addition of a chapter on the payment guarantee during a payment by token. • Update of the description of the **vads_sub_desc** field in the Generating a payment form chapter.
3.7	Lyra Collect	8/5/2019	• The hash algorithm is now available via **Settings** > **Shop**, **Keys** tab. • Addition of the **vads_identifier** field as an input parameter when creating a token. • Addition of the **Risk analysis details** category in the Data dictionary. • **vads_threeds_auth_type**: the field is always present in the response and can be empty. • The hash algorithm is now available via **Settings** > **Shop**, **Keys** tab. • Additional information provided on the computation of the IPN signature. • Additional information provided on the format of the **vads_trans_date** and **vads_presentation_date** fields. • Additional information provided on the format of the fields: **vads_product_label**, **vads_cust_zip**, **vads_order_id**, **vads_cust_first_name**,

Version	Author	Date	Comment
			vads_cust_last_name, **vads_cust_phone**, **vads_cust_cell_phone**, **vads_cust_id**, **vads_cust_city**, **vads_cust_address**. • **vads_auth_result**: correction of field format (an..11) • **vads_contracts**: Possibility to force the Terminal ID to be used.
3.6	Lyra Collect	5/22/2019	Clarifications provided on the methods of creation and termination of recurring payments via web services. Data dictionary: update of **vads_trans_date**.

3.5	Lyra Collect	3/26/2019	• Addition of a clarification on data deletion in the chapter **Managing payments by identifier**.
			• Addition of the time of recurring payment creation in the chapter **Managing payments by identifier**.
			• Update of screenshots in chapters **Creating a token in Test mode** and **Creating a token in Production mode**.
			• Update of screenshots in **Creating a recurring payment via the Expert Back Office**
			• Addition of the chapter **Defining the currency for creating or updating a token**.
			• Update of the table of parameters in the chapter **Creating a "Payment with the token creation option for the cardholder" form**.
			• Data dictionary:
			  • Clarification added about the format of the **vads_product_qty** field.
			  • Addition of the vads_presentation_date field in the **Transaction details** section.
			  • Clarification added about the format of the **vads_identifier** field.
			  • Clarification added about the format of the **vads_subscription** field.
			  • Removal of the fields **vads_ext_info_donation, vads_ext_info_donation_recipient, vads_ext_info_donation_recipient_name, vads_ext_info_donation_merchant, vads_ext_info_donation_contribution** and **vads_risk_primary_warranty**.

3.4	Lyra Collect	3/4/2019	Initial version

2. OBTAINING HELP

Looking for help? Check our FAQ on our website

https://docs.lyra.com/en/collect/faq/sitemap.html

If you have any technical questions or need assistance, our tech support is available

from Monday to Friday from 9 a.m. to 6 p.m.

by phone at: **0811900475** Service fee 0.06 € / min + call charge

by e-mail : *support-ecommerce@lyra-collect.com*

and via your Expert Back Office, **Help** > **Contact support**

To facilitate the processing of your demands, you will be asked to communicate your shop ID (an 8-digit number).

3. PRESENTATION OF THE SERVICE

Management of payments by token

The payment by token management service allows merchants to offer their clients the possibility to associate a token with a payment method, which will facilitate their subsequent payments on the website (no more need to re-enter the credit card number or the IBAN).

Tokens allow you to:

- Make fast and secure payments.

  For the buyer - avoid filling in bank details when making subsequent payments (1-click payment).

  The gateway stores the bank details in a highly secure environment, in accordance with the PCI-DSS requirements. Only the token is transferred during the exchange.

- Make recurring payments (subscriptions).

The service also allows you to:

- Identify cards that are due to expire, in order to notify the Merchant via a file containing the token of the expiring card.
- Update the bank details associated with a token via the payment page, or manually via the Expert Back Office.
- Automatically detect if the payment method has expired and offer an update in case of payment by token.
- When creating a token, detect if the payment method has been previously registered.
- Manage other buyer detail updates.

In compliance with the banking data security and protection rules implemented by PCI DSS, the payment method details are destroyed after the associated token has not been used for 15 months.

The token will remain visible in the Expert Back Office and can be updated with new details.

Recurring payment (subscription) management

The recurring payment management service allows merchants to create subscriptions with **fixed amounts and payment schedule**, also known as “recurring payments”, with or without an expiry date, within the limits of the card validity period.

When creating a recurring payment, the merchant defines the start date, the amount of the installments and the recurrence rule to apply.

Once the start date (also known as the “effective date”) has been reached, the payment gateway automatically processes the installments.

After that, the merchant can no longer edit the amount of the installment payments.

In TEST mode, transactions are created every hour in order to allow the Merchant to easily test the IPN processing.

In PRODUCTION mode, transactions are created once a day between midnight and 5 a.m.

3.1. Uniqueness of registered payment methods

By default, the gateway authorizes the buyer to register their payment method several times on the merchant website.

However, if the merchant wants to, they can enable an option via their Expert Back Office that will allow them to detect when a token is created, if the payment method has been previously registered.

IMPORTANT
It is not recommended to enable the uniqueness check of the registered payment method if you do not control its impact on your implementation.

Operating principle

Once the option is activated, the gateway verifies the validity of the payment method with the issuer each time a token is created, and then proceeds to verify the uniqueness of the payment method.

If the payment method has never been registered, then a new token associated with this payment method is created and its identifier is returned to the merchant website upon the end of payment notification.

If the payment method has already been registered (same number and expiration date), then an existing token is used and its identifier is returned to the merchant website upon the end of payment notification.

The returned buyer's data is the same as the data transmitted by the merchant, and not the same as the details of the previously registered token.

The field **vads_identifier_status** is set to **CREATED**, even if in this case no token is created.

The notification then contains an additional field set to **true**:

- **vads_identifier_previously_registered** for the notification in the Hosted Payment Form format.
- **paymentMethodTokenPreviouslyRegistered** for the notification in the REST API format.

Notes

- There is no detection of payment method uniqueness during the token update.
- If the payment method is already associated with several tokens, the end of payment notification contains the identifier of the most recent token.
- The creation of a token via Expert Back Office is refused if the payment method is already associated with another token.
- The **vads_identifier_previously_registered** field is not returned upon return to shop.
- The **vads_identifier_previously_registered** field is never returned in the end of payment notification if no duplicate payment methods are detected. Therefore, the **false** value is never sent to the merchant website.

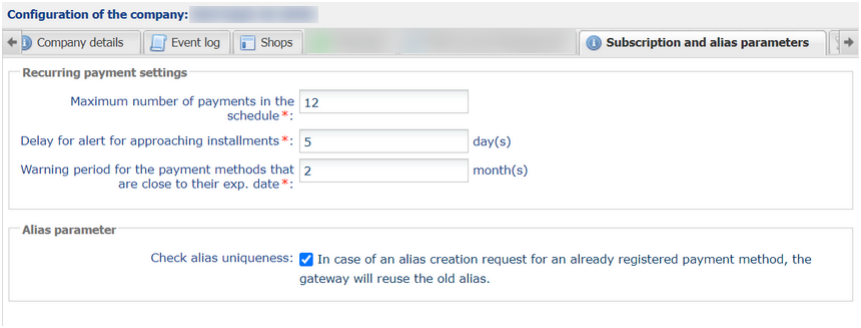
What should I do if a duplicate payment method is detected?

It depends on your business requirements.

- You can decide to do nothing and provide the service or deliver the goods to the buyer.
- You can check whether the customer code associated with the existing token matches the buyer's customer code. If this is not the case, you can search if a family tie between the two customers explains why the same payment method is used by two different customers.
- You can check if the person requesting the registration of the payment method is the same person who has already registered this payment method (e.g. by checking their contact details, e-mail address, country etc.).
- If all the controls put in place fail, that means that you might be a victim of fraud and can then decide to cancel the payment.

Activation of the payment method uniqueness detection

1. Via the Expert Back Office, go to **Settings** > **Company,** then click on the **Subscription and alias parameters** tab.



2. In the **Alias parameter** section, check the **Check alias uniqueness** box.
3. Click the **Save** button to save the changes.

4. COMPATIBLE PAYMENT METHODS

List of payment methods compatible with Payment by token Management Service:

Network code	Payment method	Card types (vads_payment_cards)	Supports payment by token
ACCORD	Illicado Gift Card	ILLICADO	✖
ACCORD	Accord brand card	ACCORD_STORE	✖
ACCORD_SANDBOX	JouéClub gift card - Sandbox mode	ILLICADO_SB	✖
AMEXGLOBAL	American Express	AMEX	✔
ANCV	e-Chèque-Vacances	E_CV	✖
AURORE	Cpay card	AURORE-MULTI	✖
CB	CB bank card	CB	✔
CB	Maestro	MAESTRO	✖
CB	Mastercard	MASTERCARD	✔
CB	Apetiz Meal Voucher card	APETIZ	✔
CB	Chèque Déjeuner Meal Voucher card	CHQ_DEJ	✔
CB	1st generation Mastercard Meal Voucher card	EDENRED	✔
CB	Sodexo Meal Voucher card	SODEXO	✔
CB	e-Carte Bleue virtual card	E-CARTEBLEUE	✔
CB	Visa	VISA	✔
CB	Visa Electron	VISA_ELECTRON	✔
CB	Visa Vpay	VPAY	✔
CONECS	Apetiz Meal Voucher card	APETIZ	✔
CONECS	Chèque Déjeuner Meal Voucher card	CHQ_DEJ	✔
CONECS	Conecs Meal Voucher card	CONECS	✔
CONECS	Sodexo Meal Voucher card	SODEXO	✔
CVCONNECT	Connect Chèque-Vacances	CVCO	✖
DINERS	Diners Club	DINERS	✔
DINERS	Discover	DISCOVER	✔
EDENRED	Edenred Ticket Eco Chèque	EDENRED_EC	✔
EDENRED	Sport & Culture Edenred Ticket	EDENRED_SC	✔
EDENRED	Edenred Ticket Compliment	EDENRED_TC	✔
EDENRED	Edenred meal voucher	EDENRED_TR	✔
FRANFINANCE	Franfinance payment in 3 installments	FRANFINANCE_3X	✖
FRANFINANCE	Franfinance payment in 4 installments	FRANFINANCE_4X	✖
FRANFINANCE_SB	Franfinance payment in 3 installments - Sandbox mode	FRANFINANCE_3X	✖
FRANFINANCE_SB	Franfinance payment in 4 installments - Sandbox mode	FRANFINANCE_4X	✖
FULLCB	Payment in 3 installments with no fees with BNPP PF	FULLCB3X	✖
FULLCB	Payment in 4 installments with no fees with BNPP PF	FULLCB4X	✖
GATECONEX	Bancontact	BANCONTACT	✖

Network code	Payment method	Card types (vads_payment_cards)	Supports payment by token
GATECONEX	Diners Club	DINERS	✓
GATECONEX	Discover	DISCOVER	✓
GATECONEX	e-Carte Bleue virtual card	E-CARTEBLEUE	✓
GATECONEX	Maestro	MAESTRO	✗
GATECONEX	Mastercard	MASTERCARD	✓
GATECONEX	Visa	VISA	✓
GATECONEX	Visa Electron	VISA_ELECTRON	✗
GATECONEX	Visa Vpay	VPAY	✗
GICC	Bancontact	BANCONTACT	✗
GICC	Maestro	MAESTRO	✗
GICC	Mastercard	MASTERCARD	✓
GICC	Visa	VISA	✓
GICC	Visa Electron	VISA_ELECTRON	✗
GICC_DINERS	Diners Club	DINERS	✓
GICC_DINERS	Discover	DISCOVER	✓
GICC_MAESTRO	Bancontact	BANCONTACT	✗
GICC_MAESTRO	Maestro	MAESTRO	✗
GICC_MASTERCARD	Mastercard	MASTERCARD	✓
GICC_VISA	Visa	VISA	✓
GICC_VISA	Visa	VPAY	✗
GICC_VISA	Visa Electron	VISA_ELECTRON	✗
GIROPAY	GIROPAY wire transfer	GIROPAY	✗
GOOGLEPAY	Google Pay wallet payment	GOOGLEPAY	✗
IDEAL	iDEAL wire transfer	IDEAL	✗
JCB	JCB	JCB	✓
KLARNA	Klarna invoice payment	KLARNA	✗
LYRA_COLLECT PPRO	Payment by Alipay Wallet	ALIPAY	✗
LYRA_COLLECT PPRO	Bancontact	BANCONTACT	✗
LYRA_COLLECT PPRO	GIROPAY wire transfer	GIROPAY	✗
LYRA_COLLECT PPRO	iDEAL wire transfer	IDEAL	✗
LYRA_COLLECT PPRO	Payment by voucher within the MultiBanco network	MULTIBANCO	✗
LYRA_COLLECT PPRO	MyBank wire transfer	MYBANK	✗
LYRA_COLLECT PPRO	Przelewy24 wire transfer	PRZELEWY24	✗
LYRA_COLLECT PPRO	Sofort wire transfer	SOFORT_BANKING	✗
LYRA_COLLECT PPRO	UnionPay card	UNION_PAY	✗
LYRA_COLLECT PPRO	Payment by We Chat Pay wallet	WECHAT	✗
ONEY	FacilyPay card payment in 3 or 4 installments	ONEY	✗
ONEY_API	Payment in 3 or 4 installments with Oney	ONEY	✗
ONEY_API_SB	Oney payment in 3 or 4 installments - Sandbox mode	ONEY	✗
ONEY_SANDBOX	FacilyPay card payment in 3 or 4 installments - Sandbox mode	ONEY_SANDBOX	✗
PAYDIREKT_V2	Paydirekt wire transfer	PAYDIREKT	✗
PAYLIB	Payment by Paylib	PAYLIB	✗
PAYPAL	Payment by PayPal	PAYPAL	✓

Network code	Payment method	Card types (vads_payment_cards)	Supports payment by token
PAYPAL_SB	Payment by PayPal - Sandbox mode	PAYPAL_SB	✓
PLANET_DCC	Mastercard	MASTERCARD	✓
PLANET_DCC	Visa	VISA	✓
POSTFINANCEV2	Payment by PostFinance Card	POSTFINANCE	✗
POSTFINANCEV2	Efinance PostFinance wire transfer	POSTFINANCE_EFIN	✗
PRESTO	Presto by Cetelem online credit solution	PRESTO	✗
SEPA	SEPA DIRECT DEBIT	SDD	✓
SOFORT	Sofort wire transfer	SOFORT_BANKING	✗
WIRECARD	Euro-Cheque card	ECCARD	✓
WIRECARD	GIROPAY wire transfer	GIROPAY	✗
WIRECARD	Maestro	MAESTRO	✓
WIRECARD	Mastercard	MASTERCARD	✓
WIRECARD	SEPA DIRECT DEBIT	SDD	✓
WIRECARD	Visa	VISA	✓
WIRECARD	Visa Electron	VISA_ELECTRON	✓
WIRECARD	Visa Vpay	VPAY	✓

5. TOKEN SHARING

It is possible to share tokens between several legal entities.

Tokens shared between several legal entities have to be unique and must be generated by the payment gateway.

However, this feature is subject to conditions. Please contact the customer service of your payment gateway to know the details.

6. MANAGING TOKENS VIA THE PAYMENT FORM

The payment form allows to make the following operations classified according to different cases.

Each of these cases corresponds to a different value of the **vads_page_action** field.

Use case	Value of the vads_page_action field
Creation of a token without payment	REGISTER
Update of information associated with the token	REGISTER_UPDATE
Creation of a token during a payment	REGISTER_PAY
Creation of a token during a recurring payment	REGISTER_SUBSCRIBE
Creation of a token during creation of a recurring payment with payment	REGISTER_ PAY_SUBSCRIBE
Payment by token	PAYMENT
Using a token to create a recurring payment	SUBSCRIBE
Payment with option for the cardholder to create a token	ASK_ REGISTER_PAY
Update of information associated with the token during a payment	REGISTER_UPDATE_PAY

Depending on the use case (value of the **vads_page_action** field), the payment process from the buyer's point of view will be different on the payment page.

6.1. Creating a token without payment

This case corresponds to a simple token creation.

The merchant website transmits buyer details to the payment gateway, in particular the e-mail address, which is mandatory.

1. The buyer verifies the information displayed on the payment page.
2. He or she clicks on the payment method that will be registered.

   The payment page displays the buyer details once again and prompts to enter the banking details.
3. If the buyer has selected a bank card, he or she enters:
   - the card number,
   - the card expiry month,
   - the card expiry year,
   - the CVV code of the card, if there is one.
4. The buyer clicks **Validate**.

   As part of the implementation of PSD2, the cardholder will be required to undergo strong authentication.

   If all the payment method verification processes have been successfully completed, the summary appears.

   It contains the newly created token. It can be later used for another financial operation.

   It is possible to send these details to the buyer by e-mail and receive:
   - the confirmation of the registration of these banking details on the payment gateway of the shop,
   - the buyer's token that they can later use for another financial operation.

   Note:

   *To find out how to send these details by e-mail, see chapter **Configuring e-mails sent to the buyer**.*

The processing of a token creation request without payment results in the creation of a **VERIFICATION** type transaction, visible in the Expert Back Office, that has the following characteristics:

- its amount is 1.00 EUR or 0.00 EUR, if supported by the acquirer,
- its status is either "Accepted" (vads_trans_status=ACCEPTED) or "Refused" (vads_trans_status=REFUSED),
- it is never captured and remains in the "Transactions is progress" tab.

Warning:

The token will not be created if the authorization or information request is rejected.

6.2. Updating token details

This case corresponds to updating, at the initiative of the buyer, the information related to his or her payment method and/or his or her personal information.

The merchant website transmits to the payment gateway:

- new information, specifically the e-mail address, which is mandatory,
- the token for update.

The presented pages are identical to the previous case (Creating a token without payment).

The processing of a token update request results in the creation of a VERIFICATION transaction type, visible in the Expert Back Office, that has the following characteristics:

- its amount is 1.00 EUR or 0.00 EUR, if supported by the acquirer,
- its status is either "Accepted" (vads_trans_status=ACCEPTED) or "Refused" (vads_trans_status=REFUSED),
- it is never captured and remains in the "Transactions is progress" tab.

Warning:

The token will not be updated if the authorization or information request is rejected.

6.3. Creating a token during a payment

In this case, the parameters necessary for registration are completed by the parameters necessary for a payment request.

The merchant website transmits buyer details to the payment gateway, in particular the e-mail address, which is mandatory.

1. The buyer verifies the information displayed on the payment page (identity, amount and currency of the transaction).

2. He or she clicks on the payment method that will be recorded and used.

   The payment page appears:

   - for the recording:
     - information about the buyer's identity,
     - entry of buyer's banking details.
   - for the payment:
     - transaction details (transaction number, amount...).

3. If the buyer has selected a bank card, he or she enters:

   - the card number,
   - the card expiry month,
   - the card expiry year,
   - the CVV code of the card, if there is one.

4. The buyer clicks **Validate**.

   As part of the implementation of PSD2, the cardholder will be required to undergo strong authentication.

   If all the payment method verification processes have been successfully completed, the summary appears.

   It contains the newly created token. It can be later used for another financial operation.

   It is possible to send these details to the buyer by e-mail and receive:

   - the confirmation of the registration of these banking details on the payment gateway of the shop,
   - confirmation of the payment.

   Note:

   *To find out how to send these details by e-mail, see chapter **Configuring e-mails sent to the buyer**.*

Warning:

The token will not be created if the authorization or information request is rejected.

6.4. Creating a token during a recurring payment

In addition to the information used in the case of **Creating a token without payment**, this use case must also include information related to the recurring payment, such as:

- the initial amount of the recurring payment (amount used for the first installment/s) if it is different (optional),
- the amount of the recurring payment (amount of installments or the amount used for the following installments when the first one is different).

The merchant website transmits buyer details to the payment gateway, in particular the e-mail address, which is mandatory.

1. The buyer verifies the information displayed on the payment page (identity, amount and currency of the transaction).

2. He or she clicks on the payment method that will be recorded and used.

   The payment page appears. It contains the following information:

   - for the recording:
     - information about the buyer's identity,
     - entry of buyer's banking details.
   - for the recurring payment:
     - the number of installments,
     - the amount per installment.

3. If the buyer has selected a bank card, he or she enters:

   - the card number,
   - the card expiry month,
   - the card expiry year,
   - the CVV code of the card, if there is one.

4. The buyer clicks **Validate**.

   As part of the implementation of PSD2, the cardholder will be required to undergo strong authentication.

   If all the payment method verification processes have been successfully completed, the summary appears.

   It contains the newly created token. It can be later used for another financial operation.

   The amounts of the recurring payment also appear in the payment summary when the payment method number verification (example: bank card) has been successfully completed.

   It is possible to send these details to the buyer by e-mail and receive:

   - the confirmation of the registration of these banking details on the payment gateway of the shop,
   - confirmation of the registration of the recurring payment.

   Note:

   *To find out how to send these details by e-mail, see chapter **Configuring e-mails sent to the buyer**.*

The processing of a token creation with creation of a recurring payment results in the creation of a VERIFICATION transaction type, visible in the Expert Back Office and having the following characteristics:

- its amount is 1.00 EUR or 0.00 EUR, if supported by the acquirer,
- its status is either "Accepted" (vads_trans_status=ACCEPTED) or "Refused" (vads_trans_status=REFUSED),
- it is never captured and remains in the "Transactions is progress" tab.

Warning:

The token will not be created if the authorization or information request is rejected.

6.5. Creating a token during creation of a recurring payment with payment

In this use case, the following information must be visible:

- the buyer details,
- the transaction identifier,
- the recurring payment details (amounts).

Example: a recurring payment of X EUR/ over N months with commission fees to be paid upon taking the order.

The merchant website transmits buyer details to the payment gateway, in particular the e-mail address, which is mandatory.

1. The buyer verifies the information displayed on the payment page (identity, amount and currency of the transaction).

2. He or she clicks on the payment method that will be recorded and used.

   The payment page appears. It contains the following information:

   - for the recording:
     - information about the buyer's identity,
     - entry of buyer's banking details.
   - for the recurring payment:
     - the number of installments,
     - the amount per installment.
   - for the payment:
     - the amount of commission fees.

3. If the buyer has selected a bank card, he or she enters:

   - the card number,
   - the card expiry month,
   - the card expiry year,
   - the CVV code of the card, if there is one.

4. The buyer clicks **Validate**.

   As part of the implementation of PSD2, the cardholder will be required to undergo strong authentication.

   If all the payment method verification processes have been successfully completed, the summary appears.

   It contains the newly created token. It can be later used for another financial operation.

   The amounts of the recurring payment as well as the commission fees also appear in the payment summary when the payment method number verification (example: bank card) has been successfully completed.

   It is possible to send these details to the buyer by e-mail and receive:

   - the confirmation of the registration of these banking details on the payment gateway of the shop,
   - confirmation of the recurring payment registration.

- confirmation of the payment.

Note:

*To find out how to send these details by e-mail, see chapter **Configuring e-mails sent to the buyer**.*

Warning:

The token will not be created if the authorization or information request is rejected.

6.6. Payment by token

Payment by token allows to use a pre-registered token for making single or multiple payments without having to select a payment method and enter banking details.

In this case, a simple confirmation step is presented with a summary of the transaction (number and amount).

1. The buyer verifies the information displayed on the payment page.

2. The buyer clicks **Validate**.

   Under PSD2, the CVV entry and cardholder authentication are required.

   An authorization request is made with the payment method associated to the token. If it is successfully completed, a summary appears.

   It is possible to send these details to the buyer by e-mail and receive:

   - the buyer's token that can be used later for another financial operation,
   - the payment details.

   Note:

   *To find out how to send these details by e-mail, see chapter **Configuring e-mails sent to the buyer**.*

   Note:

   When the token is associated with an expired payment method, the gateway automatically suggests to the buyer to enter the new banking details in order to perform the payment and update the associated token.

   The token will not be created if the authorization or information request is rejected.

Payment by token and Liability shift

By default, the liability shift in case of a legal dispute cannot be applied to a payment by token.

Even if a 3D Secure authentication was successfully performed when the token was created, the payments made with this token do not automatically benefit from the liability shift.

In order for the liability shift to be applied, the buyer must enter the CVV code of his or her card and proceed to 3D Secure authentication (or strong authentication in case of 3DS2).

Contact the Middle Office to request the activation of these options.

Note

By default (i.e. without the activation of the option "3D Secure for payments by token"), the details of the 3D Secure authentication (vads_threeds_xxxx field), are returned empty in the response, with the exception of the vads_threeds_exit_status field.

6.7. Using a token to create a recurring payment

Once a token has been created, it is possible to add one or several additional recurring payments that will use this token.

After a new recurring payment has been created, no banking detail entry will be requested. Only a confirmation on the buyer's part will be needed.

1. The buyer verifies the information displayed on the payment page.

2. The buyer clicks **Validate**.

   A summary of the created recurring payment appears.

   It is possible to send these details to the buyer by e-mail and receive:

   - the buyer’s token that can be used later for another financial operation,
   - the recurring payment details.

   Note:

   *To find out how to send these details by e-mail, see chapter **Configuring e-mails sent to the buyer**.*

Warning:

The token will not be created if the authorization or information request is rejected.

6.8. Payment with the token creation option for the cardholder

At the moment of payment, the buyer has the possibility to remember the bank details for later by ticking the corresponding checkbox. This completely secure operation facilitates future purchases for the buyer.

The merchant website transmits buyer details to the payment gateway, in particular the e-mail address, which is mandatory.

1. The buyer verifies the information displayed on the payment page (amount and currency of the transaction).

2. He or she clicks on the payment method that will be recorded and used.

   The payment page displays the information related to the buyer and prompts to enter the banking details.

   The buyer has the possibility to check the box **I would like to register my payment method details for a future purchase**. By default, this checkbox is unchecked. However, the buyer has the possibility to remember these banking details on the payment gateway of the shop. Their future purchases will be simplified.

3. If the buyer has selected a bank card, he or she enters:

   - the card number,
   - the card expiry month,
   - the card expiry year,
   - the CVV code of the card, if there is one.

4. The buyer clicks **Validate**.

   If the buyer has chosen to register their payment method, strong authentication is required.

   Once the payment method has been verified and the authorization or information request has been accepted by the bank, a summary appears with the following message:

   Your registration request with payment has been successfully recorded.

   This summary contains information that is related both to the token (the buyer ID) that can be used later for another financial operation and to the payment.

   It is possible to send these details to the buyer by e-mail and receive:

   - the confirmation of the payment,
   - the confirmation of the registration of these banking details on the shop's payment gateway.

   Note:

   *To find out how to send these details by e-mail, see chapter **Configuring e-mails sent to the buyer**.*

Warning:

The token will not be created if the authorization or information request is rejected.

7. LIFECYCLE OF A RECURRING PAYMENT

The recurring payment starts on its effective date.

The payment gateway starts creating payments following the schedule determined by the recurring payment rule sent in the form of recurring payment creation (vads_sub_desc field).

Upon each installment, if the **Instant Payment Notification URL when creating a recurring payment** rule is enabled and correctly configured, the merchant website will receive the payment result via their notification URL (IPN).

The notification contains:

- The **vads_subscription** field that indicates the recurring payment reference.
- The **vads_recurrence number** field that indicates the installment number.
- The **vads_occurrence_type** field that indicates the installment in question (first, n^{th} or last installment).
- The **vads_trans_status** field that indicates the payment status (accepted or refused).

If the payment has been refused:

- the merchant will not be notified by e-mail,
- the payment will not be automatically presented once again.

If the payment method has reached its expiry date, a refused transaction is created without a call to the issuing bank. The error details (vads_payment_error) are set to 8 - The card expiration date does not allow this action.

If the payment method data has been purged following 15 moths of inactivity, a refused transaction is created without a call to the issuing bank. The error description (vads_payment_error) is set to 107 - The payment method associated with the token is no longer valid.

Special case of daily recurrences

If you request the creation of a recurring payment to debit the payment method holder daily (RRULE:FREQ=DAILY;INTERVAL=1), and the requested effective date (vads_sub_effect_date) corresponds to the recurring payment creation date, then when the payment gateway processes this recurring payment the following morning (between midnight and 5 a.m.), 2 payments will be created:

- The payment from the previous day (that corresponds to the effective date)
- And the payment from the current day

To avoid this, it is recommended to transmit an effective date the day after the recurring payment is created.

8. LIFECYCLE OF A RECURRING PAYMENT WITH ANTICIPATED AUTHORIZATION

Thanks to this option, when a payment is rejected for a non-fraud related motive (see chapter ***List of return codes of the authorization request***), the payment gateway can automatically make another daily authorization request up to two days before the expected capture date at the bank.

As soon as the option is enabled for the shop, recurring payments are created 6 days before the date anticipated by the recurring payment rule.

It is necessary to enable the **Instant Payment Notification URL on batch authorization** rule via your Expert Back Office (see chapter ***Setting up a notification on batch authorization***).

D-6: **Creating a recurring payment**

- The authorization request is accepted.
  - The payment status will remain **Waiting for capture** until the initially scheduled date.
  - A call to the notification URL will be triggered if the **Instant Payment Notification URL when creating a recurring payment** rule is enabled. The request will contain the following values:

Field name	Value
vads_url_check_src	**REC**
vads_trans_status	**AUTHORISED**

- The authorization request is rejected for a fraud-related reason.
  - The payment is definitively **Refused**.
  - A call to the notification URL will be triggered if the **Instant Payment Notification URL when creating a recurring payment** rule is enabled. The request will contain the following values:

Field name	Value
vads_url_check_src	**REC**
vads_trans_status	**REFUSED**

- The authorization request is rejected for a non fraud-related reason.
  - The payment status will remain **Waiting for authorization**.
  - A call to the notification URL will be triggered if the **Instant Payment Notification URL when creating a recurring payment** rule is enabled. The request will contain the following values:

Field name	Value
vads_url_check_src	**REC**
vads_trans_status	**WAITING_AUTHORISATION**

D-5, D-4, D-3: **New authorization request triggered automatically**

- The authorization request is accepted.
  - The payment status will remain **Waiting for capture** until the initially scheduled date.
  - A call to the notification URL will be triggered if the **Instant Payment Notification URL on batch authorization** rule is enabled. In this case, the request will contain the following values:

Field name	Value
vads_url_check_src	**BATCH_AUTO**
vads_trans_status	**AUTHORISED**

- The authorization request is rejected for a fraud-related reason.
  - The payment is definitively **Refused**.
  - A call to the notification URL will be triggered if the **Instant Payment Notification URL on batch authorization** rule is enabled. In this case, the request will contain the following values:

Field name	Value
vads_url_check_src	**BATCH_AUTO**
vads_trans_status	**REFUSED**

- The authorization request is rejected for a non fraud-related reason.
  - The payment status will remain **Waiting for capture** until the scheduled date.
  - A call to the notification URL will be triggered if the **Instant Payment Notification URL on batch authorization** rule is enabled. In this case, the request will contain the following values:

Field name	Value
vads_url_check_src	**BATCH_AUTO**
vads_trans_status	**WAITING_AUTHORISATION**

D-2: **Last authorization request triggered automatically**

- The authorization request is accepted.
  - The payment status will remain **Waiting for capture** until the initially scheduled date.
  - A call to the notification URL will be triggered if the **Instant Payment Notification URL on batch authorization** rule is enabled. In this case, the request will contain the following values:

Field name	Value
vads_url_check_src	**BATCH_AUTO**
vads_trans_status	**AUTHORISED**

- The authorization request is rejected (regardless of the reason).
  - The payment is definitively **Refused**.
  - A call to the notification URL will be triggered if the **Instant Payment Notification URL on batch authorization** rule is enabled. In this case, the request will contain the following values:

Field name	Value
vads_url_check_src	**BATCH_AUTO**
vads_trans_status	**REFUSED**

D: **Capture of the transaction**

The payment is automatically captured.

No calls to the Notification URL will be triggered.

Notes

- *Payment rejection between D-6 and D-2*

  When the payment is definitively rejected, you must make sure that the access to the service for which the recurring payment was created is canceled on the scheduled date and not on the day when the payment rejection was received.

- *Termination of a recurring payment*

  When the buyer cancels their recurring payment, the merchant must cancel the already scheduled payments.

8.1. List of authorization request return codes

The return codes of the authorization request are returned by the issuing bank (if available).

Codes returned by the **CB** network:

Value	Description	Grounds of fraud	Value	Description	Grounds of fraud
00	Approved or successfully processed transaction		**43**	Stolen card	YES
02	Contact the card issuer		**51**	Insufficient balance or exceeded credit limit	
03	Invalid acceptor	YES	**54**	Expired card	YES
04	Keep the card	YES	**55**	Incorrect secret code	
05	Do not honor	YES	**56**	Card absent from the file	YES
07	Keep the card, special conditions	YES	**57**	Transaction not allowed for this cardholder	YES
08	Confirm after identification		**58**	Transaction not allowed for this cardholder	
12	Incorrect Transaction Code	YES	**59**	Suspected fraud	YES
13	Incorrect Transaction Amount	YES	**60**	The acceptor of the card must contact the acquirer	
14	Invalid cardholder number	YES	**61**	Withdrawal limit exceeded	
15	Unknown issuer	YES	**63**	Security rules unfulfilled	YES
17	Canceled by the buyer		**68**	Response not received or received too late	
19	Retry later		**75**	Number of attempts for entering the secret code has been exceeded	
20	Incorrect response (error on the domain server)		**76**	The cardholder is already blocked, the previous record has been saved	YES
24	Unsupported file update		**80**	Contactless payment is not accepted by the issuer	YES
25	Unable to locate the registered elements in the file		**81**	Unsecured payment is not accepted by the issuer	YES
26	Duplicate registration, the previous record has been replaced		**82**	Revocation of recurring payment for the card of a specific Merchant or for the MCC and the card	YES
27	File update edit error		**83**	Revocation of all recurring payments for the card	YES
28	Denied access to file		**90**	Temporary shutdown	
29	Unable to update		**91**	Unable to reach the card issuer	
30	Format error		**94**	Duplicate transaction	
31	Unknown acquirer company ID	YES	**96**	System malfunction	
33	Expired card	YES	**97**	Overall monitoring timeout	
34	Suspected fraud	YES	**98**	Server not available, new network route requested	
38	Expired card		**99**	Initiator domain incident	
41	Lost card	YES			

8.2. E-mail notification in case of installment rejection

Case of the option "Anticipated authorization"

When a payment is declined, a notification e-mail is sent to the merchant.

9. ESTABLISHING INTERACTION WITH THE PAYMENT GATEWAY

The merchant website and the payment gateway interact by exchanging data.

To create a payment, this data is sent in an HTML form via the buyer's browser.

At the end of the payment, the result is transmitted to the merchant website in two ways:

- Automatically by means of notifications called Instant Notification URLs (also called IPN or Instant Payment Notification), see chapter **Setting up notifications**.
- Via the browser, when the buyer clicks the button to return to the merchant website, see chapter **Managing the return to the merchant website**.

To guarantee the security of the exchange, the data is signed with a key known only to the merchant and the payment gateway.

9.1. Similarities with single payment

All the functionalities available for single payments are also available for payments by token and for recurring payments.

For more information, see the *Hosted Payment Page Implementation Guide*.

Here is a non-exhaustive list:

- Single payment, made in one operation, or split.
- Management of several currencies.
- Management of several payment methods and associated MIDs.
- Personalization of payment pages.
- Display in an iframe.

10. SETTING UP NOTIFICATIONS

Several types of notifications are provided in the Expert Back Office.

- Instant Payment Notification URL call
- E-mail sent to the merchant
- E-mail sent to the buyer
- SMS sent to the merchant
- SMS sent to the buyer

They allow to manage the events (payment accepted, payment abandoned by the buyer, payment canceled by the merchant, payment validated by the merchant, etc.) that will trigger a notification sent to the merchant website, the merchant or the buyer.

The notifications of Instant Payment Notification URL call type are very important as they represent the only reliable way of obtaining the payment result for the merchant website.

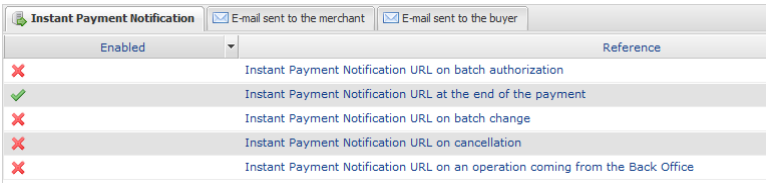
If the payment gateway is unable to access the URL of your page, an e-mail will be sent to the shop administrator.

It contains:

- The HTTP code of the encountered error,
- Parts of error analysis,
- Its consequences,
- Instructions via the Expert Back Office to resend the request to the previously defined URL.

To access notification rule management:

1. Sign in to your Lyra Collect Back Office available at this address:
   https://secure.lyra.com/portal/
2. Click **Other actions** to access Expert Back Office.
3. Go to the following menu: **Settings** > **Notification rules**.



10.1. Setting up the Instant Payment Notification

This rule allows to notify the merchant website in the following cases:

- Payment accepted
- Payment refused
- Token creation or update
- Creation of a recurring payment

This notification is required for communicating the result of a payment request, token or recurring payment creation.

It will inform the merchant website of the result even if the buyer has not clicked the "Return to the shop" button.

1. Right-click **Instant Payment Notification URL at the end of the payment**.
2. Select **Manage the rule**.
3. Enter the **E-mail address(es) to notify in case of failure** field in the **General settings** section.
   To specify several e-mail addresses, separate them with a semi-colon.
4. Check the box **Automatic retry in case of failure** if you wish to authorize the gateway to automatically resend the notification in case of a failure (can be done up to 4 times).
   For more information, please see chapter *Automatic retry in case of failure* on page 36.
5. In the **Instant Payment Notification URL of the API form V1, V2** section, specify the URL of your page in the fields **URL to notify in TEST mode** and **URL to notify in PRODUCTION mode** if you wish to receive notifications in the API form format.
6. Save the changes.

10.2. Setting up notifications in case of abandoned or canceled payments

This rule allows to notify the merchant website in the following cases:

- When the buyer abandons/cancels a payment - via the **Cancel and return to shop** button.
- When the buyer has not completed the payment and the payment session has expired.

  The maximum length of a payment session is 10 minutes.

This customization is **mandatory** if you are using the **FacilyPay Oney** payment method.

This rule is **disabled by default**.

1. Right-click **Instant Payment Notification URL on cancellation**.
2. Select **Manage the rule**.
3. Enter the **E-mail address(es) to notify in case of failure** field in the **General settings** section.

   To specify several e-mail addresses, separate them with a semi-colon.
4. Check the box **Automatic retry in case of failure** if you wish to authorize the gateway to automatically resend the notification in case of a failure (can be done up to 4 times).

   For more information, please see chapter *Automatic retry in case of failure* on page 36.
5. In the **Instant Payment Notification URL of the API form V1, V2** section, specify the URL of your page in the fields **URL to notify in TEST mode** and **URL to notify in PRODUCTION mode** if you wish to receive notifications in the API form format.
6. In the **REST API Instant Payment Notification URL** section, specify the URL of your page in the fields **Target URL of the IPN to notify in TEST mode** and **Target URL of the IPN to notify in PRODUCTION mode** if you are using the JavaScript client.
7. Save the changes.
8. Enable the rule by right-clicking **Instant Payment Notification URL on cancellation** and select **Enable the rule**.

10.3. Instant Payment Notification URL on an operation coming from the Back Office

This rule allows to notify the merchant website about every operation made via the Expert Back Office:

- Creation of a manual payment (accepted or rejected)
- Transaction update
- Transaction duplication
- Transaction refund
- Transaction cancellation
- Transaction validation
- Token creation
- Token update

1. Right-click **Instant Payment Notification URL on an operation coming from the Back Office**.
2. Select **Manage the rule**.
3. Enter the **E-mail address(es) to notify in case of failure** field in the **General settings** section.
   To specify several e-mail addresses, separate them with a semi-colon.
4. Check the box **Automatic retry in case of failure** if you wish to authorize the gateway to automatically resend the notification in case of a failure (can be done up to 4 times).
   For more information, please see chapter *Automatic retry in case of failure* on page 36.
5. In the **Instant Payment Notification URL of the API form V1, V2** section, specify the URL of your page in the fields **URL to notify in TEST mode** and **URL to notify in PRODUCTION mode** if you wish to receive notifications in the API form format.
6. In the **REST API Instant Payment Notification URL** section, specify the URL of your page in the fields **Target URL of the IPN to notify in TEST mode** and **Target URL of the IPN to notify in PRODUCTION mode** if you are using the JavaScript client.
7. Save the changes.
8. Enable the rule by right-clicking **Instant Payment Notification URL on an operation coming from the Back Office** and select **Enable the rule**.

10.4. Setting up a notification upon creating a recurring payment

This rule allows to notify the merchant website in the following cases:

- When the payment gateway creates a new installment date of a recurring payment.
- Upon each new payment attempt, after a recurring payment installment has been refused.

  Requires the activation of the anticipated authorization option.

This rule is **disabled by default**.

1. Right-click **Instant Payment Notification URL when creating a recurring payment**.
2. Select **Manage the rule**.
3. Enter the **E-mail address(es) to notify in case of failure** field in the **General settings** section.

   To specify several e-mail addresses, separate them with a semi-colon.
4. Check the box **Automatic retry in case of failure** if you wish to authorize the gateway to automatically resend the notification in case of a failure (can be done up to 4 times).

   For more information, please see chapter *Automatic retry in case of failure* on page 36.
5. In the **Instant Payment Notification URL of the API form V1, V2** section, specify the URL of your page in the fields **URL to notify in TEST mode** and **URL to notify in PRODUCTION mode** if you wish to receive notifications in the API form format.
6. In the **REST API Instant Payment Notification URL** section, specify the URL of your page in the fields **Target URL of the IPN to notify in TEST mode** and **Target URL of the IPN to notify in PRODUCTION mode** if you are using the JavaScript client.
7. Save the changes.
8. Enable the rule by right-clicking **Instant Payment Notification URL when creating a recurring payment** and select **Enable the rule**.

10.5. Setting up a notification on batch authorization

If the shop has the **Anticipated authorization** option, it is necessary to enable the **Instant Payment Notification URL on batch authorization** rule in order to receive the final payment result.

This rule is **disabled by default**.

1. Right-click **Instant Payment Notification URL on batch authorization**.
2. Select **Manage the rule**.
3. Enter the **E-mail address(es) to notify in case of failure** field in the **General settings** section.
   To specify several e-mail addresses, separate them with a semi-colon.
4. Check the box **Automatic retry in case of failure** if you wish to authorize the gateway to automatically resend the notification in case of a failure (can be done up to 4 times).
   For more information, please see chapter *Automatic retry in case of failure* on page 36.
5. In the **Instant Payment Notification URL of the API form V1, V2** section, specify the URL of your page in the fields **URL to notify in TEST mode** and **URL to notify in PRODUCTION mode** if you wish to receive notifications in the API form format.
6. In the **REST API Instant Payment Notification URL** section, specify the URL of your page in the fields **Target URL of the IPN to notify in TEST mode** and **Target URL of the IPN to notify in PRODUCTION mode** if you are using the JavaScript client.
7. Save the changes.
8. Enable the rule by right-clicking **Instant Payment Notification URL on batch authorization** and select **Enable the rule**.

10.6. Automatic retry in case of failure

Automatic retry does not apply to notifications manually triggered via the Expert Back Office.

The merchant can enable a mechanism that allows the payment gateway to automatically return notifications when the merchant website is temporarily unavailable, **up to 4 times**.

A notification will be considered as failed if the HTTP code returned by the merchant site is not on the following list: **200**,**201**, **202**, **203**, **204**, **205**, **206**, **301**, **302**,**303**, **307**, **308**.

Call attempts are scheduled at fixed intervals every 15 minutes (00, 15, 30, 45).

After each failed attempt, a notification e-mail is sent to the e-mail address specified in the configuration of the notification rule in question.

In this case, the subject of the e-mail contains the number corresponding to the notification retry attempt. It is presented as `attempt #` followed by the attempt number.

- Example of an e-mail subject following a first notification failure at the end of payment:

```
[MODE TEST] My Shop - Tr. ref. 067925 / FAILURE during the call to your IPN URL
 [unsuccessful attempt #1]
```

- Example of an e-mail subject following a second failure:

```
[MODE TEST] My Shop - Tr. ref. 067925 / FAILURE during the call to your IPN URL
 [unsuccessful attempt #2]
```

- Example of an e-mail subject following a third failure:

```
[MODE TEST] My Shop - Tr. ref. 067925 / FAILURE during the call to your IPN URL
 [unsuccessful attempt #3]
```

- Example of an e-mail subject following the last failure:

```
[MODE TEST] My Shop - Tr. ref. 067925 / FAILURE during the call to your IPN URL
 [unsuccessful attempt #last]
```

  To notify the merchant website of the last notification attempt, the e-mail subject will contain the mention `attempt #last`.

During the automatic retry, certain details are not stored in the database or are modified.

Examples of fields not available/not registered in the database:

Field name	Description
vads_page_action	Completed operation.
vads_payment_config	Payment type (immediate or installment).
vads_action_mode	Acquisition mode for payment method data.

Examples of fields sent with different values:

Field name	New value
vads_url_check_src	Always set to **RETRY** in case of automatic retry.
vads_trans_status	The transaction status may vary between the initial call and the automatic retry (cancellation by the merchant, transaction capture at the bank, etc.).
vads_hash	The value of this field is regenerated with each call.
signature	The signature value depends on the different statuses that may vary between the initial call and the automatic retry.

These e-mails contain:

- the encountered problem,
- parts of analysis depending on the error,
- its consequences,
- instructions for manually triggering the notification from the Expert Back Office.

Note:

After the fourth attempt, it is still possible to retry the IPN URL **manually** via your Expert Back Office.

Warning, during the automatic retry, any manual call to the IPN URL will affect the number of automatic attempts:

- a successful manual call will stop automatic retry,
- a failed manual call will have no impact on the current automatic retry.

10.7. Configuring e-mails sent to the buyer

In the **E-mail sent to the buyer** tab:

1. Right-click the rule to be modified and select **Enable the rule**.
2. Right-click the rule again and select **Manage the rule**.
   The rule management wizard appears.
3. In the General settings section, you can customize the label of the rule.
4. To customize the e-mail content:
   a. Click **Buyer e-mail settings**.
   b. Select the template of the e-mail to apply.
   c. Select the language that you would like to update.
   d. Click **Customize default text values** if you wish to edit the body and the subject of the “default” e-mail message.
   e. Click on **Fields to include** to display the list of fields available for e-mail customization.
   f. Select the fields that you wish to include. A detailed summary of the request processing will be added to the body of the e-mail.

Note:

*To preview the changes, click **Preview the e-mail** at the bottom of the dialog box.*

5. In order to change the events that trigger the notification:
   a. Click the **Rule conditions** tab.
      A condition is composed of a variable, a comparison operator and a reference value.
      Example: "mode = TEST", "amount exceeding 1000". During the execution of a rule, the value of a variable is retrieved and compared to the reference value.
   b. Double-click on an existing condition to edit it.
   c. Click **Add** to create a new condition.
      All the conditions must be validated for the rule to be executed.
6. Click **Save**.

11. GENERATING A PAYMENT FORM

To generate a payment request, you must create an HTML form as follows:

```
<form method="POST" action="https://secure.lyra.com/vads-payment/">
 <input type="hidden" name="parameter1" value="value1" />
 <input type="hidden" name="parameter2" value="value2" />
 <input type="hidden" name="parameter3" value="value3" />
 <input type="hidden" name="signature" value="signature"/>
 <input type="submit" name="pay" value="Pay"/>
</form>
```

It contains:

- The following technical elements:
  - The `<form>` and `</form>` tags that allow to create an HTML form.
  - The `method="POST"` attribute that defines the method used for sending data.
  - The `action="https://secure.lyra.com/vads-payment/"` attribute that defines where to send the form data.
- Form data:
  - The shop ID.
  - Information about the payment depending on the use case.
  - Additional information depending on your needs.
  - The signature that ensures the integrity of the form.

  This data is added to the form by using the `<input>` tag:

  ```
  <input type="hidden" name="parameter1" value="value1" />
  ```

  For setting the `name` and `value` attributes, see the **Data dictionary** chapter also available in the online document archive.

  All the data in the form must be encoded in **UTF-8.**

  Special characters (accents, punctuation marks, etc.) will then be correctly interpreted by the payment gateway. Otherwise, the signature will be computed incorrectly and the form will be rejected.

- The **Pay** button to submit the data:

  ```
  <input type="submit" name="pay" value="Pay"/>
  ```


Different use cases are presented in the chapters below. They will allow you to adapt your payment form to your needs.

The following table lists the different formats that you can encounter when building your form.

Notation	Description
a	Alphabetic characters (from 'A' to 'Z' and from 'a' to 'z')
n	Numeric characters
s	Special characters
an	Alphanumeric characters
ans	Alphanumeric and special characters (except '<' and '>')
3	Fixed length of 3 characters
..12	Variable length up to 12 characters
json	JavaScript Object Notation. Object containing key/value pairs separated by commas. It starts with a left brace "`{`" and ends with a right brace "`}`". Each key / value pair contains the name of the key between double-quotes followed by "`:`", followed by a value. The name of the key must be alphanumeric. The value can be: • a chain of characters (in this case it must be framed by double-quotes) • a number • an object • a table • a boolean • empty Example: `{"name1":45,"name2":"value2", "name3":false}`
bool	Boolean. Can be populated with the **true** or **false** value.
enum	Characterizes a field with a complete list of values. The list of possible values is given in the field definition.
Enum list	List of values separated by a "`;`". The list of possible values is given in the field definition. Example: `vads_payment_cards=VISA;MASTERCARD`
map	List of key / value pairs separated by a "`;`". Each key / value pair contains the name of the key followed by "`=`", followed by a value. The value can be: • a chain of characters • a boolean • a json object • an xml object The list of possible values for each key/value pair is provided in the field definition. Example: `vads_theme_config=SIMPLIFIED_DISPLAY=true;RESPONSIVE_MODEL=Model_1`

11.1. Creating a 'Create a token without payment' form

Use case: creation of a token for making fast payments in the future.

Under PSD2, strong authentication is required when registering a card. The vads_threeds_mpi field is ignored and the CHALLENGE_MANDATE value is automatically applied.

1. Use all the fields presented in the table below to create your form.

Field name	Description	Format	Value
vads_page_action	Action to perform	enum	**REGISTER**
vads_ctx_mode	Operating mode.	enum	**TEST** or **PRODUCTION**
vads_cust_email	Buyer's e-mail address.	ans..150	E.g.: mail@example.com
vads_action_mode	Acquisition mode for payment method data.	enum	**INTERACTIVE**
vads_site_id	Shop ID	n8	E.g.: 12345678
vads_trans_date	Date and time of the payment form in UTC format.	n14	E.g.: 20190501130025
vads_version	Version of the exchange protocol.	string	**V2**

2. Use the **vads_identifier** field if you wish to generate the identifier of the token associated with the payment method.

   The token format must not be an..32. This format is reserved for tokens generated by the payment gateway.

   If you have activated the detection of token uniqueness, the value of the **vads_identifier** field transmitted in the form can be different from the one present in the notification if the payment method is already associated with another token.

3. Add optional fields according to your requirements (see chapter **Using additional features**).

4. Compute the value of the **signature** field using all the fields of your form starting with **vads_** (see chapter ***Computing the signature***).

The list of fields returned in the notification is described in the chapter *Creating a token without payment* on page 67.

11.2. Creating an 'Edit information associated with the token' form

Use case: update of bank information associated with a token.

Under PSD2, strong authentication is required when registering a card. The vads_threeds_mpi field is ignored and the CHALLENGE_MANDATE value is automatically applied.

1. Use all the fields presented in the table below to create your form.

Field name	Description	Format	Value
vads_page_action	Action to perform	enum	**REGISTER_UPDATE**
vads_ctx_mode	Operating mode.	enum	**TEST** or **PRODUCTION**
vads_cust_email	Buyer's e-mail address.	ans..150	E.g.: mail@example.com
vads_action_mode	Acquisition mode for payment method data.	enum	**INTERACTIVE**
vads_identifier	(unique) token associated with a payment method.	ans..50	E.g.: MyToken **Note**: two possible formats:

Field name	Description	Format	Value
			• **an32**: if the identifier is generated by the payment gateway • **ans..50**: if the identifier is generated by the merchant.
vads_site_id	Shop ID	n8	E.g.: 12345678
vads_trans_date	Date and time of the payment form in UTC format.	n14	E.g.: 20190501130025
vads_version	Version of the exchange protocol.	string	**V2**

2. Add optional fields according to your requirements (see chapter **Using additional features**).

3. Compute the value of the **signature** field using all the fields of your form starting with **vads_** (see chapter ***Computing the signature***).

The list of fields returned in the notification is described in the chapter *Updating token details* on page 70.

11.3. Creating a 'Create a token during a payment' form

Use case: creating a token during a payment.

Under PSD2, strong authentication is required when registering a card. The vads_threeds_mpi field is ignored and the CHALLENGE_MANDATE value is automatically applied.

1. Use all the fields presented in the table below to create your form.

Field name	Description	Format	Value
vads_page_action	Action to perform	enum	**REGISTER_PAY**
vads_amount	Payment amount (in the smallest currency unit)	n..12	E.g.: 3000 for 30,00 EUR
vads_ctx_mode	Operating mode.	enum	**TEST** or **PRODUCTION**
vads_currency	Code of the currency used for the payment.	n3	E.g.: 978 for euro (EUR)
vads_cust_email	Buyer's e-mail address.	ans..150	E.g.: mail@example.com
vads_action_mode	Acquisition mode for payment method data.	enum	**INTERACTIVE**
vads_payment_config	Payment type	enum	**SINGLE**
vads_site_id	Shop ID	n8	E.g.: 12345678
vads_trans_date	Date and time of the payment form in UTC format.	n14	E.g.: 20190501130025
vads_trans_id	Unique ID of a transaction.	n6	E.g.: 123456
vads_version	Version of the exchange protocol.	string	**V2**

2. Use the **vads_identifier** field if you wish to generate the identifier of the token associated with the payment method.

   The token format must not be an..32. This format is reserved for tokens generated by the payment gateway.

   If you have activated the detection of token uniqueness, the value of the **vads_identifier** field transmitted in the form can be different from the one present in the notification if the payment method is already associated with another token.

3. Add optional fields according to your requirements (see chapter **Using additional features**).

4. Compute the value of the **signature** field using all the fields of your form starting with **vads_** (see chapter ***Computing the signature***).

The list of fields returned in the notification is described in the chapter *Creating a token during a payment* on page 73.

11.4. Creating a 'Create a token when making a recurring payment' form

Use case: subscription to a recurring payment with token creation.

Under PSD2, strong authentication is required when registering a card. The vads_threeds_mpi field is ignored and the CHALLENGE_MANDATE value is automatically applied.

IMPORTANT

No payments will be made during the subscription. Only a verification request will be made in order to validate the payment method details.

The first payment will be made once the due date is reached, between 12 a.m. and 5 a.m.

If you want the buyer to make the first installment at the moment of creating the recurring payment, see the following chapter: *'Creation of a token when creating a recurring payment with payment' form* on page 47.

1. Use all the fields presented in the table below to create your form.

Field name	Description	Format	Value
vads_page_action	Action to perform	enum	**REGISTER_SUBSCRIBE**
vads_ctx_mode	Operating mode.	enum	**TEST** or **PRODUCTION**
vads_cust_email	Buyer's e-mail address.	ans..150	E.g.: mail@example.com
vads_action_ mode	Acquisition mode for payment method data.	enum	**INTERACTIVE**
vads_site_id	Shop ID	n8	E.g.: 12345678
vads_sub_amount	Installment amount (in the smallest currency unit). **The installment amount cannot be set to 0.**	n..12	E.g.: 3000 for 30,00 EUR
vads_sub_effect_date	Recurring payment start date.	n8	Attention, this date cannot be in the past. E.g.: 20210601
vads_sub_currency	Code of the currency used for the recurring payment.	n3	E.g.: 978 for euro (EUR)

vads_sub_desc	Rule for recurring payments to apply according to the iCalendar RFC5545 specification.
	If you request the creation of a recurring payment to debit the payment method holder daily (RRULE:FREQ=DAILY;INTERVAL and the requested effective date (vads_sub_effect_date) corresponds to the recurring payment creation date, then when the payment gateway processes this recurring payment the following morning (between midnight and 5 a.m.), 2 payments will be created:
	• The payment from the previous day (that corresponds to the effective date)
	• And the payment from the current day
	To avoid this, it is recommended to transmit an effective date the day
		string	The recurring payments can be made daily, weekly or monthly.
			It is possible to specify the number of the day or the month (e.g. "the 10th of the month", "every 3 months").
			Note: the string must not contain space characters.
			Examples:
			• In order to define a weekly recurring payment:
			`RRULE:FREQ=WEEKLY`
			• In order to define a recurring payment every two weeks, on the same day and every 7 days:
			`RRULE:FREQ=WEEKLY;INTERVAL=2`
			• To schedule installment payments on the last day of each month for 12 months:
			`RRULE:FREQ=MONTHLY;`
			`BYMONTHDAY=28,29,30,31;`
			`BYSETPOS=-1;COUNT=12`

Field name	Description	Format	Value
	after the recurring payment is created.		• To schedule installment payments on the 10th of each month for 12 months: `RRULE:FREQ=MONTHLY;COUNT=12;BYMONTHDAY=10`
vads_trans_date	Date and time of the payment form in UTC format.	n14	E.g.: 20190501130025
vads_version	Version of the exchange protocol.	string	**V2**

Note:

*The **vads_sub_effect_date** time value must not be in the past.*

2. Use the **vads_identifier** field if you wish to generate the identifier of the token associated with the payment method.

   The token format must not be an..32. This format is reserved for tokens generated by the payment gateway.

   If you have activated the detection of token uniqueness, the value of the **vads_identifier** field transmitted in the form can be different from the one present in the notification if the payment method is already associated with another token.

3. Add optional fields according to your requirements (see chapter **Using additional features**).

4. Compute the value of the **signature** field using all the fields of your form starting with **vads_** (see chapter ***Computing the signature***).

The list of fields returned in the notification is described in the chapter *Creating a token during a recurring payment* on page 77.

11.5. 'Creation of a token when creating a recurring payment with payment' form

Use case: payment and creation of a recurring payment with token creation.

Under PSD2, strong authentication is required when registering a card. The vads_threeds_mpi field is ignored and the CHALLENGE_MANDATE value is automatically applied.

1. Use all the fields presented in the table below to create your form.

Field name	Description	Format	Value
vads_page_action	Action to perform	enum	**REGISTER_PAY_SUBSCRIBE**
vads_amount	Payment amount (in the smallest currency unit)	n..12	E.g.: 3000 for 30,00 EUR
vads_ctx_mode	Operating mode.	enum	**TEST** or **PRODUCTION**
vads_currency	Code of the currency used for the payment.	n3	E.g.: 978 for euro (EUR)
vads_cust_email	Buyer's e-mail address.	ans..150	E.g.: mail@example.com
vads_action_mode	Acquisition mode for payment method data.	enum	**INTERACTIVE**
vads_payment_config	Payment type	enum	**SINGLE**
vads_site_id	Shop ID	n8	E.g.: 12345678
vads_sub_amount	Installment amount (in the smallest currency unit) **The installment amount cannot be set to 0.**	n..12	E.g.: 3000 for 30,00 EUR
vads_sub_currency	Code of the currency used for the recurring payment.	n3	E.g.: 978 for euro (EUR)

vads_sub_desc	Rule for recurring payments to apply according to the iCalendar RFC5545 specification.
	If you request the creation of a recurring payment to debit the payment method holder daily (RRULE:FREQ=DAILY;INTERVAL and the requested effective date (vads_sub_effect_date) corresponds to the recurring payment creation date, then when the payment gateway processes this recurring payment the following morning (between midnight and 5a.m.), 2 payments will be created:
	• The payment from the previous day (that corresponds to the effective date)
	• And the payment from the current day
	To avoid this, it is recommended to transmit an effective date the day after the recurring payment is created.
		string	The recurring payments can be made daily, weekly or monthly.
			It is possible to specify the number of the day or the month (e.g. "the 10th of the month", "every 3 months").
			Note: the string must not contain space characters.
			Examples:
			• In order to define a weekly recurring payment:
			`RRULE:FREQ=WEEKLY`
			• In order to define a recurring payment every two weeks, on the same day and every 7 days:
			`RRULE:FREQ=WEEKLY;INTERVAL=2`
			• To schedule installment payments on the last day of each month for 12 months:
			`RRULE:FREQ=MONTHLY;`
			`BYMONTHDAY=28,29,30,31;`
			`BYSETPOS=-1;COUNT=12`
			• To schedule installment payments on the 10th of each month for 12 months:
			`RRULE:FREQ=MONTHLY;`
			`COUNT=12;BYMONTHDAY=10`

Field name	Description	Format	Value
vads_sub_effect_date	Recurring payment start date.	n8	E.g.: 20210601
vads_trans_date	Date and time of the payment form in UTC format.	n14	E.g.: 20190501130025
vads_trans_id	Unique ID of a transaction.	n6	E.g.: 123456
vads_version	Version of the exchange protocol.	string	**V2**

Note:

*The **vads_sub_effect_date** time value must not be in the past.*

2. Use the **vads_identifier** field if you wish to generate the identifier of the token associated with the payment method.

   The token format must not be an..32. This format is reserved for tokens generated by the payment gateway.

   If you have activated the detection of token uniqueness, the value of the **vads_identifier** field transmitted in the form can be different from the one present in the notification if the payment method is already associated with another token.

3. Add optional fields according to your requirements (see chapter **Using additional features**).

4. Compute the value of the **signature** field using all the fields of your form starting with **vads_** (see chapter ***Computing the signature***).

The list of fields returned in the notification is described in the chapter *Creating a token during creation of a recurring payment with payment* on page 81.

11.6. Creating a 'Payment by token' form

Use case: one-click payment (using an existing and valid token).

In the context of the application of the PSD2, the CVV entry as well as cardholder authentication are required during a payment by token, as long as the buyer is present. The vads_threeds_mpi field is taken into account to allow for potential authentication without buyer interaction.

1. Use all the fields presented in the table below to create your form.

Field name	Description	Format	Value
vads_page_action	Action to perform	enum	**PAYMENT**
vads_amount	Payment amount (in the smallest currency unit)	n..12	E.g.: 3000 for 30,00 EUR
vads_ctx_mode	Operating mode.	enum	**TEST** or **PRODUCTION**
vads_currency	Code of the currency used for the payment.	n3	E.g.: 978 for euro (EUR)
vads_action_mode	Acquisition mode for payment method data.	enum	**INTERACTIVE**
vads_identifier	(unique) token associated with a payment method.	ans..50	E.g.: MyToken **Note**: two possible formats: • **an32**: if the identifier is generated by the payment gateway • **ans..50**: if the identifier is generated by the merchant.
vads_payment_config	Payment type	enum	**SINGLE**
vads_site_id	Shop ID	n8	E.g.: 12345678
vads_trans_date	Date and time of the payment form in UTC format.	n14	E.g.: 20190501130025
vads_trans_id	Unique ID of a transaction.	n6	E.g.: 123456
vads_version	Version of the exchange protocol.	string	**V2**

2. Add optional fields according to your requirements (see chapter **Using additional features**).

3. Compute the value of the **signature** field using all the fields of your form starting with **vads_** (see chapter ***Computing the signature***).

The list of fields returned in the notification is described in the chapter *Payment by token* on page 84.

11.7. Creating a 'Use a token to create a recurring payment' form

Use case: using an existing and valid token to create a recurring payment.

IMPORTANT

No payments will be made during the subscription.

The first payment will be made once the due date is reached, between 12 a.m. and 5 a.m.

1. Use all the fields presented in the table below to create your form.

Field name	Description	Format	Value
vads_page_action	Action to perform	enum	**SUBSCRIBE**
vads_ctx_mode	Operating mode.	enum	**TEST** or **PRODUCTION**
vads_action_mode	Acquisition mode for payment method data.	enum	**INTERACTIVE**
vads_identifier	(unique) token associated with a payment method.	ans..50	E.g.: MyToken **Note**: two possible formats: • **an32**: if the identifier is generated by the payment gateway • **ans..50**: if the identifier is generated by the merchant.
vads_site_id	Shop ID	n8	E.g.: 12345678
vads_sub_amount	Installment amount (in the smallest currency unit) **The installment amount cannot be set to 0.**	n..12	E.g.: 3000 for 30,00 EUR
vads_sub_effect_date	Recurring payment start date.	n8	E.g.: 20210601
vads_sub_currency	Code of the currency used for the recurring payment	n3	E.g.: 978 for euro (EUR)

vads_sub_desc	Rule for recurring payments to apply according to the iCalendar RFC5545 specification.
	If you request the creation of a recurring payment to debit the payment method holder daily (RRULE:FREQ=DAILY;INTERVAL and the requested effective date (vads_sub_effect_date) corresponds to the recurring payment creation date, then when the payment gateway processes this recurring payment the following morning (between midnight and 5a.m.), 2 payments will be created:
	• The payment from the previous day (that corresponds to the effective date)
	• And the payment from the current day
	To avoid this, it is recommended to transmit an effective date the day
		string	The recurring payments can be made daily, weekly or monthly.
			It is possible to specify the number of the day or the month (e.g. "the 10th of the month", "every 3 months").
			Note: the string must not contain space characters.
			Examples:
			• In order to define a weekly recurring payment:
			`RRULE:FREQ=WEEKLY`
			• In order to define a recurring payment every two weeks, on the same day and every 7 days:
			`RRULE:FREQ=WEEKLY;INTERVAL=2`
			• To schedule installment payments on the last day of each month for 12 months:
			`RRULE:FREQ=MONTHLY; BYMONTHDAY=28,29,30,31; BYSETPOS=-1;COUNT=12`

Field name	Description	Format	Value
	after the recurring payment is created.		• To schedule installment payments on the 10th of each month for 12 months: `RRULE:FREQ=MONTHLY;COUNT=12;BYMONTHDAY=10`
vads_trans_date	Date and time of the payment form in UTC format.	n14	E.g.: 20190501130025
vads_version	Version of the exchange protocol.	string	**V2**

Note:

*The **vads_sub_effect_date** time value must not be in the past.*

2. Add optional fields according to your requirements (see chapter **Using additional features**).

3. Compute the value of the **signature** field using all the fields of your form starting with **vads_** (see chapter ***Computing the signature***).

The list of fields returned in the notification is described in the chapter *Creating a recurring payment* on page 87.

11.8. Creating a 'Payment with option for the cardholder to create a token' form

Use case: offer the possibility to create a token during a payment.

Under PSD2, strong authentication is required when registering a card. The vads_threeds_mpi field is ignored and the CHALLENGE_MANDATE value is automatically applied.

1. Use all the fields presented in the table below to create your form.

Field name	Description	Format	Value
vads_page_action	Action to perform	enum	**ASK_REGISTER_PAY**
vads_amount	Payment amount (in the smallest currency unit)	n..12	E.g.: 3000 for 30,00 EUR
vads_ctx_mode	Operating mode.	enum	**TEST** or **PRODUCTION**
vads_currency	Code of the currency used for the payment.	n3	E.g.: 978 for euro (EUR)
vads_cust_email	Buyer's e-mail address.	ans..150	E.g.: mail@example.com
vads_action_mode	Acquisition mode for payment method data.	enum	**INTERACTIVE**
vads_payment_config	Payment type	enum	**SINGLE**
vads_site_id	Shop ID	n8	E.g.: 12345678
vads_trans_date	Date and time of the payment form in UTC format.	n14	E.g.: 20190501130025
vads_trans_id	Unique ID of a transaction.	n6	E.g.: 123456
vads_version	Version of the exchange protocol.	string	**V2**

2. Use the **vads_identifier** field if you wish to generate the identifier of the token associated with the payment method.

   The token format must not be an..32. This format is reserved for tokens generated by the payment gateway.

   If you have activated the detection of token uniqueness, the value of the **vads_identifier** field transmitted in the form can be different from the one present in the notification if the payment method is already associated with another token.

3. Add optional fields according to your requirements (see chapter **Using additional features**).

4. Compute the value of the **signature** field using all the fields of your form starting with **vads_** (see chapter ***Computing the signature***).

The list of fields returned in the notification is described in the chapter *Payment with the token creation option for the cardholder* on page 89.

12. USING ADDITIONAL FEATURES

To obtain a custom form adapted to your needs, you can use additional optional features from the list below:

IMPORTANT
Other useful features are presented in the *Hosted Payment Page Implementation Guide*.

- Defining the currency for creating or updating a token
- Defining a different amount for the first n installments

These features are described in the following chapters. They will allow you to easily build your payment form.

12.1. Defining a different amount for the first n installments

You wish to set up a recurring payment for which the amount of the first installment(s) would differ from the ones configured by the **vads_sub_amount** field.

Example: define a recurring payment with the first 3 installments equal to 25,00 EUR and the remaining installments equal to 30,00 EUR.

To do this:

1. Use the fields required for your use case to create your payment form.
2. Use the fields below:

Field name	Description	Value
vads_sub_init_amount_number	Number of installments to which will be applied the amount defined by **vads_sub_init_amount**	3
vads_sub_init_amount	Amount of the first installments. The number of the first installments is defined by **vads_sub_init_amount_number**	2500
vads_sub_amount	Amount of each installment except the ones that will be eventually defined by **vads_sub_init_amount_number**	3000
vads_sub_currency	Currency used for all of the installments.	E.g.: 978 for euro (EUR)

Notes:

- The **vads_sub_init_amount** and **vads_sub_amount** fields cannot be set to 0.
- To define a recurring payment where the 3 first months are free, all you need to do is set the start date (**vads_sub_effect_date**) to 3 months later.

Example of a payment form:

```
<form method="POST" action="https://secure.lyra.com/vads-payment/">
<input type="hidden" name="vads_action_mode" value="INTERACTIVE" />
<input type="hidden" name="vads_capture_delay" value="0" />
<input type="hidden" name="vads_ctx_mode" value="TEST" />
<input type="hidden" name="vads_currency" value="978" />
<input type="hidden" name="vads_cust_country" value="FR" />
<input type="hidden" name="vads_cust_email" value="example@gmail.com" />
<input type="hidden" name="vads_cust_first_name" value="John" />
<input type="hidden" name="vads_cust_last_name" value="Smith" />
<input type="hidden" name="vads_cust_title" value="Mr" />
<input type="hidden" name="vads_page_action" value="REGISTER_SUBSCRIBE" />
<input type="hidden" name="vads_payment_config" value="SINGLE" />
<input type="hidden" name="vads_site_id" value="91335531" />
<input type="hidden" name="vads_trans_date" value="20190716080441" />
<input type="hidden" name="vads_trans_id" value="362812" />
<input type="hidden" name="vads_validation_mode" value="0" />
<input type="hidden" name="vads_sub_currency" value="978" />
<input type="hidden" name="vads_sub_init_amount_number" value="3" />
<input type="hidden" name="vads_sub_init_amount" value="2500" />
<input type="hidden" name="vads_sub_amount" value="3000" />
<input type="hidden" name="vads_version" value="V2" />
<input type="hidden" name="signature" value="86b2a17b9a5fcefb6c0120c57b25ec86ad1704ee"/>
<input type="submit" name="pay" value="Pay"/>
</form>
```

12.2. Defining the currency for creating or updating a token

If:

- you have a MID that supports several currencies,

- you have several shops,
- your shops are all linked to the same MID,
- each shop generates payments in a different currency,

  (e.g.: US dollar for the first shop, Euro for the second shop)

it is possible that the currency used when creating or updating a token is not supported by the shop.

By default, the payment gateway uses alphabetical order when selecting the currency for performing the necessary checks with the payment method issuer.

In order to avoid IPN processing errors, you can transmit the information about the currency to use via the form.

Note:

It will always be possible to use the token for making payments in any currency supported by the agreement.

Field name	Description	Format	Value
vads_currency	Numeric currency code to be used for the payment, in compliance with the ISO 4217 standard (numeric code).	n3	E.g.: 978 for euro (EUR)

13. COMPUTING THE SIGNATURE

To be able to compute the signature, you must have:

- all the fields that start with **vads_**
- the signature algorithm chosen in the shop configuration
- the **key**

The value of the key is available in your Expert Back Office via **Settings** > **Shop** > **Keys** tab.

The signature algorithm is defined in your Expert Back Office via **Settings** > **Shop** > **Configuration** tab.

For maximum security, it is recommended to use HMAC-SHA-256 algorithm and an alphanumeric key.

The use of SHA-1 algorithm is deprecated but maintained for compliance reasons.

To compute the signature:

1. Sort the fields that start with **vads_** alphabetically.
2. Make sure that all the fields are encoded in UTF-8.
3. Concatenate the values of these fields separating them with the "**+**" character.
4. Concatenate the result with the test or production key separating them with a "**+**".
5. According to the signature algorithm defined in your shop configuration:
   a. If your shop is configured to use "SHA-1", apply the **SHA-1** hash function to the chain obtained during the previous step. **Deprecated.**
   b. If your shop is configured to use "HMAC-SHA-256", compute and encode in Base64 format the message signature using the **HMAC-SHA-256** algorithm with the following parameters:
      - the SHA-256 hash function,
      - the test or production key (depending on the value of the **vads_ctx_mode** field) as a shared key,
      - the result of the previous step as the message to authenticate.
6. Save the result of the previous step in the **signature** field.

Example of parameters sent to the payment gateway:

```
<form method="POST" action="https://secure.lyra.com/vads-payment/">
<input type="hidden" name="vads_action_mode" value="INTERACTIVE" />
<input type="hidden" name="vads_amount" value="5124" />
<input type="hidden" name="vads_ctx_mode" value="TEST" />
<input type="hidden" name="vads_currency" value="978" />
<input type="hidden" name="vads_page_action" value="PAYMENT" />
<input type="hidden" name="vads_payment_config" value="SINGLE" />
<input type="hidden" name="vads_site_id" value="12345678" />
<input type="hidden" name="vads_trans_date" value="20170129130025" />
<input type="hidden" name="vads_trans_id" value="123456" />
<input type="hidden" name="vads_version" value="V2" />
<input type="hidden" name="signature" value="ycA5Do5tNvsnKdc/eP1bj2xa19z9q3iWPy9/rpesfS0="/>

<input type="submit" name="pay" value="Pay"/>
</form>
```

This sample form is analyzed as follows:

1. The fields whose names start with **vads_** are sorted **alphabetically**:
   - vads_action_mode
   - vads_amount
   - vads_ctx_mode
   - vads_currency
   - vads_page_action
   - vads_payment_config
   - vads_site_id
   - vads_trans_date
   - vads_trans_id
   - vads_version
2. The values of these fields are concatenated using the "**+**" character:

   ```
   INTERACTIVE+5124+TEST+978+PAYMENT+SINGLE+12345678+20170129130025+123456+V2
   ```

3. The value of the test key is added at the end of the chain and separated with the "**+**" character. In this example, the test key is **1122334455667788**

   ```
   INTERACTIVE+5124+TEST+978+PAYMENT+SINGLE+12345678+20170129130025+123456+V2+1122334455667788
   ```

4. If you use the SHA-1 algorithm, apply it to the obtained chain.

   The result that must be transmitted in the signature field is: **59c96b34c74b9375c332b0b6a32e6deeec87de2b**

5. If your shop is configured to use "HMAC-SHA-256", compute and encode in Base64 format the message signature using the **HMAC-SHA-256** algorithm with the following parameters:
   - the SHA-256 hash function,
   - the test or production key (depending on the value of the **vads_ctx_mode** field) as a shared key,
   - the result of the previous step as the message to authenticate.

   The result that must be transmitted in the signature field is:

   ycA5Do5tNvsnKdc/eP1bj2xa19z9q3iWPy9/rpesfS0=

14. SENDING THE PAYMENT REQUEST

The buyer will be able to finalize his/her purchase once he/she is redirected to the payment page.

The buyer's browser must transmit the payment form data.

14.1. Redirecting the buyer to the payment page

The URL of the payment gateway is:

https://secure.lyra.com/vads-payment/

Example of parameters sent to the payment gateway:

```
<form method="POST" action="https://secure.lyra.com/vads-payment/">
<input type="hidden" name="vads_action_mode" value="INTERACTIVE" />
<input type="hidden" name="vads_amount" value="2990" />
<input type="hidden" name="vads_ctx_mode" value="TEST" />
<input type="hidden" name="vads_currency" value="978" />
<input type="hidden" name="vads_cust_country" value="FR" />
<input type="hidden" name="vads_cust_email" value="me@example.com" />
<input type="hidden" name="vads_order_id" value="CMD012859" />
<input type="hidden" name="vads_page_action" value="REGISTER_PAY" />
<input type="hidden" name="vads_payment_config" value="SINGLE" />
<input type="hidden" name="vads_site_id" value="12345678" />
<input type="hidden" name="vads_trans_date" value="20200426101407" />
<input type="hidden" name="vads_trans_id" value="x6Z41p" />
<input type="hidden" name="vads_version" value="V2" />
<input type="hidden" name="signature" value="NM25DPLKEbtGEHCDHn8MBT4ki6aJI/ODaWhCzCnAfvY="/>
<input type="submit" name="pay" value="Pay"/>
</form>
```

14.2. Processing errors

If the payment gateway detects an error while receiving the form, an error message will appear and the buyer will not be able to proceed to the payment.

In TEST mode

The message indicates the source of the error and provides a link to the error code description to help you fix it.

In PRODUCTION mode

The message simply indicates to the buyer that a technical problem has occurred.

In both cases the merchant receives a notification e-mail.

It contains:

- the source of the error,
- a link to possible causes to facilitate its analysis,
- all the fields of the form.

The e-mail is sent to the company administrator.

If you wish to change this address or add an additional address, contact the Middle Office.

You can also create a personalized notification rule to receive this e-mail at another address.

To do this:

1. Sign in to your Expert Back Office:

   https://secure.lyra.com/portal/
2. Open the **Settings** > **Notification rules** menu.
3. Select **Advanced notification**.
4. Select the type of **E-mail sent to the merchant** notification.
5. Click **Next**.
6. Select the trigger event **Invalid payment form**.
7. In the **General settings**, fill in the fields:
   - **Rule reference**
   - **E-mail address to notify**
8. Click **Create**.

A description of the error codes with their possible causes is available on our website

https://docs.lyra.com/fr/collect/error-code/error-00.html

Other messages may appear during the payment process.

Here is a list of the most frequent messages:

Message	Description
Your payment request has been declined by your financial institution.	• The buyer's bank has rejected the authorization or information request. • The risk assessment rules have triggered the rejection of the transaction.
Your registration request has been declined by your financial institution.	• The buyer's bank has rejected the authorization or information request. • The risk assessment rules have triggered the rejection of the transaction.
This payment order is expired. Please contact your shop.	The buyer clicked on the payment link after the payment order expiration date.
This payment order has already been paid.	The buyer clicked on the payment link one more time after having already made the payment.
An error occurred during the payment request, the merchant website has been informed of the impossibility to finalize the transaction.	The payment form has been rejected. The shop administrator has received an e-mail with the details about the origin of the error.
The transaction has already been made.	The merchant website sends a transaction identifier that has already been used for another transaction (accepted or rejected). The transaction identifier must be unique within the same day (00:00:00 at 23:59:59 UTC).
Sorry, you have been disconnected due to a long period of inactivity.	• The buyer attempts to validate the card number while the payment session is expired. The session is open for about 10 minutes. • The merchant website sends a transaction identifier that has already been used but that did not result in a transaction (e.g. abandoned payment). The transaction identifier must be unique within the same day (00:00:00 at 23:59:59 UTC).
Cookies are blocked by your browser. Make sure you authorize them before retrying the operation.	The buyer has disabled cookies in his or her browser. Cookies are necessary for the payment to be processed correctly.

14.3. Managing timeouts

Payment session concept

A "payment session" is the time spent by a buyer on the payment page.

The payment session begins as soon as the payment gateway receives the payment form.

The delay of payment session is 10 minutes (except for certain payment methods).

This delay is:

- **sufficient** to enable each buyer to make his or her payment
- **by the deadline**: it is not reset to every action of the user
- **non-modifiable**: it is fixed by the payment gateway because of technical constraints.

After this delay, the payment session times out and the session data is purged.

Expiration of the payment session

In some cases the payment session will expire while the buyer has not completed the payment.

Most frequent cases:

1. Once redirected to the payment page for example, the buyer realizes that it is time to go to lunch.

   An hour later, the buyer decides to continue his or her payment and clicks on the logo corresponding to his or her payment method.

   The buyer's payment session has already expired, the payment gateway displays an error message indicating that it was disconnected due to an extended period of inactivity.

   The buyer then has the opportunity to click a button to return to the merchant website.

   The return to the shop is done via the URL specified by the merchant:

   - in the *vads_url_return* field transmitted in the payment form
   - in the "Return URL to the merchant website" field in the buyer's Expert Back Office, no *vads_url_return* field is transmitted in his or her payment form
2. Once redirected to the payment page, the buyer closes the browser (by mistake or because he or she no longer wants to make the payment).

Notification in case of session expiration

It is possible to notify the merchant website in case of expiration of the payment session.

To do so, the merchant must configure and enable the ***Instant Payment Notification URL on cancellation*** notification rule (see chapter *Setting up notifications in case of abandoned or canceled payments* on page 32).

15. IMPLEMENTING THE IPN

The script must include at least the following steps:

- Retrieve the field list sent with the POST response
- Compute the signature taking into account the received data
- Compare the computed signature with the received signature
- Analyze the nature of the notification
- Retrieve the payment result

The script may check the order status (or any information of your choice) to see if it has not been already updated.

Once these steps are completed, the script can update the database (new order status, stock update, registration of payment information, etc.).

In order to facilitate support and diagnosis by the merchant in the event of a notification error, we recommend to write messages that will allow you to know at which stage of processing the error occurred.

The gateway reads and stores the first 256 bytes of the HTTP response.

You can write messages throughout the processing. Here are some examples of messages that you can use:

Message	Use case
Data received.	Message to display when retrieving data. Allows to confirm that the notification has been received by the merchant website.
POST is empty.	Message to display when retrieving data. Allows to bring out a possible redirection that would have caused the parameters posted by the payment gateway to be lost.
An error occurred while computing the signature.	Message to be displayed when the verification of the response signature has failed.
Order successfully updated.	Message to be displayed at the end of the file once your processing has been successfully completed.
An error occurred while updating the order.	Message to be displayed at the end of the file if an error occurred during your processing.

15.1. Preparing your environment

The notifications of Instant Payment Notification URL call type are the most important as they represent the only reliable way for the merchant website to obtain the payment result.

It is therefore necessary to make sure the notifications function properly.

Here are some guidelines:

- In order for the dialog between the payment gateway and your merchant website to work, you must make sure, together with your technical teams, that the **194.50.38.0/24** IP address range is authorized on the various devices within your system (firewalls, apache server, proxy server, etc.).

  Notifications are sent from an IP address in the 194.50.38.0/24 range **in Test and Production modes**.

- Using redirection leads to losing data presented in POST.

  This is the case if there is a configuration on your devices or on the side of your host that redirects the URLs of "http://**www.**example.com" type to "http://example.com" or "**http**://example.com" to "**https**://example.com".

- HTML must not be visible on the page. Access to images or CSS slows down the exchange between the payment gateway and the merchant website.

- Avoid integrating time-consuming tasks, such as PDF invoice generation or sending e-mails in your script.

  The processing time has a direct influence on the time it takes to display the payment summary page.

  The longer the processing of the notification, the greater the delay for displaying the page. After 35 seconds, the payment gateway considers that the call has failed (timeout).

- If your page is only accessible in https, test your URL on the Qualys SSL Labs website (*https://www.ssllabs.com/ssltest/* and, if necessary, change your configuration in order to obtain the A score.

  Your SSL certificate must be signed by a certification authority known and recognized on the market.

- Make sure that you use the latest version of the TLS protocol in order to maintain a high level of security.

15.2. Retrieving data returned in the response

The data returned in the response depends on the parameters sent in the payment request, the payment type, the settings of your shop and the notification format.

The data is always sent by the payment gateway using the **POST** method.

The first step consists in retrieving the contents received via the POST method.

Examples:

- In PHP, data is stored in the super global variable **$_POST**,
- In ASP.NET (C#), you must use the **Form** property of the **HttpRequest** class,
- In Java, you must use the **getParameter** method of the **HttpServletRequest** interface.

The response consists of a field list. Each field contains a response value. The field list can be updated.

The script will have to create a loop to retrieve all the transmitted fields.

It is recommended to test the presence of the **vads_hash** field, which is only present during a notification.

```
if (empty ($_POST)){
 echo 'POST is empty';

}else{
 echo 'Data Received ';
 if (isset($_POST['vads_hash'])){

  echo 'Form API notification detected';
  //Signature computation
  //Signature verification
  //Order Update
 }
}
```


15.3. Computing the IPN signature

The signature is computed by following the same logic as for creating the payment request.

IMPORTANT

The data submitted by the payment gateway is encoded in UTF-8. Any alteration of received data will result in signature computation error.

You must compute the signature with the fields received in the notification and not the ones that you transmitted in the payment request.

1. Take all the fields whose name starts with **vads_**.
2. Sort these fields alphabetically.
3. Concatenate the values of these fields separating them with the "**+**" character.
4. Concatenate the result with the test or production key separating them with a "**+**".
5. According to the signature algorithm defined in your shop configuration:
   a. If your shop is configured to use "SHA-1", apply the **SHA-1** hash function to the chain obtained during the previous step. **Deprecated.**
   b. If your shop is configured to use "HMAC-SHA-256", compute and encode in Base64 format the message signature using the **HMAC-SHA-256** algorithm with the following parameters:
      - the SHA-256 hash function,
      - the test or production key (depending on the value of the **vads_ctx_mode** field) as a shared key,
      - the result of the previous step as the message to authenticate.

Examples in PHP:

```
function getSignature ($params,$key)
{
    /**
     *Function that computes the signature.
     * $params: table containing the fields received in the IPN.
     * $key : TEST or PRODUCTION key
     */
    //Initialization of the variable that will contain the string to encrypt
    $signature_contents = "";

    //Sorting fields alphabetically
    ksort($params);
    foreach($params as $name=>$value){

        //Recovery of vads_ fields
        if (substr($name,0,5)=='vads_'){

            //Concatenation with "+"
            $signature_contents .= $value."+";
         }
    }
    //Adding the key at the end
    $signature_contents .= $key;

    //Encoding base64 encoded chain with HMAC-SHA-256 algorithm
    $sign = base64_encode(hash_hmac('sha256',$signature_contents, $key, true));
    return $sign;
 }
```


15.4. Comparing signatures

To ensure the integrity of the response, you must compare the signature contained in the IPN with the value computed in the previous step.

IMPORTANT
You should not compare the signature of the IPN with the signature that you transmitted in your payment request.

If the signatures match

- You may consider the response as safe and proceed with the analysis.
- Otherwise, the script will have to raise an exception and notify the merchant about the anomaly.

Example in PHP:

```
if ($_POST['signature'] == $sign){

    //Processing data

}else{
    throw new Exception('An error occurred while computing the signature');
}
```

The signatures may not match in case of:

- an implementation error (error in your calculation, problem with UTF-8 encoding, etc.),
- an error in the value of the key or in the **vads_ctx_mode** field (frequent issue when shifting to production mode),
- a data corruption attempt.

15.5. Analyze the nature of the notification

The **vads_url_check_src** field allows to differentiate the notifications based on their triggering event:

- token creation (with or without the subscription of a recurring payment),
- installment payment,
- new notification sent by the merchant via the Expert Back Office.

It specifies the applied notification rule:

Value	Applied rule
PAY	The PAY value is sent in the following cases: • registration request for a mandate or a token (REGISTER) • registration request for a mandate or a token when subscribing to a recurring payment (REGISTER_SUBSCRIBE) • immediate payment (or first installment payment of a recurring payment) • payment abandoned or canceled by the buyer Only if the merchant has configured the **Instant Payment Notification URL on cancellation** rule.
BO	Execution of the notification via the Expert Back Office (right-click a transaction > **Send the Instant Payment Notification**). Check whether the vads_recurrence_number field is present: • if yes, the notification concerns the result of a recurring payment (retry of a **REC** type notification), • if not, the notification concerns a notification at the end of payment.
BATCH	The BATCH value is sent in case of an update of a transaction status after its synchronization on the acquirer side. This is the case of payments with redirection to the acquirer. Only if the merchant has configured the rule **Instant Payment Notification URL on batch change**.
BATCH_AUTO	The BATCH_AUTO value is sent in the following cases: • payment deferred for more than 7 days • installments of a recurring payment (except the first one) Only if the merchant has configured the **Instant Payment Notification URL on batch authorization** rule. The notification is sent with the authorization request for payments with "Waiting for authorization" status.
REC	The REC value is sent only for recurring payments if the merchant has configured the **Instant Payment Notification URL when creating recurring payments** rule. See the *Installment payment on page 92* chapter for more information on the sent data.
MERCH_BO	The MERCH_BO value is sent: • during operation made via the Expert Back Office (refund, cancellation, modification, validation, duplication, creation and/or update of token), only if the merchant has configured the following notification rule: **Instant Payment Notification URL on an operation coming from the Back Office**
RETRY	Automatic retry of the IPN. Check whether the vads_recurrence_number field is present: • if yes, the notification concerns the result of a recurring payment (retry of a **REC** type notification), • if not, the notification concerns a notification at the end of payment.

Table 1: Values associated with the vads_url_check_src field

After checking its value, the script can process differently depending on the nature of the notification.

For example:

If **vads_url_check_src** is set to **PAY** or **BATCH_AUTO**, the script will update the order status, etc.

If **vads_url_check_src** is set to **REC**, the script will retrieve the recurring payment reference and will increment the number of the expired installment payments in case the payment has been accepted, etc.

In the context of a recurring payment (from a REGISTER_SUBSCRIBE), the payment gateway notifies the creditor (merchant) when creating each transaction.

15.6. Processing the response data

Creating a token without payment

In order to understand the result, analyze the following fields:

Field name	Description
vads_page_action	Completed action. The returned value is **REGISTER**.
vads_identifier_status	Token creation status. Possible values are: • **CREATED**: the request for authorization or information, if supported by the acquirer is accepted. The token has been successfully created and gets displayed in the Expert Back Office. • **NOT_CREATED**: the authorization or information request has been rejected. The token is not created. • **ABANDONED**: the action has been abandoned by the buyer. The token is not created.
vads_identifier	Payment token. The returned value is: • either the value transmitted in the request, regardless of the result of token creation, even in case of abandoned action, • or the value generated by the payment gateway, if the field is not transmitted in the request and the token has been successfully created (vads_identifier_status=CREATED). Note If you have activated the token uniqueness check and the payment method is already registered with another token, then this token will be returned. The vads_identifier field will not be returned: • if it has not been transmitted in the request and the buyer abandons the action (vads_identifier_status=ABANDONED),

Field name	Description
	• if it has not been transmitted in the request and the token has not been created (vads_identifier_status=NOT_CREATED).
vads_identifier_previously_registered	Present only if both conditions are true: • You have enabled the token uniqueness check. • The used payment method is already registered with another token.
vads_cust_email	Buyer's e-mail address transmitted in the request.
vads_site_id	Shop ID The returned value is the same as the one submitted in the form.
vads_ctx_mode	Operating mode. The returned value (**TEST** or **PRODUCTION**) is the same as the one submitted in the form.

A "create a token without a payment" request triggers the creation of a VERIFICATION transaction, visible in the Expert Back Office.

The purpose of this transaction is to help the merchant, via their Back Office, understand the reasons for refusal of token creation.

Here are its characteristics:

Field name	Description
vads_operation_type	Transaction type Its value is **VERIFICATION**.
vads_trans_status	Status of the transaction. Possible values are: • **ACCEPTED** The authorization or information request has been accepted. The token has been created and is visible in the Expert Back Office. • **REFUSED** The authorization or information request has been rejected. The token is not created.
vads_occurrence_type	Transaction occurrence type. Its value is **SINGLE**.
vads_amount	Possible values: • **0** if the acquirer supports information requests, • **100** otherwise.
vads_trans_id	Transaction identifier. Its value is generated by the payment gateway.
vads_trans_uuid	Unique transaction ID. Its value is generated by the payment gateway.
vads_auth_mode	Type of request made via authorization servers. Its value is **MARK**.
vads_auth_number	Authorization number returned by the authorization server. Empty if the authorization has failed.
vads_auth_result	Return code of the authorization request returned by the issuing bank. See chapter ***Managing the return codes of the authorization request***. Empty in case of an error prior to the authorization.
vads_risk_assessment_result	List of actions made with the transaction, following the activation of the advanced risk assessment rules. **The rules related to the amount, or whose action is "Validate manually", do not apply in case of a VERIFICATION type transaction**. Possible values are: • **ENABLE_3DS**: 3D Secure enabled. • **DISABLE_3DS**: 3D Secure disabled.

Field name	Description
	• **REFUSE**: The token creation request has been refused. • **RUN_RISK_ANALYSIS**: Result of the external risk analyzer. See the description of the **vads_risk_analysis_result** field for more information. • **INFORM**: A warning message appears. The merchant is notified that a risk has been identified via one or several notification center rules.

Details of the used payment method:

Field name	Note
vads_acquirer_network	Acquirer network code.
vads_bank_code	Code of the issuing bank.
vads_bank_label	Name of the issuing bank of the payment card.
vads_bank_product	Product code of the used card.
vads_card_brand	Used payment method. See chapter **Compatible payment methods** to see the list of possible values.
vads_card_country	Country code of the card used in compliance with the ISO 3166 standard.
vads_card_number	Truncated/masked card number or IBAN and BIC used for the payment separated by "_" in case of a one-off debit payment. Since the BIC is optional, it may be left out. .
vads_expiry_month	Expiry month of the used card.
vads_expiry_year	Expiry year of the used card.

Authentication details of the cardholder:

Field name	Note
vads_threeds_auth_type	Cardholder authentication type. Strong cardholder authentication is mandatory when registering a card. The field will therefore always be populated with **CHALLENGE**.
vads_threeds_enrolled	Enrollment status of the buyer to the 3D Secure program. Possible values are: • **Y**: Authentication available. • **N**: Authentication not available. • **U**: Enrollment status to the 3D Secure program unknown. • empty: Incomplete 3DS authentication process (3DS disabled in the request, unregistered merchant or payment method not eligible for 3DS).
vads_threeds_status	3D Secure authentication result. Possible values are: • **Y**: Cardholder authentication success. • **N**: Cardholder authentication error. • **U**: Authentication impossible. • **A**: Authentication attempted but not completed. • empty: Unauthorized 3DS authentication (3DS disabled in the request, unregistered cardholder or payment method not eligible for 3DS).

The optional fields transmitted in the request are returned in the response with unmodified values.

Updating token details

In order to understand the result, analyze the following fields:

Field name	Description
vads_page_action	Completed action. The returned value is **REGISTER_UPDATE**.
vads_identifier_status	Token update status. Possible values are: • **UPDATED**: the token has been successfully updated. • **NOT_UPDATED**: the token has not been updated. • **ABANDONED**: the action has been abandoned by the buyer. The token has not been created, and therefore cannot be viewed in the Expert Back Office.
vads_identifier	Token ID to be updated. The returned value is the same as the one submitted in the form.
vads_cust_email	Buyer's e-mail address transmitted in the request.
vads_site_id	Shop ID The returned value is the same as the one submitted in the form.
vads_ctx_mode	Operating mode. The returned value (**TEST** or **PRODUCTION**) is the same as the one submitted in the form.

A "token update" request results in the creation of a VERIFICATION transaction, visible in the Expert Back Office.

The purpose of this transaction is to help the merchant, via their Back Office, understand the reasons for refusal of token creation.

Here are its characteristics:

Field name	Description
vads_operation_type	Transaction type Its value is **VERIFICATION**.
vads_trans_status	Status of the transaction. Possible values are: • **ACCEPTED** The authorization or information request has been accepted. The token has been created and is visible in the Expert Back Office. • **REFUSED** The authorization or information request has been rejected. The token is not created.
vads_occurrence_type	Transaction occurrence type. Its value is **SINGLE**.
vads_amount	Possible values: • **0** if the acquirer supports information requests, • **100** otherwise.
vads_trans_id	Transaction identifier. Its value is generated by the payment gateway.
vads_trans_uuid	Unique transaction ID. Its value is generated by the payment gateway.
vads_auth_mode	Type of request made via authorization servers. Its value is **MARK**.
vads_auth_number	Authorization number returned by the authorization server. Empty if the authorization has failed.

Field name	Description
vads_auth_result	Return code of the authorization request returned by the issuing bank. See chapter ***Managing the return codes of the authorization request***. Empty in case of an error prior to the authorization.
vads_risk_assessment_result	List of actions made with the transaction, following the activation of the advanced risk assessment rules. **The rules related to the amount, or whose action is "Validate manually", do not apply in case of a VERIFICATION type transaction**. Possible values are: • **ENABLE_3DS**: 3D Secure enabled. • **DISABLE_3DS**: 3D Secure disabled. • **REFUSE**: The token creation request has been refused. • **RUN_RISK_ANALYSIS**: Result of the external risk analyzer. See the description of the **vads_risk_analysis_result** field for more information. • **INFORM**: A warning message appears. The merchant is notified that a risk has been identified via one or several notification center rules.

Details of the used payment method:

Field name	Note
vads_acquirer_network	Acquirer network code.
vads_bank_code	Code of the issuing bank.
vads_bank_label	Name of the issuing bank of the payment card.
vads_bank_product	Product code of the used card.
vads_card_brand	Used payment method. See chapter **Compatible payment methods** to see the list of possible values.
vads_card_country	Country code of the card used in compliance with the ISO 3166 standard.
vads_card_number	Truncated/masked card number or IBAN and BIC used for the payment separated by "_" in case of a one-off debit payment. Since the BIC is optional, it may be left out. .
vads_expiry_month	Expiry month of the used card.
vads_expiry_year	Expiry year of the used card.

Authentication details of the cardholder:

Field name	Note
vads_threeds_auth_type	Cardholder authentication type. Strong cardholder authentication is mandatory when registering a card. The field will therefore always be populated with **CHALLENGE**.
vads_threeds_enrolled	Enrollment status of the buyer to the 3D Secure program. Possible values are: • **Y**: Authentication available. • **N**: Authentication not available. • **U**: Enrollment status to the 3D Secure program unknown. • empty: Incomplete 3DS authentication process (3DS disabled in the request, unregistered merchant or payment method not eligible for 3DS).
vads_threeds_status	3D Secure authentication result. Possible values are: • **Y**: Cardholder authentication success. • **N**: Cardholder authentication error.

Field name	Note
	• **U**: Authentication impossible. • **A**: Authentication attempted but not completed. • empty: Unauthorized 3DS authentication (3DS disabled in the request, unregistered cardholder or payment method not eligible for 3DS).

The optional fields transmitted in the request are returned in the response with unmodified values.

Creating a token during a payment

In order to understand the result, analyze the following fields:

Field name	Description
vads_page_action	Completed action. The returned value is **REGISTER_PAY**.
vads_identifier_status	Token creation status. Possible values are: • **CREATED**: the request for authorization or information, if supported by the acquirer is accepted. Token has been successfully created. • **NOT_CREATED**: the authorization or information request has been rejected. The token has not been created, and therefore cannot be viewed in the Expert Back Office. • **ABANDONED**: the action has been abandoned by the buyer. The token has not been created, and therefore cannot be viewed in the Expert Back Office.
vads_trans_status	Status of the transaction. Possible values are: • **AUTHORISED** The authorization or information request has been accepted. The token has been created and is visible in the Expert Back Office. • **AUTHORISED_TO_VALIDATE** The authorization or information request has been accepted. The merchant must validate the transaction manually. The token has been created and is visible in the Expert Back Office. • **CAPTURED** The authorization or information request has been accepted. The payment is visible in the “Captured transactions” tab of the Back Office. The token has been created and is visible in the Expert Back Office. • **WAITING_AUTHORISATION** The capture delay in the bank exceeds the authorization validity period. The authorization request for the total amount has not taken place yet. The token has been created and is visible in the Expert Back Office. • **WAITING_AUTHORISATION_TO_VALIDATE** **To be validated and authorized** The capture delay in the bank exceeds the authorization validity period. An authorization of 1 EUR (or information request about the CB network if the acquirer supports it) has been accepted. The merchant must manually validate the transaction in order to trigger the authorization request and capture. • **REFUSED** The authorization or information request has been rejected. The token is not created. • **ABANDONED** Operation abandoned by the buyer. The transaction is then not visible via the Expert Back Office. The token is not created.
vads_identifier	Payment token. The returned value is: • either the value transmitted in the request, regardless of the result of token creation, even in case of abandoned action, • or the value generated by the payment gateway, if the field is not transmitted in the request and the token has been successfully created (vads_identifier_status=CREATED).

Field name	Description
	Note If you have activated the token uniqueness check and the payment method is already registered with another token, then this token will be returned. The vads_identifier field will not be returned: • if it has not been transmitted in the request and the buyer abandons the action (vads_identifier_status=ABANDONED), • if it has not been transmitted in the request and the token has not been created (vads_identifier_status=NOT_CREATED).
vads_identifier_previously_registered	Present only if both conditions are true: • You have enabled the token uniqueness check. • The used payment method is already registered with another token.
vads_cust_email	Buyer's e-mail address transmitted in the request.
vads_site_id	Shop ID The returned value is the same as the one submitted in the form.
vads_ctx_mode	Operating mode. The returned value (**TEST** or **PRODUCTION**) is the same as the one submitted in the form.

To see the payment details, see the parameters below:

Transaction details:

Field name	Description
vads_operation_type	Transaction type Its value is **DEBIT**.
vads_occurrence_type	Transaction occurrence type. Its value is **SINGLE**.
vads_amount	Transaction amount. The returned value is the same as the one submitted in the form.
vads_currency	Code of the currency used for the payment.
vads_trans_id	Transaction identifier. The returned value is the same as the one submitted in the form.
vads_trans_uuid	Unique transaction ID. Its value is generated by the payment gateway.
vads_contract_used	MID associated with the transaction.
vads_auth_mode	Type of request made via authorization servers: • **MARK**: corresponds to an authorization for 1 EUR (or information request about the CB network if the acquirer supports it). Value also used if the period between the requested capture date and the current date is strictly greater than the authorization validity period. • **FULL**: corresponds to an authorization for the total amount of the transaction. Value also used if the period between the requested capture date and the current date is strictly shorter than the authorization validity period.
vads_auth_number	Authorization number returned by the authorization server. Empty if the authorization has failed.
vads_auth_result	Return code of the authorization request returned by the issuing bank. See chapter ***Managing the return codes of the authorization request***. Empty in case of an error prior to the authorization.
vads_risk_control	Result of the risk assessment. If at least one verification process returns the **ERROR** value, the transaction is rejected. See the description of the **vads_risk_analysis_result** field for more information.
vads_risk_assessment_result	List of actions made with the transaction, following the activation of the advanced risk assessment rules. Possible values are:

Field name	Description
	• **ENABLE_3DS**: 3D Secure enabled. • **DISABLE_3DS**: 3D Secure disabled. • **MANUAL_VALIDATION**: The transaction has been created via manual validation. The payment capture is temporarily blocked to allow the merchant to perform all the desired verification processes. • **REFUSE**: Transaction is declined. • **RUN_RISK_ANALYSIS**: Result of the external risk analyzer. See the description of the **vads_risk_analysis_result** field for more information. • **INFORM**: A warning message appears. The merchant is notified that a risk has been identified via one or several notification center rules.

Details of the used payment method:

Field name	Note
vads_acquirer_network	Acquirer network code.
vads_bank_code	Code of the issuing bank.
vads_bank_label	Name of the issuing bank of the payment card.
vads_bank_product	Product code of the used card.
vads_card_brand	Used payment method. See chapter **Compatible payment methods** to see the list of possible values.
vads_card_country	Country code of the card used in compliance with the ISO 3166 standard.
vads_card_number	Truncated/masked card number or IBAN and BIC used for the payment separated by "_" in case of a one-off debit payment. Since the BIC is optional, it may be left out. .
vads_expiry_month	Expiry month of the used card.
vads_expiry_year	Expiry year of the used card.

Authentication details of the cardholder:

Field name	Note
vads_threeds_auth_type	Cardholder authentication type. Strong cardholder authentication is mandatory when registering a card. The field will therefore always be populated with **CHALLENGE**.
vads_threeds_enrolled	Enrollment status of the buyer to the 3D Secure program. Possible values are: • **Y**: Authentication available. • **N**: Authentication not available. • **U**: Enrollment status to the 3D Secure program unknown. • empty: Incomplete 3DS authentication process (3DS disabled in the request, unregistered merchant or payment method not eligible for 3DS).
vads_threeds_status	3D Secure authentication result. Possible values are: • **Y**: Cardholder authentication success. • **N**: Cardholder authentication error. • **U**: Authentication impossible. • **A**: Authentication attempted but not completed.

Field name	Note
	• empty: Unauthorized 3DS authentication (3DS disabled in the request, unregistered cardholder or payment method not eligible for 3DS).

The optional fields transmitted in the request are returned in the response with unmodified values.

Creating a token during a recurring payment

In order to understand the result, analyze the following fields:

Field name	Description
vads_page_action	Completed action. The returned value is **REGISTER_SUBSCRIBE**.
vads_identifier_status	Token creation status. Possible values are: • **CREATED**: the request for authorization or information, if supported by the acquirer is accepted. Token has been successfully created. See the value of the field vads_recurrence_status to determine whether the subscription has been created. • **NOT_CREATED**: the authorization or information request has been rejected. The token has not been created, and therefore cannot be viewed in the Expert Back Office. The recurring payment is not created. • **ABANDONED**: the action has been abandoned by the buyer. The token has not been created, and therefore cannot be viewed in the Expert Back Office. The recurring payment is not created.
vads_recurrence_status	Recurrence creation status. The possible values are: • **CREATED**: The recurring payment has been successfully created. • **NOT_CREATED**: The recurring payment is not created. • **ABANDONED**: The action has been abandoned by the buyer. The recurring payment is not created.
vads_identifier	Payment token. The returned value is: • either the value transmitted in the request, regardless of the result of token creation, even in case of abandoned action, • or the value generated by the payment gateway, if the field is not transmitted in the request and the token has been successfully created (vads_identifier_status=CREATED). Note If you have activated the token uniqueness check and the payment method is already registered with another token, then this token will be returned. The vads_identifier field will not be returned: • if it has not been transmitted in the request and the buyer abandons the action (vads_identifier_status=ABANDONED), • if it has not been transmitted in the request and the token has not been created (vads_identifier_status=NOT_CREATED).
vads_identifier_previously_registered	Present only if both conditions are true: • You have enabled the token uniqueness check. • The used payment method is already registered with another token.
vads_cust_email	Buyer's e-mail address transmitted in the request.
vads_site_id	Shop ID The returned value is the same as the one submitted in the form.
vads_ctx_mode	Operating mode. The returned value (**TEST** or **PRODUCTION**) is the same as the one submitted in the form.

To see the recurring payment details, see the parameters below:

Field name	Note
vads_subscription	Recurring payment token. The returned value is: • either the value transmitted in the request, regardless of the result of subscription creation, • or the value generated by the payment gateway, if the field has not been transmitted in the request and the subscription has been successfully created (vads_recurrence_status=CREATED). The field vads_subscription will not be returned: • if it has not been transmitted in the request and the buyer abandons the action (vads_recurrence_status=ABANDONED), • if it has not been transmitted in the request and the subscription has not been created (vads_recurrence_status=NOT_CREATED).
vads_sub_amount	Installment amount (in the smallest currency unit)
vads_sub_currency	Code of the currency used for the recurring payment. E.g.: 978 for euro (EUR)
vads_sub_desc	Rule for recurring payments to apply according to the iCalendar RFC5545 specification. E.g.: RRULE:FREQ=MONTHLY
vads_sub_effect_date	Recurring payment start date. E.g.: 20210601
vads_sub_init_amount	Amount of the first installments. The number of the first installments is defined by **vads_sub_init_amount_number**. E.g.: 1000
vads_sub_init_amount_number	Number of installments to which will be applied the amount defined by **vads_sub_init_amount**. E.g.: 3

A "create a token when subscribing to a recurring payment" request triggers the creation of a VERIFICATION type transaction, visible in the Expert Back Office.

The purpose of this transaction is to help the merchant, via their Back Office, understand the reasons for refusal of token creation.

Here are its characteristics:

Field name	Description
vads_operation_type	Transaction type Its value is **VERIFICATION**.
vads_trans_status	Status of the transaction. Possible values are: • **ACCEPTED** The authorization or information request has been accepted. The token has been created and is visible in the Expert Back Office. • **REFUSED** The authorization or information request has been rejected. The token is not created.
vads_occurrence_type	Transaction occurrence type. Its value is **SINGLE**.
vads_amount	Possible values: • **0** if the acquirer supports information requests, • **100** otherwise.
vads_trans_id	Transaction identifier. Its value is generated by the payment gateway.

Field name	Description
vads_trans_uuid	Unique transaction ID. Its value is generated by the payment gateway.
vads_auth_mode	Type of request made via authorization servers. Its value is **MARK**.
vads_auth_number	Authorization number returned by the authorization server. Empty if the authorization has failed.
vads_auth_result	Return code of the authorization request returned by the issuing bank. See chapter ***Managing the return codes of the authorization request***. Empty in case of an error prior to the authorization.
vads_risk_assessment_result	List of actions made with the transaction, following the activation of the advanced risk assessment rules. **The rules related to the amount, or whose action is "Validate manually", do not apply in case of a VERIFICATION type transaction**. Possible values are: • **ENABLE_3DS**: 3D Secure enabled. • **DISABLE_3DS**: 3D Secure disabled. • **REFUSE**: The token creation request has been refused. • **RUN_RISK_ANALYSIS**: Result of the external risk analyzer. See the description of the **vads_risk_analysis_result** field for more information. • **INFORM**: A warning message appears. The merchant is notified that a risk has been identified via one or several notification center rules.

Details of the used payment method:

Field name	Note
vads_acquirer_network	Acquirer network code.
vads_bank_code	Code of the issuing bank.
vads_bank_label	Name of the issuing bank of the payment card.
vads_bank_product	Product code of the used card.
vads_card_brand	Used payment method. See chapter **Compatible payment methods** to see the list of possible values.
vads_card_country	Country code of the card used in compliance with the ISO 3166 standard.
vads_card_number	Truncated/masked card number or IBAN and BIC used for the payment separated by "_" in case of a one-off debit payment. Since the BIC is optional, it may be left out. .
vads_expiry_month	Expiry month of the used card.
vads_expiry_year	Expiry year of the used card.

Authentication details of the cardholder:

Field name	Note
vads_threeds_auth_type	Cardholder authentication type. Strong cardholder authentication is mandatory when registering a card. The field will therefore always be populated with **CHALLENGE**.
vads_threeds_enrolled	Enrollment status of the buyer to the 3D Secure program. Possible values are: • **Y**: Authentication available. • **N**: Authentication not available. • **U**: Enrollment status to the 3D Secure program unknown.

Field name	Note
	• empty: Incomplete 3DS authentication process (3DS disabled in the request, unregistered merchant or payment method not eligible for 3DS).
vads_threeds_status	3D Secure authentication result. Possible values are: • **Y**: Cardholder authentication success. • **N**: Cardholder authentication error. • **U**: Authentication impossible. • **A**: Authentication attempted but not completed. • empty: Unauthorized 3DS authentication (3DS disabled in the request, unregistered cardholder or payment method not eligible for 3DS).

The optional fields transmitted in the request are returned in the response with unmodified values.

Creating a token during creation of a recurring payment with payment

In order to understand the result, analyze the following fields:

Field name	Description
vads_page_action	Completed action. The returned value is **REGISTER_PAY_SUBSCRIBE**.
vads_identifier_status	Token creation status. Possible values are: • **CREATED**: the request for authorization or information, if supported by the acquirer is accepted. Token has been successfully created. See the value of the field vads_recurrence_status to determine whether the subscription has been created. • **NOT_CREATED**: the authorization or information request has been rejected. The token has not been created, and therefore cannot be viewed in the Expert Back Office. The recurring payment is not created. • **ABANDONED**: the action has been abandoned by the buyer. The token has not been created, and therefore cannot be viewed in the Expert Back Office. The recurring payment is not created.
vads_trans_status	Status of the transaction. Possible values are: • **AUTHORISED** The authorization or information request has been accepted. The token has been created and is visible in the Expert Back Office. • **AUTHORISED_TO_VALIDATE** The authorization or information request has been accepted. The merchant must validate the transaction manually. The token has been created and is visible in the Expert Back Office. • **CAPTURED** The authorization or information request has been accepted. The payment is visible in the “Captured transactions” tab of the Back Office. The token has been created and is visible in the Expert Back Office. • **WAITING_AUTHORISATION** The capture delay in the bank exceeds the authorization validity period. The authorization request for the total amount has not taken place yet. The token has been created and is visible in the Expert Back Office. • **WAITING_AUTHORISATION_TO_VALIDATE** **To be validated and authorized** The capture delay in the bank exceeds the authorization validity period. An authorization of 1 EUR (or information request about the CB network if the acquirer supports it) has been accepted. The merchant must manually validate the transaction in order to trigger the authorization request and capture. • **REFUSED** The authorization or information request has been rejected. The token is not created. • **ABANDONED** Operation abandoned by the buyer. The transaction is then not visible via the Expert Back Office. The token is not created.
vads_recurrence_status	Recurrence creation status. Possible values are: • **CREATED**: The recurring payment has been successfully created. • **NOT_CREATED**: The recurring payment is not created.

Field name	Description
	• **ABANDONED**: the action has been abandoned by the buyer. The recurring payment is not created.
vads_identifier	Payment token. The returned value is: • either the value transmitted in the request, regardless of the result of token creation, even in case of abandoned action, • or the value generated by the payment gateway, if the field is not transmitted in the request and the token has been successfully created (vads_identifier_status=CREATED). Note If you have activated the token uniqueness check and the payment method is already registered with another token, then this token will be returned. The vads_identifier field will not be returned: • if it has not been transmitted in the request and the buyer abandons the action (vads_identifier_status=ABANDONED), • if it has not been transmitted in the request and the token has not been created (vads_identifier_status=NOT_CREATED).
vads_identifier_previously_registered	Present only if both conditions are true: • You have enabled the token uniqueness check. • The used payment method is already registered with another token.
vads_cust_email	Buyer's e-mail address transmitted in the request.
vads_site_id	Shop ID The returned value is the same as the one submitted in the form.
vads_ctx_mode	Operating mode. The returned value (**TEST** or **PRODUCTION**) is the same as the one submitted in the form.

To see the recurring payment details, see the parameters below:

Field name	Note
vads_subscription	Recurring payment token. The returned value is: • either the value transmitted in the request, regardless of the result of subscription creation, • or the value generated by the payment gateway, if the field has not been transmitted in the request and the subscription has been successfully created (vads_recurrence_status=CREATED). The field vads_subscription will not be returned: • if it has not been transmitted in the request and the buyer abandons the action (vads_recurrence_status=ABANDONED), • if it has not been transmitted in the request and the subscription has not been created (vads_recurrence_status=NOT_CREATED).
vads_sub_amount	Installment amount (in the smallest currency unit)
vads_sub_currency	Code of the currency used for the recurring payment. E.g.: 978 for euro (EUR)
vads_sub_desc	Rule for recurring payments to apply according to the iCalendar RFC5545 specification. E.g.: RRULE:FREQ=MONTHLY
vads_sub_effect_date	Recurring payment start date. E.g.: 20210601
vads_sub_init_amount	Amount of the first installments. The number of the first installments is defined by **vads_sub_init_amount_number**.

Field name	Note
	E.g.: 1000
vads_sub_init_amount_number	Number of installments to which will be applied the amount defined by **vads_sub_init_amount**. E.g.: 3

To see the payment details, see the parameters below:

Transaction details:

Field name	Description
vads_operation_type	Transaction type Its value is **DEBIT**.
vads_occurrence_type	Transaction occurrence type. **This transaction is not part of the recurring payment.** Its value is **SINGLE**.
vads_amount	Transaction amount. The returned value is the same as the one submitted in the form.
vads_currency	Code of the currency used for the payment.
vads_trans_id	Transaction identifier. The returned value is the same as the one submitted in the form.
vads_trans_uuid	Unique transaction ID. Its value is generated by the payment gateway.
vads_contract_used	MID associated with the transaction.
vads_auth_mode	Type of request made via authorization servers: • **MARK**: corresponds to an authorization for 1 EUR (or information request about the CB network if the acquirer supports it). Value also used if the period between the requested capture date and the current date is strictly greater than the authorization validity period. • **FULL**: corresponds to an authorization for the total amount of the transaction. Value also used if the period between the requested capture date and the current date is strictly shorter than the authorization validity period.
vads_auth_number	Authorization number returned by the authorization server. Empty if the authorization has failed.
vads_auth_result	Return code of the authorization request returned by the issuing bank. Empty in case of an error prior to the authorization.
vads_risk_control	Result of the risk assessment. If at least one verification process returns the **ERROR** value, the transaction is rejected. See the description of the **vads_risk_analysis_result** field for more information.
vads_risk_assessment_result	List of actions made with the transaction, following the activation of the advanced risk assessment rules. The possible values are: • **ENABLE_3DS**: 3D Secure enabled. • **DISABLE_3DS**: 3D Secure disabled. • **MANUAL_VALIDATION**: The transaction has been created via manual validation. The payment capture is temporarily blocked to allow the merchant to perform all the desired verification processes. • **REFUSE**: Transaction is declined. • **RUN_RISK_ANALYSIS**: Result of the external risk analyzer. See the description of the **vads_risk_analysis_result** field for more information. • **INFORM**: A warning message appears. The merchant is notified that a risk has been identified via one or several notification center rules.

Details of the used payment method:

Field name	Note
vads_acquirer_network	Acquirer network code.
vads_bank_code	Code of the issuing bank.
vads_bank_label	Name of the issuing bank of the payment card.
vads_bank_product	Product code of the used card.
vads_card_brand	Used payment method. See chapter **Compatible payment methods** to see the list of possible values.
vads_card_country	Country code of the card used in compliance with the ISO 3166 standard.
vads_card_number	Truncated/masked card number or IBAN and BIC used for the payment separated by "_" in case of a one-off debit payment. Since the BIC is optional, it may be left out. .
vads_expiry_month	Expiry month of the used card.
vads_expiry_year	Expiry year of the used card.

Authentication details of the cardholder:

Field name	Note
vads_threeds_auth_type	Cardholder authentication type. Strong cardholder authentication is mandatory when registering a card. The field will therefore always be populated with **CHALLENGE**.
vads_threeds_enrolled	Enrollment status of the buyer to the 3D Secure program. Possible values are: • **Y**: Authentication available. • **N**: Authentication not available. • **U**: Enrollment status to the 3D Secure program unknown. • empty: Incomplete 3DS authentication process (3DS disabled in the request, unregistered merchant or payment method not eligible for 3DS).
vads_threeds_status	3D Secure authentication result. Possible values are: • **Y**: Cardholder authentication success. • **N**: Cardholder authentication error. • **U**: Authentication impossible. • **A**: Authentication attempted but not completed. • empty: Unauthorized 3DS authentication (3DS disabled in the request, unregistered cardholder or payment method not eligible for 3DS).

The optional fields transmitted in the request are returned in the response with unmodified values.

Payment by token

In order to understand the result, analyze the following fields:

Field name	Description
vads_page_action	Completed action. The returned value is **PAYMENT**.
vads_trans_status	Status of the transaction. Possible values are: • **AUTHORISED**

Field name	Description
	The authorization request has been accepted. • **AUTHORISED_TO_VALIDATE** The authorization request has been accepted. The merchant must validate the transaction manually. • **CAPTURED** The authorization request has been accepted. The payment is visible in the “Captured transactions” tab of the Back Office. • **WAITING_AUTHORISATION** The capture delay in the bank exceeds the authorization validity period. The authorization request for the total amount has not taken place yet. • **WAITING_AUTHORISATION_TO_VALIDATE** **To be validated and authorized** The capture delay in the bank exceeds the authorization validity period. An authorization of 1 EUR (or information request about the CB network if the acquirer supports it) has been accepted. The merchant must manually validate the transaction in order to trigger the authorization request and capture. • **REFUSED** The authorization request has been declined. • **ABANDONED** Operation abandoned by the buyer. The transaction is then not visible via the Expert Back Office.
vads_identifier	Recurring payment token to debit. The returned value is the same as the one submitted in the form.
vads_cust_email	E-mail address of the buyer associated with the token.
vads_site_id	Shop ID The returned value is the same as the one submitted in the form.
vads_ctx_mode	Operating mode. The returned value (**TEST** or **PRODUCTION**) is the same as the one submitted in the form.

To see the payment details, see the parameters below:

Transaction details:

Field name	Description
vads_operation_type	Transaction type Its value is **DEBIT**.
vads_occurrence_type	Transaction occurrence type. Its value is **SINGLE**.
vads_amount	Transaction amount. The returned value is the same as the one submitted in the form.
vads_currency	Code of the currency used for the payment.
vads_trans_id	Transaction identifier. The returned value is the same as the one submitted in the form.
vads_trans_uuid	Unique transaction ID. Its value is generated by the payment gateway.
vads_contract_used	MID associated with the transaction.
vads_auth_mode	Type of request made via authorization servers: • **MARK**: corresponds to an authorization for 1 EUR (or information request about the CB network if the acquirer supports it). Value also used if the period between the requested capture date and the current date is strictly greater than the authorization validity period. • **FULL**: corresponds to an authorization for the total amount of the transaction.

Field name	Description
	Value also used if the period between the requested capture date and the current date is strictly shorter than the authorization validity period.
vads_auth_number	Authorization number returned by the authorization server. Empty if the authorization has failed.
vads_auth_result	Return code of the authorization request returned by the issuing bank. See chapter ***Managing the return codes of the authorization request***. Empty in case of an error prior to the authorization.
vads_risk_control	Result of the risk assessment. If at least one verification process returns the **ERROR** value, the transaction is rejected. See the description of the **vads_risk_analysis_result** field for more information.
vads_risk_assessment_result	List of actions made with the transaction, following the activation of the advanced risk assessment rules. Possible values are: • **ENABLE_3DS**: 3D Secure enabled. • **DISABLE_3DS**: 3D Secure disabled. • **MANUAL_VALIDATION**: The transaction has been created via manual validation. The payment capture is temporarily blocked to allow the merchant to perform all the desired verification processes. • **REFUSE**: The transaction is refused. • **RUN_RISK_ANALYSIS**: Result of the external risk analyzer. See the description of the **vads_risk_analysis_result** field for more information. • **INFORM**: A warning message appears. The merchant is notified that a risk has been identified via one or several notification center rules.

Details of the used payment method:

Field name	Note
vads_acquirer_network	Acquirer network code.
vads_bank_code	Code of the issuing bank.
vads_bank_label	Name of the issuing bank of the payment card.
vads_bank_product	Product code of the used card.
vads_card_brand	Used payment method. See chapter **Compatible payment methods** to see the list of possible values.
vads_card_country	Country code of the card used in compliance with the ISO 3166 standard.
vads_card_number	Truncated/masked card number or IBAN and BIC used for the payment separated by "_" in case of a one-off debit payment. Since the BIC is optional, it may be left out. .
vads_expiry_month	Expiry month of the used card.
vads_expiry_year	Expiry year of the used card.

Details of strong authentication:

Field name	Note
vads_threeds_auth_type	Cardholder authentication type. Possible values are: • "Empty" if the buyer is not correctly authenticated. • **FRICTIONLESS**: cardholder authentication without interaction with the ACS. Value returned only in 3DS v2.

Field name	Note
	• **CHALLENGE**: interactive cardholder authentication (entering an OTP or replying to a series of questions). Value returned in 3DS v1 and 3DS v2.
vads_threeds_enrolled	Enrollment status of the buyer to the 3D Secure program. Possible values are: • **Y**: Authentication available. • **N**: Authentication not available. • **U**: Enrollment status to the 3D Secure program unknown. • empty: Incomplete 3DS authentication process (3DS disabled in the request, unregistered merchant or payment method not eligible for 3DS).
vads_threeds_status	3D Secure authentication result. Possible values are: • **Y**: Cardholder authentication success. • **N**: Cardholder authentication error. • **U**: Authentication impossible. • **A**: Authentication attempted but not completed. • empty: Unauthorized 3DS authentication (3DS disabled in the request, unregistered cardholder or payment method not eligible for 3DS).

The optional fields transmitted in the request are returned in the response with unmodified values.

Creating a recurring payment

In order to understand the result, analyze the following fields:

Field name	Description
vads_page_action	Completed action. The returned value is **SUBSCRIBE**.
vads_recurrence_status	Recurrence creation status. The possible values are: • **CREATED**: The recurring payment has been successfully created. • **NOT_CREATED**: The recurring payment is not created. • **ABANDONED**: action abandoned by the buyer. The recurring payment is not created.
vads_identifier	Recurring payment token to debit. The returned value is the same as the one submitted in the form.
vads_cust_email	E-mail address of the buyer associated with the token.
vads_site_id	Shop ID The returned value is the same as the one submitted in the form.
vads_ctx_mode	Operating mode. The returned value (**TEST** or **PRODUCTION**) is the same as the one submitted in the form.

To see the recurring payment details, see the parameters below:

Field name	Note
vads_subscription	Recurring payment token. The returned value is: • either the value transmitted in the request, regardless of the result of subscription creation,

Field name	Note
	• or the value generated by the payment gateway, if the field has not been transmitted in the request and the subscription has been successfully created (vads_recurrence_status=CREATED). The field vads_subscription will not be returned: • if it has not been transmitted in the request and the buyer abandons the action (vads_recurrence_status=ABANDONED), • if it has not been transmitted in the request and the subscription has not been created (vads_recurrence_status=NOT_CREATED).
vads_sub_amount	Installment amount (in the smallest currency unit)
vads_sub_currency	Code of the currency used for the recurring payment. E.g.: 978 for euro (EUR)
vads_sub_desc	Rule for recurring payments to apply according to the iCalendar RFC5545 specification. E.g.: RRULE:FREQ=MONTHLY
vads_sub_effect_date	Recurring payment start date. E.g.: 20210601
vads_sub_init_amount	Amount of the first installments. The number of the first installments is defined by **vads_sub_init_amount_number**. E.g.: 1000
vads_sub_init_amount_number	Number of installments to which will be applied the amount defined by **vads_sub_init_amount**. E.g.: 3

Details of the used payment method:

Field name	Note
vads_acquirer_network	Acquirer network code.
vads_bank_code	Code of the issuing bank.
vads_bank_label	Name of the issuing bank of the payment card.
vads_bank_product	Product code of the used card.
vads_card_brand	Used payment method. See chapter **Compatible payment methods** to see the list of possible values.
vads_card_country	Country code of the card used in compliance with the ISO 3166 standard.
vads_card_number	Truncated/masked card number or IBAN and BIC used for the payment separated by "_" in case of a one-off debit payment. Since the BIC is optional, it may be left out. .
vads_expiry_month	Expiry month of the used card.
vads_expiry_year	Expiry year of the used card.

Details of strong authentication performed when creating the token:

Field name	Note
vads_threeds_auth_type	Cardholder authentication type. Strong cardholder authentication is mandatory when registering a card. The field will therefore always be populated with **CHALLENGE**.
vads_threeds_enrolled	Enrollment status of the buyer to the 3D Secure program. Possible values are: • **Y**: Authentication available. • **N**: Authentication not available. • **U**: Enrollment status to the 3D Secure program unknown.

Field name	Note
	• empty: Incomplete 3DS authentication process (3DS disabled in the request, unregistered merchant or payment method not eligible for 3DS).
vads_threeds_status	3D Secure authentication result. Possible values are: • **Y**: Cardholder authentication success. • **N**: Cardholder authentication error. • **U**: Authentication impossible. • **A**: Authentication attempted but not completed. • empty: Unauthorized 3DS authentication (3DS disabled in the request, unregistered cardholder or payment method not eligible for 3DS).

The optional fields transmitted in the request are returned in the response with unmodified values.

Payment with the token creation option for the cardholder

In order to understand the result, analyze the following fields:

Field name	Description
vads_page_action	Completed action. The returned value is **ASK_REGISTER_PAY**.
vads_identifier_status	Token creation status. **The field will not be sent if the buyer did not get the approval to record card details**. Possible values are: • **CREATED**: the request for authorization or information, if supported by the acquirer is accepted. Token has been successfully created. • **NOT_CREATED**: the request for authorization or information has been rejected. The token has not been created, and therefore cannot be viewed in the Expert Back Office. • **ABANDONED**: action abandoned by the buyer. The token has not been created, and therefore cannot be viewed in the Expert Back Office.
vads_trans_status	Status of the transaction. Possible values are: • **AUTHORISED** The authorization or information request has been accepted. The token has been created and is visible in the Expert Back Office. • **AUTHORISED_TO_VALIDATE** The authorization or information request has been accepted. The merchant must manually validate the transaction. The token has been created and is visible in the Expert Back Office. • **CAPTURED** The authorization or information request has been accepted. The transaction is visible in the “Captured transactions” tab of the Back Office. The token has been created and is visible in the Expert Back Office. • **WAITING_AUTHORISATION** The capture delay in the bank exceeds the authorization validity period. The authorization request for the total amount has not yet been sent. The token has been created and is visible in the Expert Back Office. • **WAITING_AUTHORISATION_TO_VALIDATE**

Field name	Description
	To be validated and authorized The capture delay in the bank exceeds the authorization validity period. An 1 EUR (or information request about the CB network if the acquirer supports it) authorization has been accepted. The merchant must manually validate the transaction in order to trigger the authorization request and capture. • **REFUSED** The authorization or information request has been rejected. The token is not created. • **ABANDONED** Operation abandoned by the buyer. The transaction is then not visible via the Expert Back Office. The token is not created.
vads_identifier	Payment token. **The field will not be sent if the buyer has not given the approval to register their card details**. The returned value is: • either the value transmitted in the request, regardless of the result of token creation, even in case of abandoned action, • or the value generated by the payment gateway, if the field is not transmitted in the request and the token has been successfully created (vads_identifier_status=CREATED). Note If you have activated the token uniqueness check and the payment method is already registered with another token, then this token will be returned. The vads_identifier field will not be returned: • if it has not been transmitted in the request and the buyer abandons the action (vads_identifier_status=ABANDONED), • if it has not been transmitted in the request and the token has not been created (vads_identifier_status=NOT_CREATED).
vads_cust_email	Buyer's e-mail address transmitted in the request.
vads_site_id	Shop ID The returned value is the same as the one submitted in the form.
vads_ctx_mode	Operating mode. The returned value (**TEST** or **PRODUCTION**) is the same as the one submitted in the form.

To view the payment details, see the parameters below:

Transaction details:

Field name	Description
vads_operation_type	Transaction type Its value is **DEBIT**.
vads_amount	Transaction amount. The returned value is the same as the one submitted in the form.
vads_currency	Code of the currency used for the payment.
vads_trans_id	Transaction identifier. The returned value is the same as the one submitted in the form.
vads_trans_uuid	Unique transaction ID. Its value is generated by the payment gateway.
vads_contract_used	MID associated with the transaction.
vads_auth_mode	Type of request made via authorization servers: • **MARK**: corresponds to an authorization of 1 EUR (or information request about the CB network if the acquirer supports it).

Field name	Description
	Value also used if the period between the requested capture date and the current date is strictly greater than the authorization validity period. • **FULL**: corresponds to an authorization of the total transaction amount. Value also used if the period between the requested capture date and the current date is strictly shorter than the authorization validity period.
vads_auth_number	Authorization number returned by the authorization server. Empty if the authorization has failed.
vads_auth_result	Return code of the authorization request returned by the issuing bank. See chapter ***Managing the return codes of the authorization request***. Empty in case of an error prior to the authorization.
vads_risk_control	Result of the risk assessment. If at least one verification process returns the **ERROR** value, the transaction is rejected. See the description of the **vads_risk_analysis_result** field for more information.
vads_risk_assessment_result	List of actions made with the transaction, following the activation of the advanced risk assessment rules. Possible values are: • **ENABLE_3DS**: 3D Secure enabled. • **DISABLE_3DS**: 3D Secure disabled. • **MANUAL_VALIDATION**: The transaction has been created via manual validation. The payment capture is temporarily blocked to allow the merchant to perform all the desired verification processes. • **REFUSE**: Transaction is declined. • **RUN_RISK_ANALYSIS**: Result of the external risk analyzer. See the description of the **vads_risk_analysis_result** field for more information. • **INFORM**: A warning message appears. The merchant is notified that a risk has been identified via one or several notification center rules.

Details of the used payment method:

Field name	Note
vads_acquirer_network	Acquirer network code.
vads_bank_code	Code of the issuing bank.
vads_bank_label	Name of the issuing bank of the payment card.
vads_bank_product	Product code of the used card.
vads_card_brand	Used payment method. See chapter **Compatible payment methods** to see the list of possible values.
vads_card_country	Country code of the card used in compliance with the ISO 3166 standard.
vads_card_number	Truncated/masked card number or IBAN and BIC used for the payment separated by "_" in case of a one-off debit payment. Since the BIC is optional, it may be left out. .
vads_expiry_month	Expiry month of the used card.
vads_expiry_year	Expiry year of the used card.

Authentication details of the cardholder:

Field name	Note
vads_threeds_auth_type	Cardholder authentication type.

Field name	Note
	Strong cardholder authentication is mandatory when registering a card. The field will therefore always be populated with **CHALLENGE**.
vads_threeds_enrolled	Enrollment status of the buyer to the 3D Secure program. Possible values are: • **Y**: Authentication available. • **N**: Authentication not available. • **U**: Enrollment status to the 3D Secure program unknown. • empty: Incomplete 3DS authentication process (3DS disabled in the request, unregistered merchant or payment method not eligible for 3DS).
vads_threeds_status	3D Secure authentication result. Possible values are: • **Y**: Cardholder authentication success. • **N**: Cardholder authentication error. • **U**: Authentication impossible. • **A**: Authentication attempted but not completed. • empty: Unauthorized 3DS authentication (3DS disabled in the request, unregistered cardholder or payment method not eligible for 3DS).

The optional fields transmitted in the request are returned in the response with unmodified values.

Installment payment

In order to understand the result, analyze the following fields:

Field name	Description
vads_page_action	Completed action. The returned value is **PAYMENT**.
vads_trans_status	Status of the transaction. Possible values are: • **AUTHORISED** The authorization request has been accepted. • **AUTHORISED_TO_VALIDATE** The authorization request has been accepted. The merchant must validate the transaction manually. • **CAPTURED** The authorization request has been accepted. The payment is visible in the “Captured transactions” tab of the Back Office. • **REFUSED** The authorization request has been declined.
vads_identifier	Payment token.
vads_cust_email	Buyer’s e-mail address.
vads_site_id	Shop ID
vads_ctx_mode	Operating mode.

To see the recurring payment details, see the parameters below:

Field name	Note
vads_subscription	Recurring payment token.
vads_recurrence_number	Recurrence number of the recurring payment.

To see the payment details, see the parameters below:

Transaction details:

Field name	Description
vads_operation_type	Transaction type Its value is **DEBIT**.
vads_occurrence_type	Transaction occurrence type. Possible values: • **RECURRENT_INITIAL**: First installment. • **RECURRENT_INTERMEDIAIRE**: Nth installment. View the **vads_recurrence_number** field to know the installment number. • **RECURRENT_FINAL**: Last installment.
vads_amount	Transaction amount. The returned value is the same as the one submitted in the form.
vads_currency	Code of the currency used for the payment.
vads_trans_id	Transaction identifier. The returned value is the same as the one submitted in the form.
vads_trans_uuid	Unique transaction ID. Its value is generated by the payment gateway.
vads_contract_used	MID associated with the transaction.
vads_auth_mode	Type of request made via authorization servers: • **MARK**: corresponds to an authorization for 1 EUR (or information request about the CB network if the acquirer supports it). Value also used if the period between the requested capture date and the current date is strictly greater than the authorization validity period. • **FULL**: corresponds to an authorization for the total amount of the transaction. Value also used if the period between the requested capture date and the current date is strictly shorter than the authorization validity period.
vads_auth_number	Authorization number returned by the authorization server. Empty if the authorization has failed.
vads_auth_result	Return code of the authorization request returned by the issuing bank. See chapter ***Managing the return codes of the authorization request***. Empty in case of an error prior to the authorization.
vads_risk_control	Result of the risk assessment. If at least one verification process returns the **ERROR** value, the transaction is rejected. See the description of the **vads_risk_analysis_result** field for more information.
vads_risk_assessment_result	List of actions made with the transaction, following the activation of the advanced risk assessment rules. Possible values are: • **ENABLE_3DS**: 3D Secure enabled. • **DISABLE_3DS**: 3D Secure disabled. • **MANUAL_VALIDATION**: The transaction has been created via manual validation. The payment capture is temporarily blocked to allow the merchant to perform all the desired verification processes. • **REFUSE**: Transaction is declined. • **RUN_RISK_ANALYSIS**: Result of the external risk analyzer. See the description of the **vads_risk_analysis_result** field for more information. • **INFORM**: A warning message appears. The merchant is notified that a risk has been identified via one or several notification center rules.

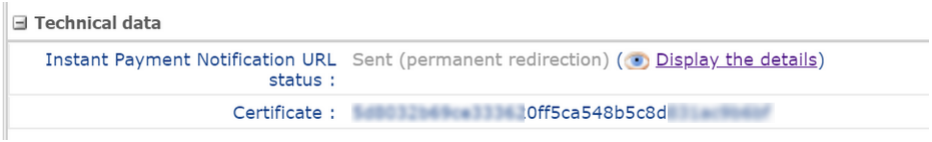
Details of the used payment method:

Field name	Note
vads_acquirer_network	Acquirer network code.
vads_bank_code	Code of the issuing bank.
vads_bank_label	Name of the issuing bank of the payment card.
vads_bank_product	Product code of the used card.
vads_card_brand	Used payment method. See chapter **Compatible payment methods** to see the list of possible values.
vads_card_country	Country code of the card used in compliance with the ISO 3166 standard.
vads_card_number	Truncated/masked card number or IBAN and BIC used for the payment separated by "_" in case of a one-off debit payment. Since the BIC is optional, it may be left out. .
vads_expiry_month	Expiry month of the used card.
vads_expiry_year	Expiry year of the used card.

15.7. Running tests and troubleshooting

In order to test the notifications, follow the the steps below:

1. Make a payment (in TEST mode or in PRODUCTION mode).
2. Once the payment is complete, look for the transaction in your Back Office (**Management** > **Transactions** or **TEST Transactions** menu if you made the payment in TEST mode).
3. Double-click the transaction to view the **transaction details**.
4. In the transaction details, search for the section entitled **Technical data**.
5. Check the status of the Instant Payment Notification URL:



The list of possible statuses is provided below:

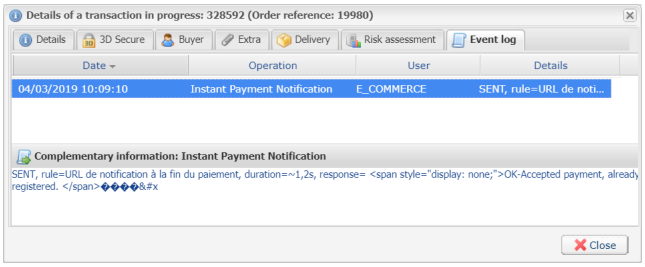
Status	Description
N/A	The transaction did not result in a notification or no notification rules have been enabled.
Undefined URL	An event has triggered the notification rule for end of payment but the URL is not configured.
Call in progress	The notification is in progress. This status is temporary.
Sent	The notification has been successfully sent and a remote device returned an HTTP 200, 201, 202, 203, 204, 205 or 206 response status code.
Sent (permanent redirection)	The merchant website has returned an HTTP 301 or 308 response status code with a new URL to contact. A new call in POST mode has been made to the new URL.
Sent (temporary redirection)	The merchant website has returned an HTTP 302 or 307 response status code with a new URL to contact. A new call in POST mode has been made to the new URL.
Sent (redirection to another page)	The merchant website has returned an HTTP 303 response status code with a new URL to contact. A new call in GET mode has been made to the new URL.
Failed	Generic error different from the codes described below.
Server unavailable	The notification has lasted more than 35s.
SSL handshake failure	Your server is incorrectly configured. Run a test on the Qualys website (*https://www.ssllabs.com/ssltest/*) and correct the errors.
Connection interrupted	Communication error.
Connection refused	Communication error.
Server error 300	Case of redirection not supported by the gateway.
Server error 304	Case of redirection not supported by the gateway.
Server error 305	Case of redirection not supported by the gateway.
Server error 400	The merchant website has returned an HTTP 400 Bad Request response status code.
Server error 401	The merchant website has returned an HTTP 401 Unauthorized response status code. Make sure that the resource is not protected by an .htaccess file.
Server error 402	The merchant website has returned an HTTP 402 Payment Required response status code.
Server error 403	The merchant website has returned an HTTP 403 Forbidden response status code. Make sure that the resource is not protected by an .htaccess file.
Server error 404	The merchant website has returned an HTTP 404 Not Found response status code. Make sure that the URL is correctly specified in the rule configuration. Make sure that the file is present on your server.
Server error 405	The merchant website has returned an HTTP 405 Method Not Allowed response status code.

Status	Description
Server error 406	The merchant website has returned an HTTP 406 Not Acceptable response status code.
Server error 407	The merchant website has returned an HTTP 407 Proxy Authentication Required response status code.
Server error 408	The merchant website has returned an HTTP 408 Request Time-out response status code.
Server error 409	The merchant website has returned an HTTP 409 Conflict response status code.
Server error 410	The merchant website has returned an HTTP 410 Gone response status code.
Server error 411	The merchant website has returned an HTTP 411 Length Required response status code.
Server error 412	The merchant website has returned an HTTP 412 Precondition Failed response status code.
Server error 413	The merchant website has returned an HTTP 413 Request Too Large response status code.
Server error 414	The merchant website has returned an HTTP 414 Request-URI Too Long response status code.
Server error 415	The merchant website has returned an HTTP 415 Unsupported Media Type response status code.
Server error 500	The merchant website has returned an HTTP 500 Internal Server Error response status code. An application error has occurred on the level of the server hosting your shop. See the logs of your HTTP server (usually apache). The issue can only be corrected with an intervention on your server.
Server error 501	The merchant website has returned an HTTP 501 Not Implemented response status code.
Server error 502	The merchant website has returned an HTTP 502 Bad Gateway / Proxy Error response status code.
Server error 503	The merchant website has returned an HTTP 503 Service Unavailable response status code.
Server error 504	The merchant website has returned an HTTP 504 Gateway Time-out response status code. The merchant server has not accepted the call within the time limit of 10s.
Server error 505	The merchant website has returned an HTTP 505 HTTP Version Not Supported response status code.

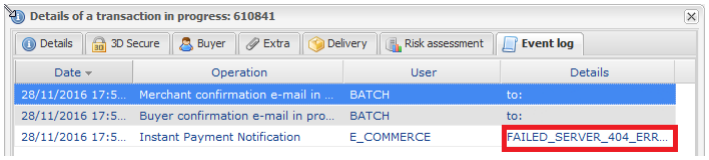
For more information on a notification, click the link **Display the details** or click the **Event log** tab and search for the line **Notification URL call**.

In order to help the merchant identify the source of the error, the gateway systematically analyses the 512 first characters returned by the merchant website and displays them in the **Details** column.

- Example of a successfully processed notification:



- Example of a failed notification:



If the payment gateway is unable to access the URL of your page, an e-mail alert will be sent to the shop administrator.

It contains:

- The HTTP code of the encountered error
- Parts of error analysis
- Its consequences
- Instructions to follow via the Expert Back Office for resending the request to the URL specified in step 4

16. APPENDIX

16.1. Automatically creating a recurring payment via Web services

- **V5 SOAP Web Services**

  Use the **createSubscription** operation.

  For more information, see the *SOAP Web service API v5 Implementation Guide* available in our online documentation archive.

- **RESTful payment Web Services**

  Use the **Charge/CreateSubscription** method to make recurring payments (subscription) using an existing and valid token.

  For more information, see the description of the *Charge/CreateSubscription* method.

16.2. Automatically canceling a recurring payment via Web services

- **V5 SOAP Web Services**

  Use the **cancelSubscription** operation to cancel a recurring payment on a given date.

  For more information, see the *SOAP Web service API v5 Implementation Guide* available in our online documentation archive.

- **RESTful payment Web Services**

  Use the **Subscription/Cancel** method to cancel a recurring payment.

  For more information, see the description of the *Subscription/Cancel* method.

16.3. Test cards

Test cards are available on the payment page.

Depending on the test scenario associated with the card, they allow:

- To create a token and/or a recurring payment only if the test result is “Payment accepted”.
- To not create a token if the test result is “Payment refused”.

In order to test the behavior when an installment is refused, you must use the following test card:

Card number	Test card checked
4970101000001002	Token creation OK - Payment refused due to exceeded limit if the installment amount is higher than 0.

Note

This card is not offered on the payment page upon token creation.

