DOKU API MERCHANT-HOSTED DOCUMENTATION

Version 1.10
June 2017
Table of Contents

1.0 Introduction ....................................................................................................................................................... 3
  1.1 Merchant Hosted Integration ................................................................................................................................. 3
2.0 Credit Card .............................................................................................................................................................. 5
  2.1 Advanced Features ................................................................................................................................................ 10
    2.1.1 BIN Filtering .................................................................................................................................................. 10
    2.1.2 2-Click Payment [Tokenization] ..................................................................................................................... 11
    2.1.3 1-Click Payment [Recurring] .......................................................................................................................... 13
    2.1.4 Installment ..................................................................................................................................................... 16
    2.1.5 Authorize & Capture ........................................................................................................................................ 18
3.0 DOKU Wallet ........................................................................................................................................................ 19
4.0 Virtual Account ...................................................................................................................................................... 23
  4.1 Bank Transfer ....................................................................................................................................................... 23
  4.2 Convenience Store ............................................................................................................................................... 27
5.0 Internet Banking .................................................................................................................................................... 30
  5.1 Mandiri Clickpay ............................................................................................................................................... 30
6.0 Appendix ............................................................................................................................................................... 35
  6.1 Payment Methods ............................................................................................................................................... 35
  6.2 Payment Channel Codes .................................................................................................................................. 35
  6.3 Parameters Required ........................................................................................................................................... 36
    6.3.1 Payment Request ........................................................................................................................................... 36
    6.3.2 Check Status Request .................................................................................................................................. 37
    6.3.2 Check Status Response .................................................................................................................................. 37
  6.2 DOKU Response Codes ....................................................................................................................................... 38
    6.2.1 General response codes .................................................................................................................................. 38
    6.2.2 Credit Card .................................................................................................................................................... 39
    6.2.3 DOKU Wallet ................................................................................................................................................ 43
    6.2.4 Virtual Account ............................................................................................................................................... 44
    6.2.5 Mandiri Clickpay ........................................................................................................................................... 44
1.0 Introduction

DOKU enables you to accept a wide range of different payment methods. Each payment method has a unique approach to integration. The payment methods can be grouped into five distinct categories:

- Credit Card
- DOKU wallet
- Bank transfer (Virtual Account)
- Convenience store
- Internet banking

For more information about the payment methods, please see Appendix 6.1.

The DOKU API supports two methods of payment processing – namely, merchant-hosted and DOKU-hosted. These two services can be tailored to accommodate the separate business needs of merchants from those having advanced IT personnel, to merchants who do not even have a website. The DOKU-hosted API is detailed in a separate document. See below for features of each service:

1. Merchant Hosted

   The payment page and data input is native to the merchant’s website, without having to redirect to a DOKU-hosted page. Having the payment form on the merchant page does not compromise the security of the cardholder however, as DOKU is PCI (Level 1) certified, and none of the cardholder data will actually be stored on the merchant’s server.

   Merchant profile: Tech-savvy merchants who want a seamless transaction process and maintain full control of the payment page branding and experience. IT expertise is required to modify the payment page and use client-side encryption to encrypt card data.

2. DOKU Hosted

   Instant payment services where the payment input form is located within the DOKU page. The selection of payment methods can be done on the merchant or DOKU page. With this service, the customer will be redirected to a DOKU-hosted page upon checkout to complete the payment.

   Merchant profile: Merchants who prefer to let DOKU manage the entire data security management and do not require extensive payment page customization.

1.1 Merchant Hosted Integration

The next section gives an example of how you can integrate with DOKU for the various payment methods. Once you have confirmed to become a DOKU merchant through our Sales process, our integration team to proceed to the technical integration stage will contact you. All new merchants will receive a shared key and a merchant code. Take note of this information, as you will need to enter them into the API script during integration. The response codes are categorized by payment method, and can be found in the appendix.
The following instructions are divided into standalone APIs for each payment method. Therefore, there may be some repetition. When you do your actual integration, some of the steps may be skipped when adding on different payment methods and features.
2.0 Credit Card

By default all credit card payments processed by DOKU will undergo 3D secure. Non-3D secure payments are available, however would require further assessment by DOKU and the bank.

Credit card integration comprises 3 easy steps:

1. Insert JavaScript
2. Initiate JavaScript parameters
3. Create payment form

To get started on your integration, follow these steps one by one by pasting the template scripts onto your website. The DOKU payment form is saved in HTML format; the template script acts as a placeholder for where the payment form will appear on your page. Prepare your HTML pages as seen in the template, and customize accordingly.

1. Insert the doku.js, fancybox.js and fancybox.css onto your website’s payment page, along with your custom style. See example:

```php
<?php
require_once('../Doku.php');
Doku_Initiate::$sharedKey = '<Put Your Shared Key Here>,'
Doku_Initiate::$mallId = '<Put Your Merchant Code Here>,'
$invoice = 'invoice_1458123951';
$params = array(
    'amount' => '10000.00',
    'invoice' => $invoice,
    'currency' => '360'
);
$words = doku_Library::doCreateWords($params);
?>
```

...and adding the following example script on to your webpage:
3. Insert the template script where you wish the payment form to appear on your website.

```html
<script type="text/javascript">
$(function() {
    var data = new Object();
    data.req_merchant_code = '1';
    data.req_chain_merchant = 'NA';
    data.req_payment_channel = '15'; // '15' = credit card
    data.req_transaction_id = '<?php echo $invoice ?>';
    data.req_currency = '<?php echo $currency ?>';
    data.req_amount = '<?php echo $amount ?>';
    data.req_words = '<?php echo $words ?>';
    data.req_form_type = 'full';
    getForm(data);
});
</script>

<form action="charge.php" method="POST" id="payment-form">
    <div doku-div='form-payment'>
        <input id="doku-token" name="doku-token" type="hidden" />
        <input id="doku-pairing-code" name="doku-pairing-code" type="hidden" />
    </div>
</form>
```

When completed correctly, the payment form should appear as seen below, and you may start receiving payments immediately. The DOKU script only provides the four fields to be filled in by the customer. All other parts of the payment page, including the logo and ‘Process Payment’ button are customizable to your needs for a completely white label payment flow.

Once your customer has input their card data and clicked the “Process Payment” button, the data will be processed by DOKU. The DOKU server then responds with the card ID, and the browser will post the data to the merchant server, according to the URL set in your action form.
Due to potential fraud, Indonesia is a 3D Secure market, where customer authentication is required to proceed with the transaction. The form of authentication differs form each bank, but typically it involves a One-Time Password sent via SMS. Not all issuing banks in Indonesia have implemented 3D secure, but the majority of them have. If you would like for the 3D secure process to be lifted, please contact DOKU for more information. Further assessment by DOKU and the acquiring bank is required to grant a non-3D secure Merchant ID.

You can retrieve the posted data from the form within your server. Then send the payment request to the DOKU server to be processed using our library. See example:
After this is done, you may display the result on your browser for the customer to see. This result page is completely customizable according to the design that you prefer.
then, you can retrieve the JSON data like this. With this JSON data, you can acknowledge for updating the payment status of transaction.

MIPPayment.processRequest ACKNOWLEDGE :

```json
{"res_approval_code":"681905",
"res_edu_status":"NA",
"res_trans_id_merchant":"124",
"res_amount":"29575.04",
"res_payment_date_time":"20160812145059",
"res_verify_score":"-1",
"res_verify_id":"
"res_verify_status":"NA",
"res_words":"7c3d05dcad40ec7b2d41d2f75e2a896ff3e898d9",
"res_response_msg":"SUCCESS",
"res_mcn":"548117******6802",
"res_mid":"000002187008751",
"res_bank":"PT. BANK CIMB NIAGA TBK.",
"res_response_code":"0000",
"res_session_id":"b2c981931fcff3c3f6db89f1a7c33656b10b11a6",
"res_payment_channel":"15"}
```
2.1 Advanced Features

2.1.1 BIN Filtering

BIN filtering is a feature that can be used to filter Credit Cards issued by certain banks for the purpose of doing promotions. Each issuer has a unique BIN number, which is made up of the first 6 digits in the Credit Card number. The conditions you set in the filter will specify which BIN numbers that are allowed to make payments on your site. When a card number that has been blocked by the BIN filter is entered, the DOKU server will not be able to process the payment.

In order to apply the BIN filter, you must insert the following conditions into a prepayment request. The prepayment request must be sent before the actual payment request. You can add several conditions by separating with a comma. See example:

```
<?php
require_once('./Doku.php');

Doku_Initiate::$sharedKey = '<Put Your Shared Key Here>';
Doku_Initiate::$mallId = '<Put Your Merchant Code Here>';

$data = array('req_token_id' => $data['doku-token'],
               'req_pairing_code' => $data['doku-pairing-code'],
               'req_bin_filter' => array("42*1", "4??3??", "411111", "5*"),
               'req_words' => $words);

$responsePrePayment = Doku_Api::doPrePayment($data);
if($responsePrePayment->responseCode == '0000'){
    // prepayment success
    // Pre payment success, then you can do payment
} else{
    // Pre payment fail
}
```

Afterwards, you may insert your original payment request form. Refer to the Appendix for prepayment response codes.
2.1.2 2-Click Payment [Tokenization]

2-click payment enables the customer to make a purchase without having to input card details or personal information, apart from the CVV number. This process is typically used by merchants that have repeat customers who will benefit from a faster checkout by reducing the number of fields the customer needs to fill in. If the card issuer requires 3D secure verification process, the customer will still have to complete this to make a purchase. In order for this process to work, the customer enters all of the card information only during the very first time they make a purchase. DOKU stores this data in a secure form and gives the merchant a token, which is paired to the customer’s login credentials on the merchant website. After this process, has been completed, each time they make a payment from hereon out, they only have to input the CVV.

Follow these steps to apply 2-click payment to your credit card payment process:

1. Insert the additional script to your server under the payment data.
2. Generate and save the token during the first payment.
3. For subsequent payments, retrieve the token from your database and send it to the DOKU server.

1. To initialize 2-click payment, follow the same steps as general credit card processing, but add the additional parameter “data.reg_customer_id” to your javascript under the payment data. The “data.reg_customer_id” parameter may represent the customer ID that you assign to each customer within your database. This ID will be paired with the token that DOKU gives in the status response.

The javascript with the additional parameter will generate the following payment form, which enables the customer to save their credit card, for faster payment. When a transaction is sent as a 2-Click payment to DOKU, in addition to the 4 fields for credit card data, DOKU will also display on the merchant website a tick box asking the customer’s approval to save the card.
2. When the customer has filled in their card details and clicked “Process Payment”, the data is sent to the DOKU server. Because the ‘data.reg_customer_id’ parameter has been added to the payment form, the DOKU server will create a token to pair with the Customer ID. If the customer checks the box next to “Save this Credit Card for faster checkout?” the payment response to the Merchant server will include this token. See example response:

```json
{
    "res_approval_code": "844647",
    "res_trans_id_merchant": "1706221359",
    "res_amount": "30000.00",
    "res_payment_date_time": "20160319114638",
    "res_verify_score": "-1",
    "res_verify_id": "",
    "res_verify_status": "NA",
    "res_words": "7553a51a09175a2462eb9150c7135f4a8d58ff616db022ca42e0ef65666ebf0",
    "res_response_msg": "SUCCESS",
    "res_mcn": "**************1111",
    "res_mid": "000100013000195",
    "res_bank": "JPMORGAN CHASE BANK",
    "res_response_code": "0000",
    "res_session_id": "4cf212f141a1d7fe672db93db75cc069,PRODUCTION",
    "res_payment_channel": "15",
    "res_bundle_token": {
        "res_token_payment": "0be4lclc653dbc8e1e6c24155c629fe237325a06",
        "res_token_msg": "SUCCESS",
        "res_token_code": "0000"
    }
}
```

When the payment response is received, store it in your database for the next payment using the 2-Click service.

3. After a successful first payment, (assuming that the merchant has been correctly storing the Token data) only a slight modification needs to be made to the javascript. Add the extra parameter ‘data.reg_token_payment’ as seen below, by using the token value that was obtained during the first payment:
The above script will generate the following payment form:

As you can see from the screenshot above, the customer no longer needs to fill out the credit card data apart from the CVV number. When the customer clicks the “Process Payment” button, it will follow the same process as regular card payments.

### 2.1.3 1-Click Payment [Recurring]

Using the same principles as 2-Click Payment, 1-Click payment takes it a step further and allows the customer to make a purchase with a single click on the website. This means that they can skip the process of inputting their card details, personal information, CVV number and 3D secure. The customer will have to enter the card details and complete the 3D secure verification process only during the first time they make a purchase. By eliminating the extra steps, you are able to create a more seamless and easy checkout process, which may lead to a lower drop-off rate. However, please note that this is subject to DOKU’s and the bank’s approval due to an increase in fraud risk. Please contact DOKU if you are interested to implement the 1-Click Payment feature.

Follow these steps to apply 1-click payment to your credit card payment process:

1. Insert the additional script to your server under the payment data.
2. Generate and save the token during the first payment.
3. For subsequent payments, retrieve the token from your database and send it to the DOKU server.
1. To initialize 1-click payment, follow the same steps as general credit card processing, but add the additional parameter “data.reg_customer_id” to your javascript under the payment data. The “data.reg_customer_id” parameter may represent the customer ID that you assign to each customer within your database. This ID will be paired with the token that DOKU gives in the status response.

The javascript with the additional parameter will generate the following payment form, which enables the customer to save their credit card.

```javascript
<script type="text/javascript">$(function() {
    var data = new Object();
    data.req_merchant_code = '1';
    data.req_chain_merchant = 'NA';
    data.req_payment_channel = '15';
    data.req_transaction_id = '<?php echo $invoice ?>';
    data.req_currency = '<?php echo $currency ?>';
    data.req_amount = '<?php echo $amount ?>';
    data.req_words = '<?php echo $words ?>';
    data.req_form_type = 'full';
    data.req_customer_id = '12124';
    getForm(data);
});</script>
```

2. When the customer has filled in their card details and clicked “Process Payment”, the data is sent to the DOKU server. Because the ‘data.reg_customer_id’ parameter has been added to the payment form, the DOKU server will create a token to pair with the Customer ID. If the customer checks the box next to “Save this credit card for faster checkout?”, the payment response to the Merchant server will include this token. See example response:
When the payment response is received, store it in your database for the next payment using the 1-Click service.

3. After a successful first payment, (assuming that the merchant has been correctly storing the Token data) only a slight modification needs to be made to the javascript. Add the extra parameter ‘data.reg_token_payment’ as seen below, by using the token value that was obtained during the first payment:

```javascript
var data = new Object();
  data.req_merchant_code = '1';
  data.req_chain_merchant = 'NA';
  data.req_payment_channel = '15';
  data.req_transaction_id = '<?php echo $invoice ?>';
  data.req_currency = '<?php echo $currency ?>';
  data.req_amount = '<?php echo $amount ?>';
  data.req_words = '<?php echo $words ?>';
  data.req_form_type = 'full';
  data.req_customer_id = '12124';
  data.req_token_payment = '0bea1c1c653dbc8e1e6c24155c629fe237325a06';
getForm(data);
```

The above script will not generate another payment form, but will instead just display the “Process Payment” button.

4. Once your customer has input their card data and clicked the “Process Payment” button, the data will be processed and DOKU will send a response containing the card ID. When you retrieve this data from your server, create the payment request following the instructions for regular credit card payments. However, when you make the payment request, you must change the method of payment to `Doku_Api::doDirectPayment` instead of `Doku_Api::doPayment`. See example:
2.1.4 Installment

Installment allows your customers to pay in plans using their credit cards. To enable this feature, you will need an agreement with the participating banks; please contact DOKU sales representative for assistance. This feature applies to both cards issued by banks that have their own installment product (on-us
installment) and those without installment product (off-us installment), each with different installment plans and rate. There is only a single integration process that you need to go through to enable both types of installment:

1. Create an installment form on your payment page

   Installment integration follows the same steps as a normal credit card payment, only with a few additional parameters. To start, please create an installment selection form on your payment page for your customer to choose their card issuer and preferred payment plan along with the doku.js credit card form. The installment form may look like this:

   ![Installment Form Example]

2. Send Payment Request

   There is no difference in the parameters sent from doku.js to your server, please do it as you would for a normal credit card payment. However, pair the credit card payment request parameters together with the installment information as you would need to combine the two when conducting the server-to-server payment charging process.

3. Charge payment to DOKU server along with the installment parameters

   Send a payment charge request to DOKU using the usual DOKU library, only with the additional installment parameters. **Installment_acquirer** refers to the acquirer bank code, **tenor** is the number of months in the installment plan, **plan_id** is a promotion code the merchant has form the bank, and **installment_off_us** with value ‘O’ only for off-us transaction.

```php
$dataPayment = array(    'req_mall_id' => Doku_Initiate::$mallId,    'req_chain_merchant' => 'NA',    'req_amount' => '10000.00',    'req_words' => $words,    'req_purchase_amount' => '10000.00',    'req_trans_id_merchant' => $invoice_no,    'req_currency' => '360',    'req_purchase_currency' => '360',    'req_session_id' => sha1(date('YmdHis')),    'req_name' => $customer['name'],    'req_payment_channel' => 15,    'req_basket' => $basket,    'req_email' => $customer['data_email'],    'req_token_id' => $token,    'req_installment_acquirer' => '100' => from merchant    'req_tenor' => '12', => from merchant    'req_plan_id' => '001' => from merchant    'req_installment_off_us' => '0' => only for off-us transaction)
```
2.1.5 Authorize & Capture

Authorize & Capture is a feature that allows you to block a certain amount from the customer’s credit card limit (Authorize), then hold it for a certain period before charging a payment – which can be a different amount from what you block (Capture). To enable this feature, you will need an agreement with the acquiring bank; please contact DOKU sales representative for more detail.

The integration process for Authorize & Capture works as follows:

1. Authorize

To send a payment authorization request, follow the exact same steps as a normal credit card payment charging request, with additional parameters that you send during the server-to-server process:

```php
$dataPayment = array(
    'req_mall_id' => Doku_Initiate::$mallId,
    'req_chain_merchant' => 'NA',
    'req_amount' => '10000.00',
    'req_words' => $words,
    'req_purchase_amount' => '10000.00',
    'req_trans_id_merchant' => $invoice_no,
    'req_request_date_time' => date('YmdHis'),
    'req_currency' => '360',
    'req_purchase_currency' => '360',
    'req_session_id' => sha1(date('YmdHis')),
    'req_name' => $customer['name'],
    'req_payment_channel' => 15,
    'req_basket' => $basket,
    'req_email' => $customer['data_email'],
    'req_token_id' => $token,
    'req_authorize_expiry_date' => '1440' // from merchant, per minute (optional)
    'req_payment_type' => 'A'
);
```

`req_authorize_expiry_date` refers to the time limit (in minutes) that you want to impose on authorizing a certain amount of the card limit; the maximum is 15 days. While the value 'A' for `req_payment_type` refers to an Authorize request. During this process DOKU will check whether there is still enough limit on the card to Authorize the amount.

After successfully completing the Authorize process, you will receive an approval_code parameter. Store and pair it with your `trans_id_merchant` parameter to later complete the Capture process.

2. Capture

To capture, simply send a server-to-server charging request to DOKU as shown by example below. You do not have to follow the full credit card steps since the customer does not have to fill in their card details anymore and getToken process can be bypassed.
3.0 DOKU Wallet

Doku Wallet integration comprises 3 easy steps:

1. Insert JavaScript
2. Initiate JavaScript parameters
3. Create payment form

To get started on your integration, follow these steps one by one by pasting the template scripts onto your website. The DOKU payment form is saved in HTML format; the template script acts as a placeholder for where the payment form will appear on your page. Prepare your HTML pages as seen in the template, and customize accordingly.

1. Insert the doku.js, fancybox.js and fancybox.css onto your website’s payment page, along with your custom style. See example:

```php
<?php
require_once('../Doku.php');
Doku_Initiate::$sharedKey = '<Put Your Shared Key Here>';
Doku_Initiate::$mallId = '<Put Your Merchant Code Here>';
$invoice = 'invoice_1458123951';
$params = array(
    'amount' => '10000.00',
    'invoice' => $invoice,
    'currency' => '360'
);
$words = doku_Library::doCreateWords($params);
?>
```

2. Initialize the payment form by creating the words using doku_library...
...and adding the following example script on to your webpage:

```javascript
$(function() {
  var data = new Object();
  data.req_merchant_code = '1';
  data.req_chain_merchant = 'NA';
  data.req_payment_channel = '04';
  data.req_transaction_id = '<?php echo $invoice ?>';
  data.req_currency = '<?php echo $currency ?>';
  data.req_amount = '<?php echo $amount ?>';
  data.req_words = '<?php echo $words ?>';
  data.req_form_type = 'full';

  getForm(data);
});
</script>

Remember to use ‘04’ for the payment_channel parameter, which refers to DOKU Wallet.

3. Insert the template script where you wish the payment form to appear on your website.

```html
<form action="charge.php" method="POST" id="payment-form">
  <div doku-div='form-payment'>
    <input id="doku-token" name="doku-token" type="hidden" />
  </div>
</form>
```

When completed correctly, the payment form should appear as seen below, and you may start receiving payments immediately. The DOKU script only provides the two fields to be filled in by the customer. All other parts of the payment page, including ‘Process Payment’ button are customizable to your needs.

![DOKU Payment Form](image)

Once your customer has input their DOKU ID and signed in to their account, their balance and payment options will appear in a pop-pup window, which follows the DOKU template, like this:
The customer may choose to pay with their existing cash balance and may apply a Voucher or Promo Code if he/she has a valid one. The customer may also choose to pay with the Credit Card that has been linked to their DOKU Wallet. Note however, that to activate the credit card option, further process is required. If the Credit Card option is not yet activated for you and you would like to do so, please contact DOKU.

When the customer clicks the “Process Payment” button, the DOKU server will process the data and send a response to the browser. The browser will then post the data to the merchant server, according to the URL set in your action form.
You can retrieve the posted data from the form within your server. Then send the payment request to the DOKU server to be processed using our library. See example:

```php
<?php
    require_once('../../Doku.php');

    Doku_Initiate::$sharedKey = '<Put Your Shared Key Here>';
    Doku_Initiate::$mallId = '<Put Your Merchant Code Here>';

    $token = $_POST['doku-token'];
    $pairing_code = $_POST['doku-pairing-code'];
    $invoice_no = $_POST['doku-invoice-no'];

    $params = array(
        'amount' => '10000.00',
        'invoice' => $invoice_no,
        'currency' => '360',
        'pairing_code' => $pairing_code,
        'token' => $token
    );
    $words = Doku_Library::doCreateWords($params);

    $basket[] = array(
        'name' => 'sayur',
        'amount' => '10000.00',
        'quantity' => '1',
        'subtotal' => '10000.00'
    );
    $customer = array(
        'name' => 'TEST NAME',
        'data_phone' => '081211111111',
        'data_email' => 'test@test.com',
        'data_address' => 'bojong gede #1 08/01'
    );
    $data = array(
        'req_token_id' => $token,
        'req_pairing_code' => $pairing_code,
        'req_customer' => $customer,
        'req_basket' => $basket,
        'req_words' => $words
    );

    $responsePrePayment = Doku_Api::doPrePayment($data);
    if($responsePrePayment->res_response_code == '0000'){  //prepayment success
        $dataPayment = array(
            'req_mall_id' => '1',
            'req_chain_merchant' => 'NA',
            'req_amount' => '10000.00',
            'req_words' => $words,
            'req_purchase_amount' => '10000.00',
            'req_trans_id_merchant' => $invoice_no,
            'req_request_date_time' => date('YmdHis'),
            'req_currency' => '360',
            'req_purchase_currency' => '360',
            'req_session_id' => sha1(date('YmdHis')),
            'req_name' => $customer['name'],
            'req_payment_channel' => 04,
            'req_basket' => $basket,
            'req_email' => $customer['data_email'],
            'req_token_id' => $token
        );

        $result = Doku_Api::doPayment($dataPayment);
        if($result->res_response_code == '0000'){
            echo 'SUCCESS';
        }else{
            echo 'FAILED';
        }
    }else{
        //prepayment fail
    }
```
After this, you may display the result on your browser for the customer to see.

![Transaction Success](image)

**4.0 Virtual Account**

DOKU Virtual Account aggregates the funds using 3 different entities – Bank Permata, Bank Sinarmas, Alfa Group and Indomaret. When the customer clicks ‘Process Payment’, DOKU will generate a one-time use, 11 digit payment code which is valid at any Prima, ALTO or Bersama ATM as well as all of Alfa Group’s and Indomaret convenience stores. For each of the different acquiring entities, the first 5 digit codes will define where the payment should be made. See codes below:

<table>
<thead>
<tr>
<th>Bank / Store</th>
<th>First 5 digit code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Permata</td>
<td>89650</td>
</tr>
<tr>
<td>Mandiri</td>
<td>88899</td>
</tr>
<tr>
<td>Sinarmas</td>
<td>88900</td>
</tr>
<tr>
<td>CIMB Niaga</td>
<td>51490</td>
</tr>
<tr>
<td>AlfaGroup</td>
<td>88888</td>
</tr>
<tr>
<td>Indomaret</td>
<td>88xxx</td>
</tr>
</tbody>
</table>

So a payment code that is valid for payment at an Alfa store would look like this: 88888-39421877483. And a bank transfer with Permata acquiring would look like this: 89650-39421877483.

Integration for ATM transfer and convenient store is practically identical; however, keep in mind that you will have to set the first 5 digits according to the payment method, and the last 11 digits will be queried from the DOKU server.

**4.1 Bank Transfer**

Follow these simple steps for Bank Transfer integration:

1. Generate Payment Code
2. Display Payment Code in your browser
3. Receive Payment Notification
4. Notify DOKU server that Payment Notification has been received
1. By using the DOKU PHP Library, you can make a payment code request with ease. The request process is performed host to host. Examples of the request is seen below:

```php
<?php
require_once ('../Doku.php');

date_default_timezone_set ('Asia/Jakarta');
Doku_Initiate::$sharedKey = <Put Your Shared Key Here>;
Doku_Initiate::$mallId = <Put Your Merchant Code Here>

$params = array(
    'amount' => $_POST['amount'],
    'invoice' => $_POST['trans_id'],
    'currency' => $_POST['currency']
);

$words = Doku_Library::doCreateWords($params);

$customer = array(
    'name' => 'TEST NAME',
    'data_phone' => '08121111111',
    'data_email' => 'test@test.com',
    'data_address' => 'bojong gede #1 08/01'
);

$dataPayment = array(
    'req_mall_id' => $_POST['mall_id'],
    'req_chain_merchant' => $_POST['chain_merchant'],
    'req_amount' => $params['amount'],
    'req_words' => $words,
    'req_trans_id_merchant' => $_POST['trans_id'],
    'req_purchase_amount' => $params['amount'],
    'req_request_date_time' => date('YmdHis'),
    'req_session_id' => sha1(date('YmdHis')),
    'req_email' => $customer['data_email'],
    'req_name' => $customer['name'],
    'req_basket' => 'sayur,10000.00,1,10000.00;',
    'req_address' => 'Plaza Asia Office Park Unit 3 Kav 59',
    'req_mobile_phone' => '081987987999',
    'req_expiry_time' => 'default 360 ' (optional)
);

$response = Doku_Api::doGeneratePaycode($dataPayment);
if($response->res_response_code == '0000'){
    echo 'GENERATE SUCCESS -- '
} else{
    echo 'GENERATE FAILED -- '
}
?>
```

The parameter `req_expiry_time` refers to the custom expiry window for the payment to be made. Exceeding this time limit will render the payment code invalid. You may set the time limit however you like, in minute format. If you do not set the expiry time parameter, DOKU will set it at the default time of 360 minutes (6 hours).

The DOKU server responds in JSON, like this:
2. Display the result in your browser however you wish. Remember to add the Permata/Sinarmas code in front of the payment code. If you want to choose on behalf of the merchant, which bank they are transferring to, the result can be displayed like this:

```json
{
    "res_pay_code": "62700000003",
    "res_pairing_code": "290316110837531987",
    "res_response_msg": "SUCCESS",
    "res_response_code": "0000"
}
```

Notes: for template, it depends on merchant requested.

Otherwise, you can display all three options like this:
3. Once the customer has made a payment, DOKU will send a payment notification containing the payment parameters to your server. The notification sent from DOKU will look something like this:

```
PAYMENTDATETIME=20160329110948
PURCHASECURRENCY=360
PAYMENTCHANNEL= FROM DOKU
AMOUNT=10000.00
PAYMENTCODE=00100000029
WORDS=01d9b362d3c1b80ff9196c6a565c49e5d9b03b8a
RESULTMSG=SUCCESS
TRANSIDMERCHANT=ZA912172
BANK=PERMATA
STATUSTYPE=P
APPROVALCODE=068992
RESPONSECODE=0000
SESSIONID=7b6647973dd13211a7fcf42eba79acea68aa69a1
```

4. Notify the DOKU server that you have received the payment notification, using the following example script:

```php
<?php

$PAYMENTDATETIME = $_POST['PAYMENTDATETIME'];
$PURCHASECURRENCY = $_POST['PURCHASECURRENCY'];
$PAYMENTCHANNEL = $_POST['PAYMENTCHANNEL'];
$AMOUNT = $_POST['AMOUNT'];
$PAYMENTCODE = $_POST['PAYMENTCODE'];
$WORDS = $_POST['WORDS'];
$RESULTMSG = $_POST['RESULTMSG'];
$TRANSIDMERCHANT = $_POST['TRANSIDMERCHANT'];
$BANK = $_POST['BANK'];
$STATUSTYPE = $_POST['STATUSTYPE'];
$APPROVALCODE = $_POST['APPROVALCODE'];
$RESPONSECODE = $_POST['RESPONSECODE'];
$SESSIONID = $_POST['SESSIONID'];

$WORDS_GENERATED = <function to generate words>
if ( $WORDS == $WORDS_GENERATED )
{
    echo "CONTINUE";
}
else
{
    echo "WORDS NOT MATCH";
}
?>
```
4.2 Convenience Store

Follow these simple steps for convenience store payment integration:

1. Generate Payment Code
2. Display Payment Code in your browser
3. Receive Payment Notification
4. Notify DOKU server that Payment Notification has been received

1. By using the DOKU PHP Library, you can make a payment code request with ease. The request process is performed host to host. Examples of the request is seen below:

```php
<?php
require_once('./Doku.php');

date_default_timezone_set('Asia/Jakarta');
Doku_Initiate::$sharedKey = '<Put Your Shared Key Here>';
Doku_Initiate::$mallId = '<Put Your Merchant Code Here>';

$params = array(
    'amount' => $_POST['amount'],
    'invoice' => $_POST['trans_id'],
    'currency' => $_POST['currency']
);

$words = Doku_Library::doCreateWords($params);

$customer = array(
    'name' => 'TEST NAME',
    'data_phone' => '081211111111',
    'data_email' => 'test@test.com',
    'data_address' => 'bojong gede #1 08/01'
);

$dataPayment = array(
    'req_mall_id' => Doku_Initiate::$mallId,
    'req_chain_merchant' => 'NA',
    'req_amount' => $params['amount'],
    'req_words' => $words,
    'req_trans_id_merchant' => $_POST['trans_id'],
    'req_purchase_amount' => $params['amount'],
    'req_request_date_time' => date('YmdHis'),
    'req_session_id' => sha1(date('YmdHis')),
    'req_email' => $customer['data_email'],
    'req_name' => $customer['name'],
    'req_basket' => 'sayur,10000.00,1,10000.00;',
    'req_address' => 'Plaza Asia Office Park Unit 3 Kav 59',
    'req_mobile_phone' => '081987987999'
    'req_expiry_time' => '60'
);

$response = Doku_Api::doGeneratePaycode($dataPayment);

if($response->res_response_code == '0000'){
    echo 'GENERATE SUCCESS -- ';}
else{
    echo 'GENERATE FAILED -- ';}
?>
```
The parameter ‘reg_expiry_time’ refers to the custom expiry window for the payment to be made. Exceeding this time limit will render the payment code invalid. You may set the time limit however you like, in minute format.

The DOKU server responds in JSON, as follows:

```
{
  "res_pay_code": "627000000003",
  "res_pairing_code": "290316110837531987",
  "res_response_msg": "SUCCESS",
  "res_response_code": "0000"
}
```

2. Display the result in your browser however you wish, but we also recommend you include a “How to Pay” section to inform the customer of the steps they need to complete at the cashier. Remember to add the Alfa or Indomaret code in front of the payment code. For example, the result can be displayed like this:

![Kode Pembayaran & Detail Transaksi Mini Market](image1)

![Invoice AMOUNT](image2)

![Indomaret Payment Code](image3)

<table>
<thead>
<tr>
<th>Payment Code</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>8822000000000214</td>
<td>IDR 10000.00</td>
</tr>
</tbody>
</table>

How to Pay at Indomaret

1. Take note of your payment code and go to your nearest Indomaret store.
2. Tell the cashier that you wish to make a DOKU payment.
3. If the cashier is unaware of DOKU, provide the instruction to
3. Once the customer has made a payment over the counter, DOKU will send a payment notification containing the payment parameters to your server. The notification sent from DOKU will look something like this:

```
PAYMENTDATETIME=20160329110948
PURCHASECURRENCY=360
PAYMENTCHANNEL=From Channel DoKu
AMOUNT=10000.00
PAYMENTCODE=00100000029
WORDS=01d9b362d3c1b80ff9196c6a565c49e5d9b03b8a
RESULTMSG=SUCCESS
TRANSIDMERCHANT=ZA912172
BANK=ALFA
STATUSTYPE=P
APPROVALCODE=068992
RESPONSECODE=0000
SESSIONID=7b6647973dd13211a7fcf42eba79ace68aa69a1
```

```
PAYMENTDATETIME=20160329110948
PURCHASECURRENCY=360
PAYMENTCHANNEL=From Channel DoKu
AMOUNT=10000.00
PAYMENTCODE=00100000029
WORDS=01d9b362d3c1b80ff9196c6a565c49e5d9b03b8a
RESULTMSG=SUCCESS
TRANSIDMERCHANT=ZA912172
BANK=INDOMARET
STATUSTYPE=P
APPROVALCODE=068992
RESPONSECODE=0000
SESSIONID=7b6647973dd13211a7fcf42eba79ace68aa69a1
```

4. Notify the DOKU server that you have received the payment notification, using the following example script:

```php
<?php

$PAYMENTDATETIME = $_POST['PAYMENTDATETIME'];
$PURCHASECURRENCY = $_POST['PURCHASECURRENCY'];
$PAYMENTCHANNEL = $_POST['PAYMENTCHANNEL'];
$AMOUNT = $_POST['AMOUNT'];
$PAYMENTCODE = $_POST['PAYMENTCODE'];
$WORDS = $_POST['WORDS'];
$RESULTMSG = $_POST['RESULTMSG'];
$TRANSIDMERCHANT = $_POST['TRANSIDMERCHANT'];
$BANK = $_POST['BANK'];
$STATUSTYPE = $_POST['STATUSTYPE'];
$APPROVALCODE = $_POST['APPROVALCODE'];
$RESPONSECODE = $_POST['RESPONSECODE'];
$SESSIONID = $_POST['SESSIONID']

$WORDS_GENERATED = <function to generate words>

if ( $WORDS == $WORDS_GENERATED )
{
    echo "CONTINUE";
}
else
{
    echo "WORDS NOT MATCH";
}
?>```
5.0 Internet Banking

Each bank has its own flow and authentication process for Internet Banking Payments. The majority of Internet Banking is hosted on the respective banks’ own webpages where the customer enters his/her credentials and completes the authentication. So even though the initial payment steps will occur on the merchant page, it is redirected to the bank page eventually. Out of the Internet Banking facilities that are supported by DOKU currently, only Mandiri Clickpay allows for a merchant hosted flow.

5.1 Mandiri Clickpay

Mandiri Clickpay integration comprises 3 easy steps:

1. Insert JavaScript
2. Add JavaScript code
3. Create payment form

To get started on your integration, follow these steps one by one by pasting the template scripts onto your website. The DOKU payment form is saved in HTML format; the template script acts as a placeholder for where the payment form will appear on your page. Prepare your HTML pages as seen in the template, and customize accordingly.

The form you create in this step will receive the Mandiri Clickpay input from the customer.

1. Insert the doku.js onto your website’s payment page, along with your custom style. See example:

   ```html
   <script src="doku.js"></script>
   <link href="http://staging.doku.com/doku-js/assets/css/doku.css" rel="stylesheet">
   <script src='http://staging.doku.com/doku-js/assets/js/jquery.payment.min.js'></script>
   ```

2. Initialize the payment form by adding the following example script onto your webpage.

   ```html
   <script type="text/javascript">
   jQuery(function($) {
   $('.cc-number').payment('formatCardNumber');
   var challenge3 = Math.floor(Math.random() * 999999999);
   $('#CHALLENGE_CODE_3').val(challenge3);
   });
   
   $(function() {
   var data = new Object();
   data.req_cc_field = 'cc_number';
   data.req_challenge_field = 'CHALLENGE_CODE_1';
   dokuMandiriInitiate(data);
   });
   </script>
   ```

3. Insert the template script where you wish the payment form to appear on your website.


Mandiri ClickPay WORDS = sha1
[AMOUNT+MALLID+<shared key> +TRANSIDMERCHANT+ CURRENCY]

<form method="post" action="../example/mandiri-clickpay-charge.php">
    <div id="mandiriclickpay" class="channel">
        <div class="list-chacode">
            <ul>
                <li>
                    <div class="text-chacode">Challenge Code 1</div>
                    <input type="text" id="CHALLENGE_CODE_1" name="CHALLENGE_CODE_1" readonly="true" required/>
                </li>
                <li>
                    <div class="text-chacode">Challenge Code 2</div>
                    <div class="num-chacode" id="challenge_div_2">0000100000</div>
                    <input type="hidden" name="CHALLENGE_CODE_2" value="0000100000"/>
                </li>
                <li>
                    <div class="text-chacode">Challenge Code 3</div>
                    <input type="hidden" name="CHALLENGE_CODE_3" value=""/>
                </li>
            </ul>
        </div>
        <div class="validasi">
            <div class="styled-input fleft width50">
                <input type="text" required="" name="response_token">
                <label>Response Token</label>
            </div>
        </div>
        <input type="hidden" name="invoice_no" value="invoice_1458547984">
        <input type="button" value="Process Payment" class="default-btn" onclick="this.form.submit();">
    </div>
</form>
When completed correctly, the payment form should appear as seen below, and you may start receiving payments using Mandiri Clickpay immediately.

![Mandiri Clickpay](image)

Once your customer has input their Mandiri Clickpay data and clicked the “Process Payment” button, the data will be processed on DOKU. The DOKU server responds to the browser, and the data will be posted to the merchant server, according to the URL set in your action form.

You can retrieve the posted data from the form within your server. Then send the payment request to the DOKU server to be processed using our library. See example:
<?php
require_once(../Doku.php);

Doku_Initiate::$sharedKey = "<Put Your Shared Key Here>";
Doku_Initiate::$mallId = "<Put Your Merchant Code Here>";

$params = array(
    'amount' => '100000.00',
    'invoice' => $_POST['invoice_no']
);

$cc = str_replace(-, '', $_POST['cc_number']);

$words = Doku_Library::doCreateWords($params);

$customer = array(
    'name' => 'TEST NAME',
    'data_phone' => '081211111111',
    'data_email' => 'test@test.com',
    'data_address' => 'bojong gede #1 08/01'
);

$basket[] = array(
    'name' => 'sayur',
    'amount' => '10000.00',
    'quantity' => '1',
    'subtotal' => '10000.00'
);

$basket[] = array(
    'name' => 'buah',
    'amount' => '10000.00',
    'quantity' => '1',
    'subtotal' => '10000.00'
);

$dataPayment = array(
    'req_mall_id' => '1',
    'req_chain_merchant' => 'NA',
    'req_amount' => $params['amount'],
    'req_words' => $words,
    'req_purchase_amount' => $params['amount'],
    'req_trans_id_merchant' => $_POST['invoice_no'],
    'req_request_date_time' => date('YmdHis'),
    'req_currency' => '360',
    'req_purchase_currency' => '360',
    'req_session_id' => sha1(date('YmdHis')),
    'req_name' => $customer['name'],
    'req_payment_channel' => '02',
    'req_email' => $customer['data_email'],
    'req_card_number' => $cc,
    'req_basket' => $basket,
    'req_challenge_code_1' => $_POST['CHALLENGE_CODE_1'],
    'req_challenge_code_2' => $_POST['CHALLENGE_CODE_2'],
    'req_challenge_code_3' => $_POST['CHALLENGE_CODE_3'],
    'req_response_token' => $_POST['response_token'],
    'req_mobile_phone' => $customer['data_phone'],
    'req_address' => $customer['data_address']
);

$response = Doku_Api::doDirectPayment($dataPayment);

if($response->res_response_code == '0000'){
    echo 'PAYMENT SUCCESS -- ';}
else{
    echo 'PAYMENT FAILED -- ';}

var_dump($response);
?>

The DOKU server will send a response in JSON, like this:
After this is done, you may display the result on your browser for the customer to see.

```json
{
    "res_response_msg": "SUCCESS",
    "res_transaction_code": "d4efbb8c4ebb9a3597c05aa32f2b341e77f98e63",
    "res_mcn": "4***********1111",
    "res_approval_code": "1234",
    "res_trans_idMerchant": "1706332101",
    "res_payment_date": "20160319184828",
    "res_bank": "MANDIRI CLICK PAY",
    "res_amount": "30000.00",
    "res_message": "PAYMENT APPROVED",
    "res_response_code": "0000",
    "res_session_id": "50d240541f4d8d7565b18cb5ca93a660"
}
```
### 6.0 Appendix

#### 6.1 Payment Methods

<table>
<thead>
<tr>
<th>Payment Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Credit Card</strong></td>
<td>• Visa, Mastercard and AMEX for Overseas Partner. JCB upon request&lt;br&gt;• Direct API available&lt;br&gt;• Features (acquirer dependant): 3D and non 3D Secure, recur, authorize capture, online refund, tokenization, installment, BIN filtering, point rewards</td>
</tr>
<tr>
<td><strong>Internet Banking</strong></td>
<td>• Available: Mandiri Clickpay,&lt;br&gt;• Each bank has different authentication process through OTP or token&lt;br&gt;• Direct API only available for Mandiri Clickpay. The rest is re-direct only</td>
</tr>
<tr>
<td><strong>DOKU Wallet</strong></td>
<td>• E-wallet product issued by DOKU&lt;br&gt;• Source of fund: cash balance, linked credit card, linked bank account&lt;br&gt;• Max. balance is Rp1,000,000 for non-KYC and Rp10,000,000 for KYC users&lt;br&gt;• Authenticate with email, password and static PIN that is pre-set by the user&lt;br&gt;• Direct API available</td>
</tr>
<tr>
<td><strong>Convenience Store</strong></td>
<td>• Accessible in almost 12,000 Alfa group stores (Alfa Express, Alfa Midi, Alfa Mart, Lawson and DAN+DAN) and 14,000 Indomaret stores&lt;br&gt;• Generate 16 digit payment code at checkout, user goes to nearest store and makes payment over the counter with cash or non-cash&lt;br&gt;• Max. cash transaction value of Rp5,000,000&lt;br&gt;• Merchant can set payment code expiry time for every transaction&lt;br&gt;• Direct API available</td>
</tr>
<tr>
<td><strong>Bank Transfer</strong></td>
<td>• Virtual account housed in Bank Permata, Mandiri or SinarMas but payable from any bank that is connected to ATM Bersama, Prima or Alto networks (over 120 banks in Indonesia)&lt;br&gt;• Generate 16 digit payment code at checkout, user makes payment via ATM or Internet/mobile banking that is connected to 1 of the 3 networks&lt;br&gt;• Merchant can set payment code expiry time for every transaction&lt;br&gt;• Direct API available</td>
</tr>
</tbody>
</table>

#### 6.2 Payment Channel Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>02</td>
<td>Mandiri ClickPay</td>
</tr>
<tr>
<td>04</td>
<td>DOKU Wallet</td>
</tr>
<tr>
<td>05</td>
<td>ATM Permata VA LITE</td>
</tr>
<tr>
<td>06</td>
<td>BRI e-Pay</td>
</tr>
<tr>
<td>07</td>
<td>ATM Permata VA</td>
</tr>
<tr>
<td>14</td>
<td>AlfaGroup</td>
</tr>
<tr>
<td>15</td>
<td>Credit Card Visa/Master Multi Currency</td>
</tr>
<tr>
<td>16</td>
<td>Credit Card Tokenization</td>
</tr>
<tr>
<td>17</td>
<td>Recurring Payment</td>
</tr>
<tr>
<td>18</td>
<td>KlikPayBCA</td>
</tr>
<tr>
<td>19</td>
<td>CIMB Clicks</td>
</tr>
<tr>
<td>21</td>
<td>Sinarmas VA Full</td>
</tr>
<tr>
<td>22</td>
<td>Sinarmas VA Lite</td>
</tr>
<tr>
<td>23</td>
<td>MOTO</td>
</tr>
</tbody>
</table>
### 6.3 Parameters Required

In this section of the Appendix, you will find the list of required parameters for the different APIs.

#### 6.3.1 Payment Request

**Credit Card Payment Request Parameter**

<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Comments</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>req_mall_id</td>
<td>N</td>
<td></td>
<td>Given by DOKU</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>req_chain_merchant</td>
<td>N</td>
<td></td>
<td>Given by DOKU, if not using Chain, default value is NA</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>req_amount</td>
<td>N</td>
<td>12.2</td>
<td>Total amount. Eg:10000.00</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>req_purchase_amount</td>
<td>N</td>
<td>12.2</td>
<td>Total amount. Eg:10000.00</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>req_trans_id_merchant</td>
<td>AN</td>
<td>...30</td>
<td>Transaction ID from Merchant</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>req_words</td>
<td>AN</td>
<td>...200</td>
<td>Hashed key combination encryption (use SHA1 method). The hashed key generated from combining these parameters value in this order: Amount + Mall ID+ &lt;shared key&gt; + Invoice No. + Purchase Currency + Token ID + Pairing Code</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>req_request_date_time</td>
<td>N</td>
<td>X</td>
<td>YYYYMMDHHMMSS</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>req_currency</td>
<td>N</td>
<td>3</td>
<td>ISO3166 , numeric code</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>req_purchase_currency</td>
<td>N</td>
<td>3</td>
<td>ISO3166 , numeric code</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>req_session_id</td>
<td>AN</td>
<td>...48</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>req_name</td>
<td>AN</td>
<td>...50</td>
<td>Travel arranger name / buyer name</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>req_email</td>
<td>ANS</td>
<td>...100</td>
<td>Customer email</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>req_basket</td>
<td>ANS</td>
<td>...1024</td>
<td>Show transaction description. Use comma to separate each field and semicolon for each item. Item 1, 1000.00,2,20000.00;item2,15000.00,2,30000.00</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>token_id</td>
<td>AN</td>
<td></td>
<td>Sent by DOKU during getToken process</td>
<td>Yes</td>
</tr>
<tr>
<td>15</td>
<td>req_shipping_address</td>
<td>ANS</td>
<td>...100</td>
<td>Shipping address contains street and number</td>
<td>Optional</td>
</tr>
<tr>
<td>16</td>
<td>req_shipping_city</td>
<td>ANS</td>
<td>...100</td>
<td>City name</td>
<td>Optional</td>
</tr>
<tr>
<td>17</td>
<td>req_shipping_state</td>
<td>AN</td>
<td>...100</td>
<td>State / province name</td>
<td>Optional</td>
</tr>
<tr>
<td>18</td>
<td>req_shipping_country</td>
<td>A</td>
<td>2</td>
<td>ISO3166 , alpha-2</td>
<td>Optional</td>
</tr>
<tr>
<td>19</td>
<td>req_shipping_zip_code</td>
<td>N</td>
<td>...10</td>
<td>Zip Code</td>
<td>Optional</td>
</tr>
<tr>
<td>20</td>
<td>req_payment_channel</td>
<td>N</td>
<td>2</td>
<td>See payment channel code list</td>
<td>Optional</td>
</tr>
<tr>
<td>21</td>
<td>req_address</td>
<td>ANS</td>
<td>...100</td>
<td>Home address contains street and number</td>
<td>Optional</td>
</tr>
<tr>
<td>22</td>
<td>req_city</td>
<td>ANS</td>
<td>...100</td>
<td>City name</td>
<td>Optional</td>
</tr>
<tr>
<td>23</td>
<td>req_state</td>
<td>ANS</td>
<td>...100</td>
<td>State / province name</td>
<td>Optional</td>
</tr>
<tr>
<td>24</td>
<td>req_country</td>
<td>A</td>
<td>2</td>
<td>ISO3166 , alpha-2</td>
<td>Optional</td>
</tr>
<tr>
<td>25</td>
<td>req_zip_code</td>
<td>N</td>
<td>...10</td>
<td>Zip Code</td>
<td>Optional</td>
</tr>
<tr>
<td>26</td>
<td>req_mobile_phone</td>
<td>ANS</td>
<td>...11</td>
<td>Home Phone</td>
<td>Yes</td>
</tr>
<tr>
<td>27</td>
<td>req_work_phone</td>
<td>ANS</td>
<td>...13</td>
<td>Work Phone / Office Phone</td>
<td>Optional</td>
</tr>
<tr>
<td>28</td>
<td>req_birth_date</td>
<td>N</td>
<td>...8</td>
<td>YYYYMMDD</td>
<td>Optional</td>
</tr>
</tbody>
</table>
6.3.2 Check Status Request
Check transaction status for all type of payment channel

<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Comments</th>
<th>Mandatory</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>req_mall_id</td>
<td>N</td>
<td></td>
<td>Given by DOKU</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>req_chain_merchant</td>
<td>N</td>
<td></td>
<td>Given by DOKU, if not using Chain, default value is NA</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>req_trans_id_merchant</td>
<td>AN</td>
<td>...30</td>
<td>Transaction ID from merchant</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>req_session_id</td>
<td>AN</td>
<td>...48</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>req_words</td>
<td>AN</td>
<td>...200</td>
<td>Hashed key combination encryption (use SHA1 method). The hashed key is generated from combining the parameter values in this order : req_mall_id+&lt;shared key&gt;+ req_trans_id_merchant</td>
<td>Yes</td>
</tr>
</tbody>
</table>

6.3.2 Check Status Response
Response of check status request will be in JSON format. Below is the sample:

```json
{
  "res_currency": "360",
  "res_status_type": "P",
  "res_threed_secure_status": "",
  "res_approval_code": "",
  "res_purchase_currency": "360",
  "res Edu_status": "NA",
  "res_trans_id_merchant": "1494246982571",
  "res_amount": "500000.00",
  "res_payment_date_time": "20170508194517",
  "res_verify_score": "-1",
  "res_verify_id": "",
  "res_verify_status": "NA",
  "res_words": "08009f077a52d54dfa0443d58f99ec34227ddd49",
  "res_liability": "",
  "res_result_msg": "SUCCESS",
  "res_mcc": "0000000064",
  "res_bank": "DANAMON",
  "res_response_code": "0000",
  "res_payment_code": "891100000000064",
  "res_session_id": "1494246982571",
  "res_payment_channel": "33",
  "res_brand": "" //for CC
}
```
### 6.2 DOKU Response Codes

In this section of the Appendix, you will find the list of response codes and their description for the different payment methods.

#### 6.2.1 General response codes

The response codes listed in this section include both prepayment and payment response codes, and mostly apply to all payment methods. These are the most common response codes you will receive from DOKU.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>Successful approval</td>
</tr>
<tr>
<td>5555</td>
<td>Failed decrypt</td>
</tr>
<tr>
<td>5501</td>
<td>Payment channel not registered</td>
</tr>
<tr>
<td>5502</td>
<td>Merchant is disabled</td>
</tr>
<tr>
<td>5503</td>
<td>Maximum attempt 3 times</td>
</tr>
<tr>
<td>5504</td>
<td>Words not match</td>
</tr>
<tr>
<td>5505</td>
<td>Invalid parameter</td>
</tr>
<tr>
<td>5506</td>
<td>Notify failed</td>
</tr>
<tr>
<td>5507</td>
<td>Invalid parameter detected / Customer click cancel process</td>
</tr>
<tr>
<td>5508</td>
<td>Re-enter transaction</td>
</tr>
<tr>
<td>5509</td>
<td>Payment code already expired</td>
</tr>
<tr>
<td>5510</td>
<td>Cancel by Customer</td>
</tr>
<tr>
<td>5511</td>
<td>Not an error, payment code has not been paid by Customer</td>
</tr>
<tr>
<td>5512</td>
<td>Insufficient Parameter</td>
</tr>
<tr>
<td>5514</td>
<td>Reject by Fraud System</td>
</tr>
<tr>
<td>5515</td>
<td>Duplicate PNR</td>
</tr>
<tr>
<td>5516</td>
<td>Transaction Not Found</td>
</tr>
<tr>
<td>5517</td>
<td>Error in Authorization process</td>
</tr>
<tr>
<td>5518</td>
<td>Error parsing XML</td>
</tr>
<tr>
<td>5519</td>
<td>Customer stop at 3D Secure page</td>
</tr>
<tr>
<td>5520</td>
<td>Transaction Failed via scheduler</td>
</tr>
<tr>
<td>5521</td>
<td>Invalid Merchant</td>
</tr>
<tr>
<td>5522</td>
<td>Rates were not found</td>
</tr>
<tr>
<td>5523</td>
<td>Failed to get Transaction status</td>
</tr>
<tr>
<td>5524</td>
<td>Failed to void transaction</td>
</tr>
<tr>
<td>5525</td>
<td>Transaction can not be process</td>
</tr>
<tr>
<td>5526</td>
<td>Transaction is voided because timeout to wallet</td>
</tr>
<tr>
<td>5527</td>
<td>Transaction will be process as Off Us Instalment</td>
</tr>
<tr>
<td>5529</td>
<td>Invalid Merchant</td>
</tr>
<tr>
<td>5530</td>
<td>Internal server error</td>
</tr>
<tr>
<td>5531</td>
<td>Pairing Code does not exist</td>
</tr>
<tr>
<td>5532</td>
<td>Invalid Payment Channel</td>
</tr>
<tr>
<td>5533</td>
<td>Failed to inquiry list of fund</td>
</tr>
<tr>
<td>5534</td>
<td>Invalid Pairing Code</td>
</tr>
<tr>
<td>5535</td>
<td>Invalid Token</td>
</tr>
</tbody>
</table>
### 6.2.2 Credit Card

The response codes in this section only apply to credit card transactions.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>VISA</th>
<th>MASTERCARD</th>
<th>ORIGIN</th>
<th>ACTIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>Refer to card issuer</td>
<td>Refer to card issuer</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
</tr>
<tr>
<td>0002</td>
<td>Refer to card issuer, special condition</td>
<td>Refer to card issuer, special condition</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
</tr>
<tr>
<td>0003</td>
<td>Invalid merchant or service provider</td>
<td>Invalid Merchant</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or acquiring bank</td>
</tr>
<tr>
<td>0004</td>
<td>Pickup card</td>
<td>Capture card</td>
<td>VISA/MASTER</td>
<td>Should consider blocking the card temporarily or Block login ID</td>
</tr>
<tr>
<td>0005</td>
<td>Do Not Honor</td>
<td>Do Not Honor</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
</tr>
<tr>
<td>0006</td>
<td>Error</td>
<td></td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
</tr>
<tr>
<td>0007</td>
<td>Pickup card, special condition [other than lost/stolen card]</td>
<td></td>
<td>VISA/MASTER</td>
<td>Should consider blocking the card</td>
</tr>
<tr>
<td>0008</td>
<td></td>
<td>Honor with ID</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
</tr>
<tr>
<td>0010</td>
<td>Partial Approval - Private label</td>
<td></td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
</tr>
<tr>
<td>0011</td>
<td>VIP Approval</td>
<td></td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
</tr>
<tr>
<td>0012</td>
<td>Invalid Transaction</td>
<td>Invalid Transaction</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>Code</td>
<td>Error Description</td>
<td>VISA/MASTER</td>
<td>ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>----------------------------------------------------------------------------------</td>
<td>-------------</td>
<td>----------------</td>
<td></td>
</tr>
<tr>
<td>0013</td>
<td>Invalid amount (currency conversion field overflow, Visa Cash - Invalid load mount)</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0014</td>
<td>Invalid Account number (no such number)</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0015</td>
<td>No such issuer</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0019</td>
<td>Re-enter transaction</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0021</td>
<td>No Action taken (unable to back out prior transaction)</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0025</td>
<td>Unable to locate record in file, or account number is missing from inquiry</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0028</td>
<td>File is temporarily unavailable</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0030</td>
<td>-</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0041</td>
<td>Pickup card (lost card)</td>
<td>VISA/MASTER</td>
<td>Should consider blocking the card temporarily or Block login ID</td>
<td></td>
</tr>
<tr>
<td>0043</td>
<td>Pickup card (stolen card)</td>
<td>VISA/MASTER</td>
<td>Should consider blocking the card temporarily or Block login ID</td>
<td></td>
</tr>
<tr>
<td>0051</td>
<td>Insufficient funds</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0052</td>
<td>No checking account</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0053</td>
<td>non savings account</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0054</td>
<td>Expired card</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0055</td>
<td>Incorrect PIN (Visa cash - invalid or missing SI signature)</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0057</td>
<td>Transaction not permitted to cardholder (Visa cash - incorrect routing, not a load request)</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0058</td>
<td>Transaction not allowed at terminal</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0061</td>
<td>Activity amount limit exceeded</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Error Description</td>
<td>Card Types</td>
<td>Action</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>----------------------------------------------------------------------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>0062</td>
<td>Restricted card (for example in country exclusion table)</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0063</td>
<td>Security violation</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0065</td>
<td>Activity count limit exceeded</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0075</td>
<td>Allowable number of PIN-entry tries exceeded</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0076</td>
<td>Unable to locate previous message (no match on Retrieval Reference number)</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0077</td>
<td>Previous message located for a repeat or reversal, but repeat or reversal data are inconsistent with original message</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0080</td>
<td>Invalid date (For use in private label card transactions and check acceptance transactions)</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0081</td>
<td>PIN Cryptographic error found [error found by VIC security module during PIN decryption]</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0082</td>
<td>Incorrect CW/1CW</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0083</td>
<td>Unable to verify PIN</td>
<td>VISA/MASTER</td>
<td>Tell Customer to contact the Bank Issuer of the card used.</td>
<td></td>
</tr>
<tr>
<td>0084</td>
<td>-</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0085</td>
<td>No reason to decline a request for account number verification or address verification</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0091</td>
<td>Issuer unavailable or switch inoperative (STIP not applicable or available for this transaction)</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0092</td>
<td>Destination cannot be found for routing</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0093</td>
<td>Transaction cannot be completed; violation of law</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0094</td>
<td>Duplicate transmission detected</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>0096</td>
<td>System malfunction or certain field error conditions</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Issuance</td>
<td>Contact</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------</td>
<td>----------</td>
<td>---------</td>
<td></td>
</tr>
<tr>
<td>00NO</td>
<td>Force STIP</td>
<td>VISA/MASTER</td>
<td>DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>00N3</td>
<td>Cash service not available</td>
<td>VISA/MASTER</td>
<td>DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>00N4</td>
<td>Cash request exceeds issuer limit</td>
<td>VISA/MASTER</td>
<td>DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>00N7</td>
<td>Decline for CW2 failure</td>
<td>VISA/MASTER</td>
<td>DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>00P2</td>
<td>Invalid biller information</td>
<td>VISA/MASTER</td>
<td>DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>00P5</td>
<td>PIN Change/Unblock request declined</td>
<td>VISA/MASTER</td>
<td>DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>00P6</td>
<td>Unsafe PIN</td>
<td>VISA/MASTER</td>
<td>DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>00TO</td>
<td>Timeout / Transaction's response exceed time limit</td>
<td>DOKU</td>
<td>DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td>00UE</td>
<td>Unknown Exception / PosServer not responding</td>
<td>DOKU</td>
<td>DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
</tbody>
</table>
### 6.2.3 DOKU Wallet

The response codes in this section only apply to DOKU Wallet transactions.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0E01</td>
<td>FAILED GET MERCHANT</td>
</tr>
<tr>
<td>0E02</td>
<td>MASTER MERCHANT INACTIVE</td>
</tr>
<tr>
<td>0E03</td>
<td>INVALID WORDS FROM MERCHANT</td>
</tr>
<tr>
<td>0E04</td>
<td>INVALID MERCHANT</td>
</tr>
<tr>
<td>0E05</td>
<td>FAILED TO PROCESS PAYMENT</td>
</tr>
<tr>
<td>0E06</td>
<td>PAYMENT METHOD NOT DEFINE</td>
</tr>
<tr>
<td>0E07</td>
<td>FAILED EXECUTE PRE AUTH PLUGINS</td>
</tr>
<tr>
<td>0E08</td>
<td>FAILED EXECUTE POST AUTH PLUGINS</td>
</tr>
<tr>
<td>0E09</td>
<td>INVALID PAY ID</td>
</tr>
<tr>
<td>0E10</td>
<td>ERROR PAY ID</td>
</tr>
<tr>
<td>0E11</td>
<td>FAILED EXECUTE PRE TRANS MIP PLUGINS</td>
</tr>
<tr>
<td>0E12</td>
<td>VERIFY RESPONSE STOP FROM MERCHANT</td>
</tr>
<tr>
<td>0E13</td>
<td>FAILED VERIFY TO MERCHANT</td>
</tr>
<tr>
<td>0E14</td>
<td>FAILED SEND PAYMENT CASH WALLET</td>
</tr>
<tr>
<td>0E15</td>
<td>NOTIFY RESPONSE STOP FROM MERCHANT</td>
</tr>
<tr>
<td>0E16</td>
<td>FAILED NOTIFY TO MERCHANT</td>
</tr>
<tr>
<td>0E18</td>
<td>FAILED EXECUTE POST TRANS MIP PLUGINS</td>
</tr>
<tr>
<td>0E19</td>
<td>NOT ENOUGH CASH BALANCE AND DON’T HAVE CREDIT CARD</td>
</tr>
<tr>
<td>0E20</td>
<td>SPENDER NO HAVE LINK TO CREDIT CARD</td>
</tr>
<tr>
<td>0E21</td>
<td>ERROR CHECK 3D SECURE CREDIT CARD</td>
</tr>
<tr>
<td>0E22</td>
<td>PIN/OTP IS NOT VALID</td>
</tr>
<tr>
<td>0E23</td>
<td>PLEASE INPUT CVV2</td>
</tr>
<tr>
<td>0E24</td>
<td>INVALID SESSION</td>
</tr>
<tr>
<td>0E25</td>
<td>FAILED SEND LINK AUTHENTICATION TO CARD HOLDER</td>
</tr>
<tr>
<td>0E26</td>
<td>INSUFFICIENT PARAMS</td>
</tr>
<tr>
<td>0E27</td>
<td>FAILED EXECUTE PRE TRANS CIP PLUGINS</td>
</tr>
<tr>
<td>0E28</td>
<td>FAILED EXECUTE POST TRANS CIP PLUGINS</td>
</tr>
<tr>
<td>0E29</td>
<td>FAILED SEND PAYMENT MIP CREDIT CARD</td>
</tr>
<tr>
<td>0E30</td>
<td>YOU DO NOT HAVE PIN</td>
</tr>
<tr>
<td>0E31</td>
<td>DUPLICATE INVOICE NO</td>
</tr>
<tr>
<td>0E32</td>
<td>URL NOT FOUND</td>
</tr>
<tr>
<td>0E33</td>
<td>CUSTOMER NOT FOUND</td>
</tr>
<tr>
<td>0E34</td>
<td>VOID PROCESS FAILED</td>
</tr>
<tr>
<td>0E35</td>
<td>Failed Send ONE TIME PIN to your email</td>
</tr>
<tr>
<td>0E36</td>
<td>Failed Send Link for create PIN to your email</td>
</tr>
<tr>
<td>0E37</td>
<td>THIS SPENDER CAN’T TRANSACT IN THIS MERCHANT</td>
</tr>
<tr>
<td>0E38</td>
<td>You have reach your DOKU ID Transaction Limit</td>
</tr>
<tr>
<td>0E39</td>
<td>Process MIP Transaction Failed</td>
</tr>
<tr>
<td>0E99</td>
<td>ERROR SYSTEM</td>
</tr>
</tbody>
</table>
### 6.2.4 Virtual Account
The response codes in this section only apply to Convenience Store and Bank Transfer transactions.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>Decline (internal error)</td>
</tr>
<tr>
<td>0013</td>
<td>Invalid amount</td>
</tr>
<tr>
<td>0014</td>
<td>Bill not found</td>
</tr>
<tr>
<td>0066</td>
<td>Decline</td>
</tr>
<tr>
<td>0088</td>
<td>Bill already paid</td>
</tr>
</tbody>
</table>

### 6.2.5 Mandiri Clickpay
The response codes in this section only apply to Mandiri Clickpay transactions.

<table>
<thead>
<tr>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001</td>
<td>Internal system error: cannot parse message</td>
</tr>
<tr>
<td>0002</td>
<td>Internal system error: unmatched signature hash</td>
</tr>
<tr>
<td>0003</td>
<td>Internal system error: Cannot process message</td>
</tr>
<tr>
<td>0004</td>
<td>Internal system error: Error on field</td>
</tr>
<tr>
<td>0005</td>
<td>Internal system error: Transaction not found</td>
</tr>
<tr>
<td>0006</td>
<td>Internal system error: Create VPA response error</td>
</tr>
<tr>
<td>0101</td>
<td>Internal system error: Create velis-authenticator message</td>
</tr>
<tr>
<td>0102</td>
<td>Internal system error: Runtime try/catch error when creating VTCPStream</td>
</tr>
<tr>
<td>0103</td>
<td>Internal system error: Cannot connect to velis-authenticator</td>
</tr>
<tr>
<td>0104</td>
<td>Internal system error: Send request to velis-authenticator failed</td>
</tr>
<tr>
<td>0105</td>
<td>Internal system error: Waiting response from velis-authenticator failed</td>
</tr>
<tr>
<td>0106</td>
<td>Internal system error: Read response from velis-authenticator failed</td>
</tr>
<tr>
<td>0107</td>
<td>Internal system error: Parse response from velis-authenticator failed</td>
</tr>
<tr>
<td>0108</td>
<td>Internal system error: Signature key from velis-authenticator is invalid</td>
</tr>
<tr>
<td>1101</td>
<td>User not registered: Channel not register in database (not found)</td>
</tr>
<tr>
<td>1102</td>
<td>User not registered: User not active</td>
</tr>
<tr>
<td>1103</td>
<td>User not registered: User has deleted</td>
</tr>
<tr>
<td>1104</td>
<td>User not registered: User not found</td>
</tr>
<tr>
<td>1105</td>
<td>User not registered: Channel for User not active</td>
</tr>
<tr>
<td>1106</td>
<td>User not registered: Channel for User has deleted - no access</td>
</tr>
<tr>
<td>1107</td>
<td>User not registered: Channel for User not register / not found</td>
</tr>
<tr>
<td>1108</td>
<td>User has blocked: User has disabled</td>
</tr>
<tr>
<td>1109</td>
<td>User has blocked</td>
</tr>
<tr>
<td>1110</td>
<td>User has blocked: Channel for User has disabled</td>
</tr>
<tr>
<td>1111</td>
<td>User has blocked: Channel for User has blocked</td>
</tr>
<tr>
<td>1112</td>
<td>User already activated: User has invalid status (or already active)</td>
</tr>
<tr>
<td>1113</td>
<td>User already activated: Channel for User has invalid status (or already active)</td>
</tr>
<tr>
<td>1114</td>
<td>Invalid token: Token of User not active</td>
</tr>
<tr>
<td>1115</td>
<td>Invalid token: Token of User has disable</td>
</tr>
<tr>
<td>1116</td>
<td>Invalid token: Token of User has deleted</td>
</tr>
<tr>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>1117</td>
<td>Invalid token: Token of User not found</td>
</tr>
<tr>
<td>1118</td>
<td>Invalid token: Method CR not allowed for Token of User</td>
</tr>
<tr>
<td>1119</td>
<td>Invalid token: Method RO not allowed for Token of User</td>
</tr>
<tr>
<td>1120</td>
<td>Invalid token: Method SG not allowed for Token of User</td>
</tr>
<tr>
<td>1121</td>
<td>Invalid token: Device Token Type not valid (only support VS = VASCO Token)</td>
</tr>
<tr>
<td>1122</td>
<td>Invalid token response: Code Not Verified</td>
</tr>
<tr>
<td>1123</td>
<td>Invalid token response: Code Replay Attempt</td>
</tr>
<tr>
<td>1124</td>
<td>Invalid token response: Challenge Too Small</td>
</tr>
<tr>
<td>1125</td>
<td>Invalid token response: Challenge Too Long</td>
</tr>
<tr>
<td>1126</td>
<td>Invalid token response: Challenge Check Digit Wrong (Host Check Challenge Mode)</td>
</tr>
<tr>
<td>1127</td>
<td>Invalid token response: Challenge Character Not Decimal</td>
</tr>
<tr>
<td>1128</td>
<td>Invalid token response: Challenge Corrupt (Host Check Challenge Mode)</td>
</tr>
<tr>
<td>1129</td>
<td>Invalid token response: Response Length Out of Bounds</td>
</tr>
<tr>
<td>1130</td>
<td>Invalid token response: Response Too Small</td>
</tr>
<tr>
<td>1131</td>
<td>Invalid token response: Response Too Long</td>
</tr>
<tr>
<td>1132</td>
<td>Invalid token response: Response Check Digit Wrong</td>
</tr>
<tr>
<td>1133</td>
<td>Invalid token response: Response Character Not Decimal</td>
</tr>
<tr>
<td>1134</td>
<td>Invalid token response: Response Character Not Hexadecimal</td>
</tr>
<tr>
<td>1135</td>
<td>Invalid token response: Token Authentication Failed</td>
</tr>
<tr>
<td>1199</td>
<td>Receive error response from VA</td>
</tr>
<tr>
<td>0201</td>
<td>Internal system error: Create DSP-ISO message failed</td>
</tr>
<tr>
<td>0202</td>
<td>Internal system error: No active DSPSession</td>
</tr>
<tr>
<td>0203</td>
<td>Internal system error: Cannot send request to DSP-Silverlake</td>
</tr>
<tr>
<td>0204</td>
<td>Internal system error: Waiting response from DSP-Silverlake</td>
</tr>
<tr>
<td>0205</td>
<td>Internal system error: Read response from DSP-Silverlake without bit 39</td>
</tr>
<tr>
<td>0206</td>
<td>Internal system error: Read response from DSP-Silverlake without bit 126</td>
</tr>
<tr>
<td>0207</td>
<td>Invalid card number: Card number not belong to this CIF</td>
</tr>
<tr>
<td>2101</td>
<td>Invalid card number: Card not found</td>
</tr>
<tr>
<td>2102</td>
<td>Not enough balance</td>
</tr>
<tr>
<td>2103</td>
<td>Invalid customer account</td>
</tr>
<tr>
<td>2104</td>
<td>DSP-Silverlake system error</td>
</tr>
<tr>
<td>2199</td>
<td>Receive error response from DSP-Silverlake</td>
</tr>
<tr>
<td>0301</td>
<td>Internal system error: Cannot connect to VAM</td>
</tr>
<tr>
<td>3101</td>
<td>Invalid XML request: Invalid data XML (tc)</td>
</tr>
<tr>
<td>3102</td>
<td>Invalid XML request: Invalid data XML (userid)</td>
</tr>
<tr>
<td>3103</td>
<td>Invalid XML request: Invalid data XML (trace number)</td>
</tr>
<tr>
<td>3104</td>
<td>Invalid XML request: Invalid data XML (reference number)</td>
</tr>
<tr>
<td>3105</td>
<td>Invalid XML request: Invalid data XML (datetime)</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>3106</td>
<td>Invalid XML request: Invalid data XML (merchantid)</td>
</tr>
<tr>
<td>3107</td>
<td>Invalid XML request: Invalid data XML (bankid)</td>
</tr>
<tr>
<td>3108</td>
<td>Invalid XML request: Invalid data XML (item detail)</td>
</tr>
<tr>
<td>3109</td>
<td>Invalid XML request: Invalid data XML (amount)</td>
</tr>
<tr>
<td>3110</td>
<td>Invalid XML request: Invalid data XML (challenge)</td>
</tr>
<tr>
<td>3111</td>
<td>Invalid XML request: Invalid data XML (authentication)</td>
</tr>
<tr>
<td>3112</td>
<td>Invalid XML request: Invalid data XML (signature)</td>
</tr>
<tr>
<td>3113</td>
<td>Invalid XML request: Invalid data XML (aggregator)</td>
</tr>
<tr>
<td>3114</td>
<td>Invalid XML request: Error parse XML</td>
</tr>
<tr>
<td>3115</td>
<td>Invalid XML request: XML data is null</td>
</tr>
<tr>
<td>3116</td>
<td>Invalid XML request: Unmatched signature request</td>
</tr>
<tr>
<td>3117</td>
<td>Invalid XML request: Cannot find Aggregator</td>
</tr>
<tr>
<td>3118</td>
<td>User already registered: Duplicate UserID</td>
</tr>
<tr>
<td>3119</td>
<td>Customer account not found: Cannot find customer account</td>
</tr>
<tr>
<td>3120</td>
<td>Not registered UserID</td>
</tr>
<tr>
<td>3121</td>
<td>Daily transaction limit is reached</td>
</tr>
<tr>
<td>3122</td>
<td>Maximum transaction limit is reached</td>
</tr>
<tr>
<td>3123</td>
<td>Transaction payment rejected: Invalid limit configuration</td>
</tr>
<tr>
<td>3124</td>
<td>Transaction payment rejected: Cannot find Merchant ID</td>
</tr>
<tr>
<td>3125</td>
<td>Transaction payment rejected: Inactive merchant</td>
</tr>
<tr>
<td>3126</td>
<td>Transaction payment rejected: Cannot find Bank Commission</td>
</tr>
<tr>
<td>3127</td>
<td>Transaction payment rejected: Cannot find Bank Commission Tearing</td>
</tr>
<tr>
<td>3128</td>
<td>Transaction payment rejected: Cannot find Aggregator Commission</td>
</tr>
<tr>
<td>3129</td>
<td>Transaction payment rejected: Cannot find Aggregator Commission Tearing</td>
</tr>
<tr>
<td>3130</td>
<td>Transaction payment rejected: Duplicate Transaction request</td>
</tr>
<tr>
<td>3131</td>
<td>Reversal rejected: Cannot find original data for reversal</td>
</tr>
<tr>
<td>3132</td>
<td>Reversal rejected: Cannot find merchant account for reversal</td>
</tr>
<tr>
<td>3133</td>
<td>Registration failed: Failed add customer channel</td>
</tr>
<tr>
<td>3134</td>
<td>Unregistered failed: Failed remove customer channel</td>
</tr>
<tr>
<td>3135</td>
<td>Merchant registration failed: Duplicate Merchant</td>
</tr>
<tr>
<td>3201</td>
<td>Error init database</td>
</tr>
<tr>
<td>3202</td>
<td>Error write to database</td>
</tr>
<tr>
<td>4000</td>
<td>No connection to Aggregator</td>
</tr>
<tr>
<td>9000</td>
<td>Other error</td>
</tr>
<tr>
<td>9013</td>
<td>Unable to send request to bank</td>
</tr>
</tbody>
</table>