## Table of Contents

1.0 Introduction ......................................................................................................................................................... 3  
   1.1 Payment Flow ........................................................................................................................................................ 4  
   1.2 Integration ............................................................................................................................................................ 5  

2.0 Credit Card .......................................................................................................................................................... 6  

3.0 DOKU Wallet ....................................................................................................................................................... 14  

4.0 Virtual Account .................................................................................................................................................... 24  
   4.1 Bank Transfer ........................................................................................................................................................ 24  
   4.2 Convenience Store .............................................................................................................................................. 29  

5.0 Internet Banking .................................................................................................................................................... 33  
   5.1 Mandiri Clickpay ................................................................................................................................................. 33  

6.0 Customization ....................................................................................................................................................... 38  

7.0 Appendix .............................................................................................................................................................. 40  
   7.1 Payment Methods ............................................................................................................................................... 40  
   7.2 Payment Request Parameters ............................................................................................................................... 41  
   7.3 DOKU Response Codes ..................................................................................................................................... 41  
      7.3.1 General response codes ............................................................................................................................... 41  
      7.3.2 Credit Card .................................................................................................................................................. 43  
      7.3.3 DOKU Wallet ............................................................................................................................................. 46  
      7.3.4 Virtual Account ......................................................................................................................................... 47  
      7.3.5 Mandiri Clickpay ....................................................................................................................................... 48  
   7.4 Check Payment Status API .................................................................................................................................. 50
1.0 Introduction

This document will act as a tutorial to help you integrate your Android mobile application to DOKU API and start receiving payments from mobile transactions. The DOKU Android SDK enables you to accept payments from customers who make a purchase on mobile devices through your Android application, and currently supports the following payment methods:

- Credit Card
- DOKU Wallet
- Bank Transfer
- Convenience Store
- Mandiri Clickpay

The payment page and data input is native to the merchant’s mobile app, 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.

The DOKU API only provides the fields to be filled in by the customer. All other parts of the payment page, including the logo, colour and ‘Process Payment’ button are customizable to your needs for a completely white label payment flow. Learn how to customize your page in Section 6.0. Once you have installed your SDK, the payment page will look something like this:
1.1 Payment Flow

1. When the customer inputs their card/payment data into the payment form on your app, the embedded SDK will send a token request with the card information to the DOKU server.
2. The DOKU server will generate a **cc_token** and **pairing_code** and send the response to the DOKU SDK within the merchant app. Through this process, none of the credit card data is captured by the merchant, and everything is securely processed by DOKU.
3. The merchant app sends the card data to the merchant server, according to the action parameter in the SDK.
4. The merchant server will send a payment request to DOKU, containing the payment data such as price, customer information and the token.
5. DOKU processes the payment and sends a response in JSON format.
6. The merchant displays the payment result on the app, according to the response sent by DOKU.

*See Appendix (Section 7.0) for response codes.*
1.2 Integration

The following section gives an example of how you can integrate the various payments into your Android mobile application. Once you have confirmed to become a DOKU merchant through our Sales process, you will be contacted by our integration team to proceed to the technical integration stage. 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 instructions are divided into separate sections for each payment method, as the integration process will differ for each method. Credit Card and DOKU Wallet payments use a mobile SDK, while Bank Transfer, Convenience Store and Internet Banking payments use the Merchant-hosted web API.

SDK integration has the following requirements:

- Android studio - IDE
- Build SDK v21
- Minimum Android SDK 11 Gingerbread
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 steps:
1. Embed the DOKU Android SDK into your mobile app
2. Retrieve token
3. Send payment request

1. Embed the DOKU Android SDK into your mobile app

Download the AAR library at http://doku.com/AndroidOCO/SDKOCO-DummyMerchant_V2.zip

a. and paste it to your App’s libs folder. See example:

b. Set the required permissions by pasting the following script into your app’s manifest.xml file

```xml
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
```
c. The SDK uses a type of plugin called Gradle for processing and layouting. Processing includes payment and prepayment requests, while layouting refers to the format of the input form (e.g. how to display the credit card number). Add a reference to Gradle using flatDirs script on the top level of the all projects section (see below).

Add the flatDir script to your 'allprojects':

```gradle
flatDir {
    dirs 'libs'
}
```

d. Create the dependencies for your project by adding the following code into the 'build.gradle' section:

```gradle
compile 'com.doku.sdkocov2:dkocov2-release:1.0@aar'
```

The installation process is complete; you may now commence integration with the payment library.
2. Retrieve Token

a. Initiate the SDK by pasting the following script to your app:

```java
Direct SDK directSDK = new DirectSDK();
```

b. Below is an example code for the parameter which will be sent by the merchant app to the SDK (see the Payment Flow diagram for details). See the Appendix (Section 7.2) for parameter format details.

```java
PaymentItems paymentItems = new PaymentItems();
paymentItems.setTransactionAmount(AppsUtil.generateMoneyFormat("15000"));
paymentItems.setBasket ينبغي ("{"name":"sayur","amount":"10000.00","quantity":1,"subtotal":"10000.00"},{"name":"buah","amount":"10000.00","quantity":1,"subtotal":"10000.00")};
paymentItems.setDataCurrency(360);
paymentItems.setDataBasket("
{"name":"sayur","amount":10000.00,"quantity":1,"subtotal":10000.00"}
{"name":"buah","amount":10000.00,"quantity":1,"subtotal":10000.00"}");
paymentItems.setDataMerchantChain("NA");
paymentItems.setDataSessionID(String.valueOf(AppsUtil.nDigitRandomNo(9)));
paymentItems.setDataTransactionID(invoiceNumber);
paymentItems.setDataMerchantCode("2074");
paymentItems.setDataImei(telephonyManager.getDeviceId());
paymentItems.setMobilePhone("08123123112");
paymentItems.setPublicKey("""); //PublicKey can be obtained from the DOKU Back Office
directSDK.setCart_details(paymentItems);
```

c. Set the payment channel code as follows:

```
15 = for Credit Card
```

directSDK.setPaymentChannel(15);

d. In order for your Android mobile app to receive response from the SDK, you will need to create an interface handler in your activity or fragment. This is an example of how the code will look like:

```java
directSDK.getResponse(new iPaymentCallback() {
    @Override
    public void onSuccess(final String text) {
        try {
            respongetokenSDK = new JSONObject(text);
            if (respongetokenSDK.getString("res_response_code").equalsIgnoreCase("0000")) {
                //do your background AsyncTask service to merchant server handler here
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
    @Override
    public void onError(final String text) {
        //error handling here
    }
    @Override
    public void onException(Exception eSDK) {
        eSDK.printStackTrace();
    }}, getApplicationContext());
```
e. Once you have entered all the conditions above correctly, the merchant app should receive the following response token from the SDK which indicates success.

```json
{  "res_token_id": 53beb0d8da617828d1c6295d822d04b4a12c33ea,
   "res_pairing_code": "07041613200336243915",
   "res_name": "test",
   "res_data_email": "test@mail.com",
   "res_response_msg": "SUCCESS",
   "res_device_id": "867804025368595",
   "res_token_code": "0000",
   "res_amount": "15000.00",
   "res_response_code": "0000",
   "res_payment_channel": "15",
   "res_transaction_id": "1410757974"
}
```

f. After getting your token, there will be instances where your app send it forward to your server, for example for charging payments. To do this, we will use AsyncTask from Android to send it asynchronously. Please see the script below as an example.

```java
private class RequestPayment extends AsyncTask<String, String, JSONObject> {

    @Override
    protected JSONObject doInBackground(String... args) {
        JSONObject defResp = null;
        try {
            List<NameValuePair> data = new ArrayList<NameValuePair>(3);
            data.add(new BasicNameValuePair("data", jsonRespon));
            // Getting JSON from URL
            String conResult = ApiConnection.httpsConnection(MainActivity.this,
                          Constants.URL_CHARGING_DOKU_DAN_CC, data);
            defResp = new JSONObject(conResult);
        } catch (JSONException e) {
            e.printStackTrace();
        } return defResp;
    }
}
```
3. Send Payment Request

   a. Your app will send the data response to your server, which you can pass on to the DOKU server to process the payment.

   ```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'];
   $deviceid = $_POST['deviceid'];
   $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,
       'deviceid' => $deviceid
   );

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

   $dataPayment = array(
       'req_mall_id' => Doku_Initiate::$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_address' => $customer['data_address'],
       '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';
   }
   ```

At this stage, add **token_id** & **pairing_code** to WORDS:

'WORDS = Amount + Mall ID + Shared Key + Invoice + 360 + Token ID + Pairing Code + Device ID'
b. If you have successfully charged payments from a credit card, you will receive responses from DOKU Server as shown below. (see response codes definitions in Appendix)

Example success response for payments made using a credit card:

```json
{
  "res_tid": "13019501",
  "res_trx_code": "ae9e54a6f0788c41a8c4d466f83fc7c268fc0725",
  "res_currency": "IDR",
  "res_approval_code": "900059",
  "res_eci": "",
  "res_chain_mall_id": "",
  "res_card_number": "4***********1111",
  "res_amount": "15000.00",
  "res_message": "PAYMENT APPROVED",
  "res_issuer_bank": "JPMORGAN CHASE BANK",
  "res_mall_id": "2",
  "res_liability": "MERCHANT",
  "res_mid": "000100013000195",
  "res_result": "SUCCESS",
  "res_payment_date": "20160411180526",
  "res_three_d_secure_status": "FALSE",
  "res_bank": "BNI",
  "res_invoice_number": "1210090970",
  "res_response_code": "0000",
  "res_session_id": "7a573175e3762c01145be0ae155fcacedce030e3",
  "res_payment_channel": "15"
}
```
2.1 Advanced Features

2.1.1 2-Click Payment

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 ‘CustomerID’ to your script under the payment data. The ‘CustomerID’ 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.

```java
PaymentItems paymentItems = new PaymentItems();
paymentItems.setDataAmount(AppsUtil.generateMoneyFormat("15000"));
paymentItems.setDataBasket('[{"name":"sayur","amount":"10000.00","quantity":"1","subtotal":"10000.00"},
{"name":"buah","amount":"10000.00","quantity":"1","subtotal":"10000.00"}]');
paymentItems.setDataCurrency("360");
paymentItems.setDataWords(AppsUtil.SHA1(AppsUtil.generateMoneyFormat("15000") + "2074" + 
"eA6I1lJJS19J" + invoiceNumber + 360 +
telephonyManager.getDeviceId()));
paymentItems.setDataMerchantChain("NA");
paymentItems.setDataSessionID(String.valueOf(AppsUtil.nDigitRandomNo(9)));
paymentItems.setDataTransactionID(invoiceNumber);
paymentItems.setDataMerchantCode("2074");
paymentItems.setDataImei(telphonyManager.getDeviceId());
paymentItems.setMobilePhone("08123123112");
paymentItems.isProduction(false); //set ‘true’ for production and ‘false’ for development
paymentItems.setPublicKey(""); //PublicKey can be obtained from the DOKU Back Office
paymentItems.setCustomerID("12124");
directSDK.setCart_details(paymentItems);
```
The script 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 'CustomerID' 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 credit card data” 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_word": "7553a51a091175a2462eb9150c7135f4a8d58ff161db022ca42e0ef65666ebf0",
  "res_response_msg": "SUCCESS",
  "res_mch": "4***********1111",
  "res_bk": "JPMORGAN CHASE BANK",
  "res_payment_channel": "15",
  "res_bundle_token": "{"res_token_payment": "0bea1c1c653dbc8e1e6c24155c629fe237325a06",
    "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 script. Add the extra parameter 'TokenPayment' as seen below, by using the token value that was obtained during the first payment:

```java
PaymentItems paymentItems = new PaymentItems();
paymentItems.setDataAmount(AppsUtil.generateMoneyFormat("15000"));
paymentItems.setDataBasket('"\"name\":\"sayur\",\"amount\":\"10000.00\",\"quantity\":\"1\",\"subtotal\":\"10000.00\"\"}],["\"name\":\"buah\",\"amount\":\"10000.00\",\"quantity\":\"1\",\"subtotal\":\"10000.00\"]");
paymentItems.setDataCurrency("360");
paymentItems.setDataWords(AppsUtil.SHA1(AppsUtil.generateMoneyFormat("15000") + "2074" + "ea61iJJ9J" + invoiceNumber + 360 + telephonyManager.getDeviceId());
paymentItems.setDataMerchantChain("NA");
paymentItems.setDataSessionID(String.valueOf(AppsUtil.nDigitRandomNo(9)));
paymentItems.setDataTransactionID(invoiceNumber);
paymentItems.setDataMerchantCode("2074");
paymentItems.setDatamobilePhone(telephonyManager.getDeviceId());
paymentItems.setMobilePhone("08123131212");
paymentItems.setTokenPayment("0bea1c1c653dbc8e1e6c24155c6299e237325a06");
directSDK.setCart_details(paymentItems);
```

The above script will generate the following payment form:

![Payment Form Screenshot]

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

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 mobile app. 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. Change method of payment in the payment request form.

1. To initialize 1-click payment, follow the same steps as general credit card processing, but add the additional parameter ‘CustomerID’ to your script under the payment data. The ‘CustomerID’ 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.

```java
PaymentItems paymentItems = new PaymentItems();
paymentItems.setServerScript(AppsUtil.getServerScript("15000"));
paymentItems.setBasket("["name":"sayur","amount":"10000.00","quantity":1,"subtotal":"10000.00"],
{name":"buah","amount":"10000.00","quantity":1,"subtotal":"10000.00"}");
paymentItems.setCurrency("360");
paymentItems.setSessionID(String.valueOf(AppsUtil.nDigitRandomNo(9)));
paymentItems.setTransactionID(invoiceNumber);
paymentItems.setMerchantCode("2074");
paymentItems.setImei(telephonyManager.getDeviceId());
paymentItems.setCustomerID("12124");
directSDK.setCart_details(paymentItems);
```
The script with the additional parameter will generate the following payment form, which enables the customer to save their credit card:

![Payment Form](image)

2. When the customer has filled in their card details and clicked “Process Payment”, the data is sent to the DOKU server. Because the ‘CustomerID’ 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 credit card data”, 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": "7553a51a091775a2462eb9150c7135f4a8d58ff161db022ca42e0ef65666ebf0",
    "res_response_msg": "SUCCESS",
    "res_mcn": "4***********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": "0bea1c1c653dbc8e6e6624155c629fe237325a06",
                            "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 1-Click service.
3. For future payments, the request will work server-to-server without involving the SDK. 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:

```php
<?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' => '1234567899'
);

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

$dataPayment = array(
    'req_mall_id' => Doku_Initiate::$mallId,
    'req_chain_merchant' => 'NA',
    'req_amount' => $params['amount'],
    'req_words' => $words,
    'req_purchase_amount' => $params['amount'],
    'req_trans_id_merchant' => $params['invoice'],
    '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_email' => $customer['data_email'],
    'req_basket' => $basket,
    'req_address' => $customer['data_address'],
    'req_token_payment' => '0bea1c1c653dbc8e1e6c24155c629fe237325a06',
    'req_customer_id' => '12124'
);

$response = Doku_Api::doDirectPayment($dataPayment);
if($response->res_response_code == '0000'){
    echo 'PAYMENT SUCCESS -- ';}
else{
    echo 'PAYMENT FAILED -- ';}
?>
At this stage, add `token_id` & `pairing_code` to WORDS:

`WORDS = Amount + Mall ID + Shared Key + Invoice + 360 + Token ID + Pairing Code + Device ID`

### 3.0 DOKU Wallet

DOKU Wallet integration comprises 3 steps:

1. Embed the DOKU Android SDK into your mobile app
2. Retrieve token
3. Send payment request

1. Embed the DOKU Android SDK into your mobile app

   a. Download the AAR library at [http://doku.com/AndroidOCO/SDKOCO-DummyMerchant_V2.zip](http://doku.com/AndroidOCO/SDKOCO-DummyMerchant_V2.zip) and paste it to your App’s `libs` folder. See example:

   ![Example of embedding the DOKU Android SDK](image)

   b. Set the required permissions by pasting the following script into your app’s `manifest.xml` file

   ```xml
   <uses-permission android:name="android.permissionINTERNET" />  
   <uses-permission android:name="android.permissionACCESS_NETWORK_STATE" />  
   <uses-permission android:name="android.permissionACCESS_WIFI_STATE" />  
   <uses-permission android:name="android.permissionREAD_PHONE_STATE" />  
   ```

   c. The SDK uses a type of plugin called Gradle for processing and layouting. Processing includes payment and prepayment requests, while layouting refers to the format of the input form (e.g. how
to display the credit card number). Add a reference to Gradle using flatDirs script on the top level of the all projects section (see below).

Add the flatDir script to your `allprojects`:

```groovy
flatDir {
  dirs 'libs'
}
```

d. Create the dependencies for your project by adding the following code into the `build.gradle` section:

```groovy
compile 'com.doku.sdkocov2:/sdkocov2-release:1.0@aar'
```

The installation process is complete; you may now commence integration with the payment library.

2. Retrieve Token
a. Initiate the SDK by pasting the following script to your app:

```
Direct SDK directSDK = new DirectSDK();
```

b. Below is an example code for the parameter which will be sent by the merchant app to the SDK (see the Payment Flow diagram for details). See the appendix for parameter format details.

```
PaymentItems paymentItems = new PaymentItems();
paymentItems.setdataAmount(AppsUtil.generateMoneyFormat("15000"));
paymentItems.setBasket("[{"name":"sayur","amount":"10000.00","quantity":"1","subtotal":"10000.00"},{"name":"buah","amount":"10000.00","quantity":"1","subtotal":"10000.00"}]" );
paymentItems.setCurrency("360");
paymentItems.setSessionID(String.valueOf(AppsUtil.randomNo(9)));
paymentItems.setDataMerchantCode("2074");
paymentItems.setDataImei(telephonyManager.getDeviceId());
directSDK.setCart_details(paymentItems);

```
c. Set the payment channel code as follows:

```
04 = for DOKU Wallet
```

directSDK.setPaymentChannel(04);

d. In order for your Android mobile app to receive response from the SDK, you will need to create an interface handler in your activity or fragment. This is an example of how the code will look like:

```
directSDK.getResponse(new iPaymentCallback() {
    @Override
    public void onSuccess(final String text) {
        try {
            respongetokenSDK = new JSONObject(text);
            if (respongetokenSDK.getString("res_response_code").equalsIgnoreCase("0000")) {
              //Do your background AsyncTask service to merchant server handler here
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
    @Override
    public void onError(final String text) {
        //error handling here
    }
    @Override
    public void onException(Exception eSDK) {
        eSDK.printStackTrace();
    }, getApplicationContext());
```
Once you have entered all the conditions above correctly, the merchant app should receive the following response token from the SDK which indicates success.

```json
e. After getting your token, there will be instances where your app send it forward to your server, for example for charging payments. To do this, we will use AsyncTask from Android to send it asynchronously. Please see the script below as an example.

```java
private class RequestPayment extends AsyncTask<String, String, JSONObject> {

  @Override
  protected JSONObject doInBackground(String... args) {
    JSONObject defResp = null;

    try {
      List<NameValuePair> data = new ArrayList<NameValuePair>(3);
      data.add(new BasicNameValuePair("data", jsonRespon));

      // Getting JSON from URL
      String conResult = ApiConnection.httpsConnection(MainActivity.this,
                          Constants.URL_CHARGING_DOKU_DAN_CC, data);

      defResp = new JSONObject(conResult);
    } catch (JSONException e) {
      e.printStackTrace();
    }

    return defResp;
  }
}
```
3. Send Payment Request
   a. Your app will send the data response to your server, which you can pass on to the DOKU server to process the payment.

```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']
$deviceid = $_POST['deviceid']
$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,
    'deviceid' => $deviceid
);
$words = Doku_Library::doCreateWords($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'
);
$dataPayment = array(
    'req_mall_id' => Doku_Initiate::$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' => 04,
    'req_basket' => $basket,
    'req_address' => $customer['data_address'],
    'req_email' => $customer['data_email'],
    'req_token_id' => $token
);
$result = Doku_Api::doPayment($dataPayment);
if($result->res_response_code == '0000'){
    echo 'SUCCESS' ;
    //success
} else{
    echo 'FAILED';
    //failed
}
```

At this stage, add **token_id & pairing_code** to **WORDS:**

```
'WORDS = Amount + Mall ID + Shared Key + Invoice + 360 + Token ID + Pairing Code + Device ID'
```
b. If you have successfully charged payments from a DOKU Wallet, you will receive responses from DOKU Server as shown below. See response code definitions in Appendix (Section 7.3).

Example success response for payments made using DOKU Wallet:

```
{
  "res_dp_mall_id": 47,
  "res_tracking_id": 30267,
  "res_response_msg": "Berhasil",
  "res_approval_code": "253052",
  "res_trans_id_merchant": "1438202478",
  "res_payment_channel_code": "01",
  "res_status": "Success",
  "res_bank": "CASH",
  "res_response_code": "0000",
  "res_payment_channel": "04"
}
```
4.0 Virtual Account

DOKU Virtual Account aggregates the funds using 3 different entities – Bank Permata, Bank Sinarmas and Alfa Group. 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 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:

Permata: 89650
Sinarmas: 88900
Alfa: 88888

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 app
3. Receive Payment Notification
4. Notify DOKU server that Payment Notification has been received

To get started on your integration, follow these steps one by one by pasting these scripts into your app.

1. Get the device ID from your customer by pasting the script into your app:

```java
TelephonyManager telephonyManager = (TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE);
telephonyManager.getDeviceId();
```
Create a "Get Payment Code" button, and display it in your app like this:
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' => '60'
);

$response = Doku_Api::doGeneratePaycode($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:

```json
{
"res_pay_code": "627000000003",
"res_pairing_code": "290316110837531987",
"res_response_msg": "SUCCESS",
"res_response_code": "0000"
}
```
2. Display the result in your app however you wish. If you choose to display the 11 digits only, letting the customer choose their payment method (remember to add a “How To” in the instructions), the result can be display like this:

Alternatively, 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=05
AMOUNT=1000.00
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 app
3. Receive Payment Notification
4. Notify DOKU server that Payment Notification has been received

To get started on your integration, follow these steps one by one by pasting these scripts into your app.

1. Get the device ID from your customer by pasting the script into your app:

```java
TelephonyManager telephonyManager = (TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE);
telephonyManager.getDeviceId();
```

Create a “Get Payment Code” button, and display it in your app like this:
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' => $_POST['mall_id'],
    'req_chain_merchant' => $_POST['chain_merchant'],
    'req_amount' => $params['amount'],
    'req_words' => $words,
    'req_purchase_amount' => $params['amount'],
    'req_trans_id_merchant' => $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;'
);

$response = Doku_Api::doGeneratePaycode($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:

```json
{
    "res_pay_code": "627000000003",
    "res_pairing_code": "290316110837531987",
    "res_response_msg": "SUCCESS",
    "res_response_code": "0000"
}
```
2. Display the result in your app however you wish. If you choose to display the 11 digits only, letting the customer choose their payment method (remember to add a “How To” in the instructions), the result can be displayed like this:

Alternatively, you can display all three options like this:
5. 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=05
AMOUNT=1000.00
PAYMENTCODE=00100000029
WORDS=01d9b362d3c1b80ff9196c6a565c49e5d9b03b8a
RESULTMSG=SUCCESS
TRANSIDMERCHANT=2A912172
BANK=PERMATA
STATUSTYPE=P
APPROVALCODE=068992
RESPONSECODE=0000
SESSIONID=7b6647973dd13211a7fcf42eba79acea68aa69al
```

6. 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 your 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. Create payment form and send parameters to your server
2. Insert the challenge code formula
3. Send payment request

To get started on your integration, follow these steps one by one by pasting these scripts into your app.

1. Create the payment form with a field for debit card number input, like this:
Get the device ID from your customer by pasting the script into your app:

```java
TelephonyManager telephonyManager = (TelephonyManager)
    getSystemService(Context.TELEPHONY_SERVICE);
telephonyManager.getDeviceId()
```

2. Once your customer has input their Mandiri Debit Card number, they will receive the first challenge code to enter into their Mandiri Token Device.

The challenge code formulas have different formats according to the table below:

<table>
<thead>
<tr>
<th>Challenge Code 1</th>
<th>Last 10 digits from debit card</th>
</tr>
</thead>
<tbody>
<tr>
<td>Challenge Code 2</td>
<td>Amount for charging</td>
</tr>
<tr>
<td>Challenge Code 3</td>
<td>8 digit random number</td>
</tr>
</tbody>
</table>

Obtain challenge code 1 by copying the following script:

```java
yourEditTextHere.addTextChangedListener(new TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {
    
    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {
    }

    @Override
    public void afterTextChanged(Editable s) {
        // Remove spacing char
        if (s.length() > 0 && (s.length() % 5) == 0) {
            final char c = s.charAt(s.length() - 1);
            if (space == c) {
                s.delete(s.length() - 1, s.length());
            }
        }

        // Insert char where needed.
        if (s.length() > 0 && (s.length() % 5) == 0) {
            char c = s.charAt(s.length() - 1);
            if (Character.isDigit(c) && TextUtils.split(s.toString(),
                String.valueOf(space)).length <= 3) {
                s.insert(s.length() - 1, String.valueOf(space));
            }
        }

        if (s.length() == 19) {
            mandiriCardNumber = debitCard.getText().toString().replace("-", "");
            challengeValue1.setText(mandiriCardNumber.substring(6, 16));
        }
    }
});
```
Generate random integers for Challenge Code 3 with the following script:

```java
public static int nDigitRandomNo(int digits) {
    int max = (int) Math.pow(10, (digits)) - 1; //for digits = 7, max will be 9999999
    int min = (int) Math.pow(10, digits - 1); //for digits = 7, min will be 1000000
    int range = max - min; //This is 8999999
    Random r = new Random();
    int x = r.nextInt(range); //This will generate random integers in range 0 - 8999999
    int nDigitRandomNo = x + min; //Our random number will be any random number x + min
    return nDigitRandomNo;
}
```

To execute the above function, use this method:

```java
String randomNumber = String.valueOf(nDigitRandomNo(8));
```
Once the customer has completed the Challenge Codes and clicked the “Process Payment” button, your app will send the data to your server. Then your server will pass through the data to the DOKU server, 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>';

$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_request_date_time' => date('YmdHis'),
    'req_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 -- '; 
}
A success response from Mandiri Clickpay will look like this:

```json
{
    "res_response_msg": "SUCCESS",
    "res_transaction_code": "cb37e335793405d0741979785b1fe65814dab182",
    "res_mcn": "4***********1111",
    "res_trans_id_merchant": "invoice_1460447091",
    "res_payment_date": "20160412144451",
    "res_bank": "MANDIRI CLICK PAY",
    "res_amount": "15000.00",
    "res_message": "PAYMENT APPROVED",
    "res_response_code": "0000",
    "res_session_id": "11c5075b3602138814a392fda3eca0ac9113dd76"
}
```

After this is done, you may display the result on your app for the customer to see.
6.0 Customization

In order to customize your payment page, you need to set the layout settings every time you call the library. If you do not customize your page, the layout will have the default theme set by DOKU, and will look like this:

Here is a sample script for how to set up your custom layout:

```java
LayoutItems layoutItems = new LayoutItems();
layoutItems.setFontPath("fonts/dinbold.ttf");
layoutItems.setToolbarColor("#289c64");
layoutItems.setToolbarTextColor("#FFFFFF");
layoutItems.setFontColor("#121212");
layoutItems.setBackgroundColor("#eaeaea");
layoutItems.setLabelTextColor("#9a9a9a");
layoutItems.setButtonBackground(getResources().getDrawable(R.drawable.button_orange));
layoutItems.setButtonTextColor("#FFFFFF");

layoutItems.setButtonBackground(getResources().getDrawable(R.drawable.button_orange));
layoutItems.setButtonTextColor("#FFFFFF");

directSDK.setLayout(layoutItems);
```
Then your code will look like this:

```java
DirectSDK directSDK = new DirectSDK();
PaymentItems paymentItems = new PaymentItems();
paymentItems.setDataAmount(AppsUtil.generateMoneyFormat("15000"));
paymentItems.setDataBasket("[{"name":"sayur","amount":"10000.00","quantity":"1","subtotal":"10000.00"},{"name":"buah","amount":"10000.00","quantity":"1","subtotal":"10000.00"}]";
paymentItems.setDataCurrency("360");
paymentItems.setDataWords(AppsUtil.SHA1(AppsUtil.generateMoneyFormat("15000") + "2074" + "eae61117jsj19j" + invoiceNumber + 360 + Token ID + Pairing Code + telephonyManager.getDeviceId()));
paymentItems.setDataSessionID(String.valueOf(AppsUtil.nDigitRandomNo(9)));
paymentItems.setDataTransactionID(invoiceNumber);
paymentItems.setDataMerchantCode("2074");
paymentItems.setDataImei(telephonyManager.getDeviceId());
paymentItems.setMobilePhone("08123123112");
directSDK.setCart_details(cardDetails);

LayoutItems layoutItems = new LayoutItems();
layoutItems.setFontPath("fonts/dinbold.ttf");
layoutItems.setToolbarColor("#289c64");
layoutItems.setToolbarTextColor("#FFFFFF");
layoutItems.setFontColor("#121212");
layoutItems.setBackgroundColor("#eaeaea");
layoutItems.setLabelTextColor("#9a9a9a");
layoutItems.setButtonBackground(getResources().getDrawable(R.drawable.button_orange));
layoutItems.setButtonTextColor("#FFFFFF");
directSDK.setLayout(layoutItems);
directSDK.setPaymentChannel(menuPaymentChannel);
directSDK.getResponse(new iPaymentCallback() {
    @Override
    public void onSuccess(final String text) {
        try {
            respongetTokenSDK = new JSONObject(text);
            if (respongetTokenSDK.getString("res_response_code").equalsIgnoreCase("0000")) {
                //do your background service to merchant server handler here
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
    @Override
    public void onError(final String text) {
        //error handling here
    }
    @Override
    public void onException(Exception eSDK) {
        eSDK.printStackTrace();
    }
});
```
## 7.0 Appendix

### 7.1 Payment Methods

<table>
<thead>
<tr>
<th>Payment Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| Credit Card        | • Visa and Mastercard for Overseas Partner. JCB upon request  
                      • Direct API available  
                      • Features (acquirer dependant): 3D and non 3D Secure, BIN filtering, 1-Click Payment, 2-Click Payment |
| Internet Banking   | • Available: Mandiri Clickpay, BCA Klikpay, BRI e-Pay, Danamon, Muamalat, Permata  
                      • Each bank has different authentication process through OTP or token  
                      • Direct API only available for Mandiri Clickpay. The rest is re-direct only |
| DOKU Wallet        | • E-wallet product issued by DOKU  
                      • Source of fund: cash balance or linked credit card  
                      • Max. transaction value is Rp1,000,000 for non-KYC and Rp5,000,000 for KYC users  
                      • Authenticate with email, password and static PIN that is pre-set by the user  
                      • Direct API available |
| Convenience Store  | • Accessible in almost 10,000 Alfa group stores (Alfa Express, Alfa Midi, Alfa Mart, Lawson and DAN+DAN)  
                      • Generate 16 digit payment code at checkout, user goes to nearest store and makes payment over the counter with cash or non-cash  
                      • Max. transaction value of Rp2,000,000  
                      • Merchant can set payment code expiry time for every transaction  
                      • Direct API available |
| Bank Transfer      | • Virtual account housed in Bank Permata but payable from any bank that is connected to ATM Bersama, Prima or Alto networks (over 120 banks in Indonesia)  
                      • 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  
                      • Merchant can set payment code expiry time for every transaction  
                      • Direct API available |
### 7.2 Payment Request Parameters

<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Merchant Code</td>
<td>String</td>
<td>Merchant Code from DOKU</td>
</tr>
<tr>
<td>2.</td>
<td>Words</td>
<td>String</td>
<td>Data encrypted from Merchant (amount + mall_id + shared_key + transaction_id + currency + token ID + pairing code + imei device)</td>
</tr>
<tr>
<td>3.</td>
<td>Transaction ID</td>
<td>String</td>
<td>Invoice number</td>
</tr>
<tr>
<td>4.</td>
<td>Amount</td>
<td>String</td>
<td>Amount to purchase</td>
</tr>
<tr>
<td>5.</td>
<td>Currency</td>
<td>String</td>
<td>Currency that are used</td>
</tr>
<tr>
<td>6.</td>
<td>Chain Merchant</td>
<td>String</td>
<td>Chain merchant number</td>
</tr>
<tr>
<td>7.</td>
<td>Basket</td>
<td>String</td>
<td>Order items</td>
</tr>
<tr>
<td>8.</td>
<td>Session ID</td>
<td>String</td>
<td>Unique key from merchant</td>
</tr>
<tr>
<td>9.</td>
<td>Imei Device</td>
<td>String</td>
<td>Device Imei id from merchant</td>
</tr>
<tr>
<td>10.</td>
<td>Mobile Phone</td>
<td>String</td>
<td>Customer mobile phone number</td>
</tr>
<tr>
<td>11.</td>
<td>Merchant Status</td>
<td>String</td>
<td>Status merchant ‘TRUE’: production, ‘FALSE’: staging’</td>
</tr>
<tr>
<td>12.</td>
<td>Public Key</td>
<td>String</td>
<td>Public key from DOKU</td>
</tr>
</tbody>
</table>

#### Extra Parameters for Credit Card Tokenization

<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.</td>
<td>Customer ID</td>
<td>String</td>
<td>Customer ID set from merchant</td>
</tr>
<tr>
<td>14.</td>
<td>Token Payment</td>
<td>String</td>
<td>Token from DOKU for save CC</td>
</tr>
</tbody>
</table>

### 7.3 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.

#### 7.3.1 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>Undefined error</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>
</tbody>
</table>

www.doku.com
PT Nusa Satu Inti Artha
Plaza Asia Office Park Unit 3
Jl. Jenderal Sudirman Kav. 59
Jakarta 12190 Indonesia
<table>
<thead>
<tr>
<th>Code</th>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<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>
<tr>
<td>5536</td>
<td>Time Out</td>
</tr>
<tr>
<td>5537</td>
<td>Invalid Currency</td>
</tr>
<tr>
<td>5538</td>
<td>Invalid Purchase Currency</td>
</tr>
<tr>
<td>5539</td>
<td>3D Secure Enrolment check failed</td>
</tr>
<tr>
<td>5540</td>
<td>3D Secure Authentication failed</td>
</tr>
<tr>
<td>5541</td>
<td>Form Type is not valid</td>
</tr>
<tr>
<td>5542</td>
<td>Duplicate Transaction ID</td>
</tr>
<tr>
<td>5543</td>
<td>Please check 3D Secure result</td>
</tr>
<tr>
<td>5544</td>
<td>Failed to delete token</td>
</tr>
<tr>
<td>5545</td>
<td>Failed to Void</td>
</tr>
<tr>
<td>5547</td>
<td>BIN are not allowed in promo</td>
</tr>
<tr>
<td>5548</td>
<td>Invalid Parameter</td>
</tr>
<tr>
<td>5553</td>
<td>Failed to tokenize</td>
</tr>
<tr>
<td>5557</td>
<td>Failed to process transaction</td>
</tr>
<tr>
<td>5558</td>
<td>Invalid Order</td>
</tr>
<tr>
<td>5559</td>
<td>Failed to inquiry list of fund</td>
</tr>
<tr>
<td>5560</td>
<td>Invalid Purchase Currency</td>
</tr>
<tr>
<td>5561</td>
<td>3D Secure Authentication failed</td>
</tr>
<tr>
<td>5562</td>
<td>Form Type is not valid</td>
</tr>
<tr>
<td>5563</td>
<td>Duplicate Transaction ID</td>
</tr>
<tr>
<td>5564</td>
<td>Please check 3D Secure result</td>
</tr>
<tr>
<td>5565</td>
<td>Failed to delete token</td>
</tr>
<tr>
<td>5566</td>
<td>Failed to Void</td>
</tr>
<tr>
<td>5567</td>
<td>BIN are not allowed in promo</td>
</tr>
<tr>
<td>5568</td>
<td>Invalid Parameter</td>
</tr>
<tr>
<td>5569</td>
<td>Failed to tokenize</td>
</tr>
</tbody>
</table>
### 7.3.2 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></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>Honor with ID</td>
<td></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>0013</td>
<td>Invalid amount (currency conversion field overflow, Visa Cash - Invalid load mount)</td>
<td>Invalid Amount</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>0014</td>
<td>Invalid account number (no such number)</td>
<td>Invalid Card Number</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>0015</td>
<td>No such issuer</td>
<td>Invalid issuer</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>0019</td>
<td>Re-enter transaction</td>
<td></td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>0021</td>
<td>No Action taken (unable to back out prior transaction)</td>
<td></td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>0025</td>
<td>Unable to locate record in file, or account number is missing from inquiry</td>
<td></td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>0028</td>
<td>File is temporarily unavailable</td>
<td></td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>0030</td>
<td>Format error</td>
<td>VISA/MASTER</td>
<td></td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Reason</td>
<td>Brand</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------------------------------------</td>
<td>-------------------</td>
<td></td>
</tr>
<tr>
<td>0041</td>
<td>Pickup card (lost card)</td>
<td>Lost Card</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0043</td>
<td>Pickup card (stolen card)</td>
<td>Stolen Card</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0051</td>
<td>Insufficient funds</td>
<td>Insufficient Funds/Over Credit limit</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0052</td>
<td>No checking account</td>
<td></td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0053</td>
<td>non savings account</td>
<td></td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0054</td>
<td>Expired card</td>
<td>Expired Card</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0055</td>
<td>Incorrect PIN (Visa cash - invalid or missing SI signature)</td>
<td>Invalid PIN</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0057</td>
<td>Transaction not permitted to cardholder [Visa cash - incorrect routing, not a</td>
<td>Transaction not permitted to</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0058</td>
<td>Transaction not allowed at terminal</td>
<td>Transaction not permitted to</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0061</td>
<td>Activity amount limit exceeded</td>
<td>Exceeds withdrawal amount limit</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0062</td>
<td>Restricted card (for example in country exclusion table)</td>
<td>Restricted Card</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0063</td>
<td>Security violation</td>
<td>Security Violation</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0065</td>
<td>Activity count limit exceeded</td>
<td>Exceeds withdrawal count limit</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0075</td>
<td>Allowable number of PIN-entry tries exceeded</td>
<td>Allowable number of PIN tries exceeded</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0076</td>
<td>Unable to locate previous message (no match on Retrieval Reference number)</td>
<td>Invalid/nonexistent &quot;To Account&quot; specified</td>
<td>VISA/MASTERC</td>
<td></td>
</tr>
<tr>
<td>0077</td>
<td>Previous message located for a repeat or reversal, but repeat or reversal</td>
<td>Invalid/nonexistent &quot;From Account&quot; specified</td>
<td>VISA/MASTERC</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/MASTERC</td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Contact DOKU or ACQUIRING BANK</td>
<td></td>
</tr>
</tbody>
</table>

- Should consider blocking the card temporarily or Block login ID.
- Should consider blocking the card temporarily or Block login ID.
- Tell Customer to contact the Bank Issuer of the card used.
- Tell Customer to contact the Bank Issuer of the card used.
- Tell Customer to contact the Bank Issuer of the card used.
- Tell Customer to contact the Bank Issuer of the card used.
- Tell Customer to contact the Bank Issuer of the card used.
- Tell Customer to contact the Bank Issuer of the card used.
- Tell Customer to contact the Bank Issuer of the card used.
- Tell Customer to contact the Bank Issuer of the card used.
- Tell Customer to contact the Bank Issuer of the card used.
- Contact DOKU or ACQUIRING BANK.
- Contact DOKU or ACQUIRING BANK.
- Contact DOKU or ACQUIRING BANK.
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Card</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<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>
</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>
</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>
</tr>
<tr>
<td>0084</td>
<td>Authorization Life Cycle</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</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>
</tr>
<tr>
<td>0091</td>
<td>Issuer unavailable or switch inoperative (STIP not applicable or available for this transaction)</td>
<td>Authorization System or Issuer system inoperative</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>0092</td>
<td>Destination cannot be found for routing</td>
<td>Unable to route transaction</td>
<td>Contact DOKU or ACQUIRING BANK</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>
</tr>
<tr>
<td>0094</td>
<td>Duplicate transmission detected</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>0096</td>
<td>System malfunction / System malfunction or certain field error conditions</td>
<td>System Error</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>00NO</td>
<td>Force STIP</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>00N3</td>
<td>Cash service not available</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>00N4</td>
<td>Cash request exceeds issuer limit</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>00N7</td>
<td>Decline for CW2 failure</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>00P2</td>
<td>Invalid biller information</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>00P5</td>
<td>PIN Change/Unblock request declined</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>00P6</td>
<td>Unsafe PIN</td>
<td>VISA/MASTER</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>00TO</td>
<td>Timeout / Transaction’s response exceed time limit</td>
<td>Timeout / Transaction’s response exceed time limit</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
<tr>
<td>00UE</td>
<td>Unknown Exception / PosServer not responding</td>
<td>Unknown Exception / PosServer not responding</td>
<td>Contact DOKU or ACQUIRING BANK</td>
</tr>
</tbody>
</table>
### 7.3.3 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>
7.3.4 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>
7.3.5 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>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 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>Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------------------------------------------</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>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>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>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>Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>------------------------------------------------------------------------</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>

### 7.4 Check Payment Status API
DOKU provides an API for merchants to check the status of a specific transaction. The implementation of this API is optional and can be used if the merchant wants to re-confirm the status of a particular transaction. This API can be accessed by the HTTP POST Method.

HTTP action URL: https://pay.doku.com/Suite/CheckStatus

To use this API, you should send the below parameters to the above URL:

<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Type</th>
<th>Length</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MALLID</td>
<td>N</td>
<td></td>
<td>Given by DOKU</td>
</tr>
<tr>
<td>2</td>
<td>CHAINMERCHANT</td>
<td>N</td>
<td></td>
<td>Given by DOKU</td>
</tr>
<tr>
<td>3</td>
<td>TRANSIDMERCHANT</td>
<td>AN</td>
<td>…30</td>
<td>Transaction ID from merchant</td>
</tr>
<tr>
<td>4</td>
<td>SESSIONID</td>
<td>AN</td>
<td>…48</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>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: MALLID+&lt;shared key&gt;+TRANSIDMERCHANT. For transaction with currency other than 360 (IDR), use: MALLID+&lt;shared key&gt;+TRANSIDMERCHANT+CURRENCY</td>
</tr>
</tbody>
</table>

Response Status

Once the API is executed, DOKU will respond with the payment status in XML format as per below. You can check the Response Code from the table above in this Appendix.

```xml
<?xml version="1.0"?>
<PAYMENT_STATUS>
  <AMOUNT></AMOUNT>
  <TRANSIDMERCHANT></TRANSIDMERCHANT>
  <RESPONSECODE></RESPONSECODE>
  <APPROVALCODE></APPROVALCODE>
  <RESULTMSG></RESULTMSG>
  <PAYMENTCHANNEL></PAYMENTCHANNEL>
  <SESSIONID></SESSIONID>
  <BANK></BANK>
  <MCN></MCN>
  <PAYMENTDATETIME></PAYMENTDATETIME>
  <VERIFYID></VERIFYID>
  <VERIFYSCORE></VERIFYSCORE>
  <VERIFYSTATUS></VERIFYSTATUS>
  <CURRENCY></CURRENCY>
  <PURCHASECURRENCY></PURCHASECURRENCY>
  <BRAND></BRAND>
  <CHNAME></CHNAME>
  <THREEDSECURESTATUS></THREEDSECURESTATUS>
  <LIABILITY></LIABILITY>
  <EDUSTATUS></EDUSTATUS>
</PAYMENT_STATUS>
7.5 Information for ProGuard users

Please add the following additional parameters in your proguard-rules.pro:

- dontwarn org.bouncycastle.**
- dontwarn com.doku.sdkocov2.**