



ipay88 Indonesia

Technical Documentation
Version 1.7.4

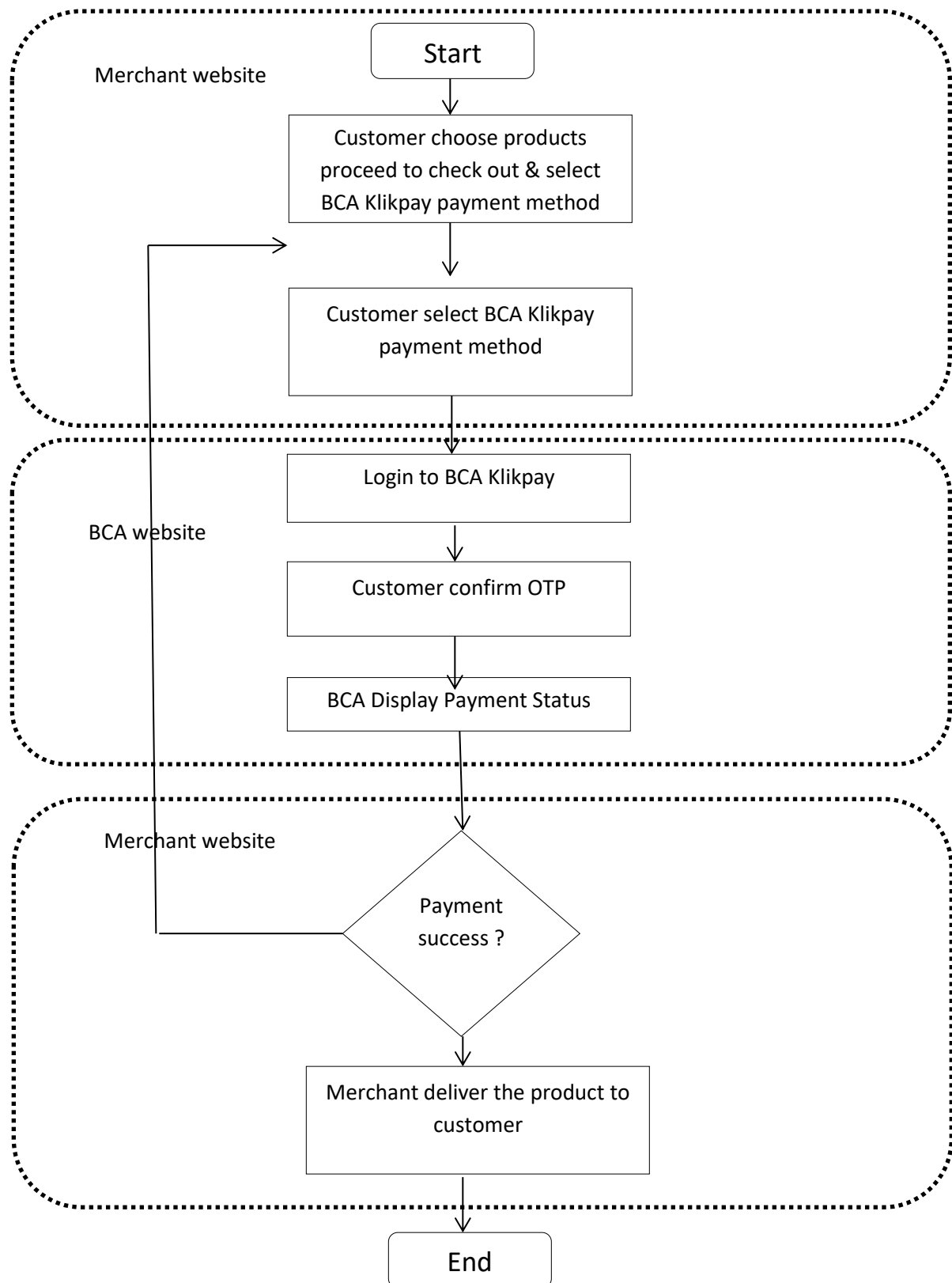
BCA KlikPay

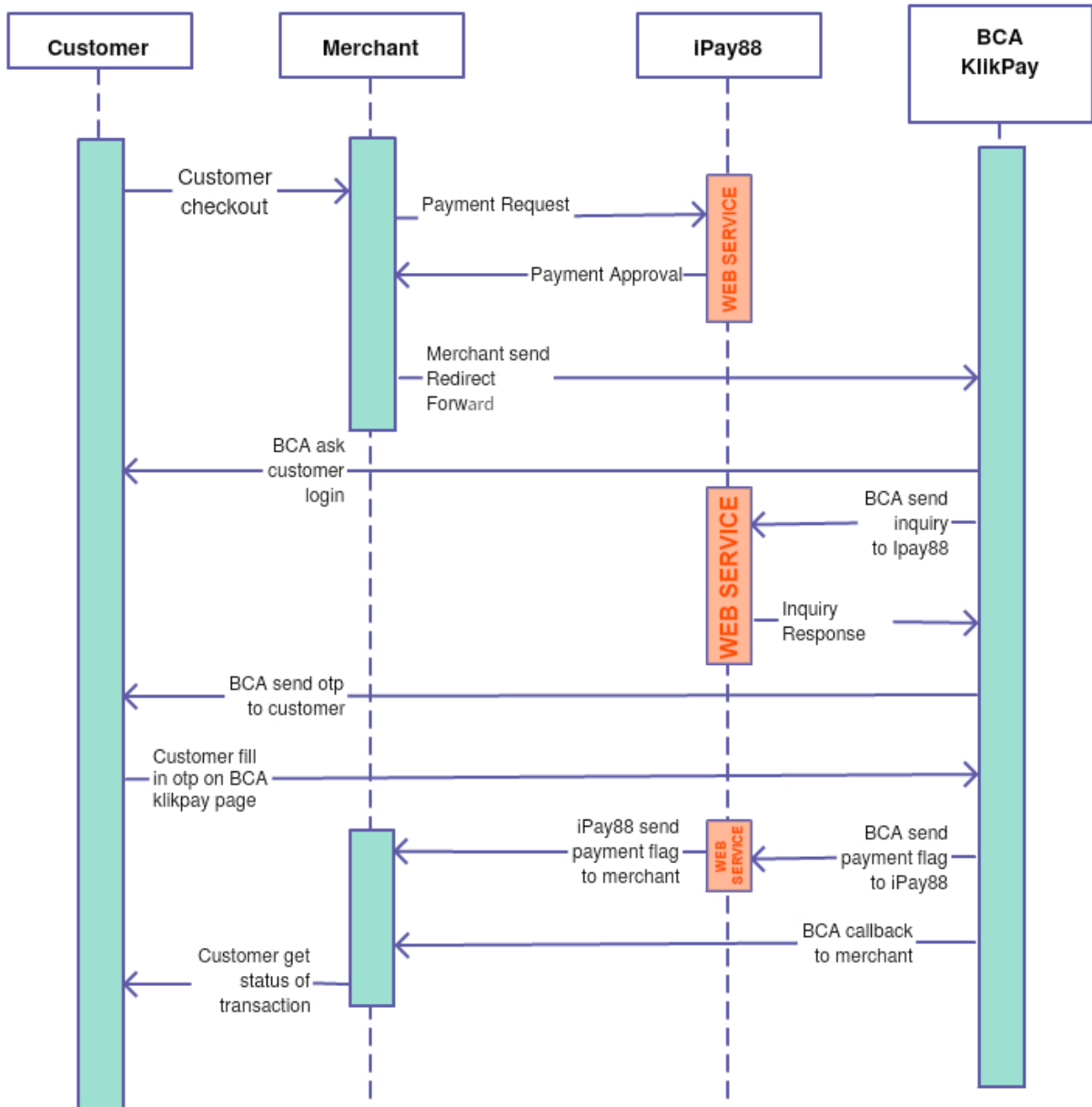


Table of Contents

I. INTRODUCTION	5
II. SYSTEM PROCESS	5
II. 1. Payment Request	5
II. 1. a. Request.....	5
II. 1. b. Request Signature Algorithm	10
II. 1. c. Payment Approval	10
II. 1. d. Payment Approval Signature Algorithm	11
II. 2. Redirect Forward.....	12
III. 2. a. Input	12
II. 2. b. Signature Algorithm	13
II. 3. Payment Flag Invocations to Merchants.....	14
II. 3. a. Transaction Flow	14
II. 3. b. Input	14
II. 3. c. Auth Key Algorithm.....	15
II.3.d. Output	15
III. Key ID For Transaction	16

Flowchart







I. INTRODUCTION

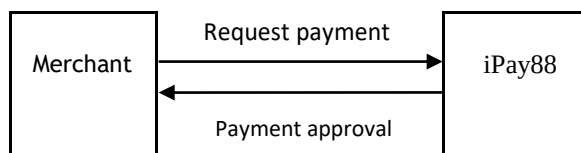
This document was made for the online shopping payment using KlikPay project in BCA, supporting the needs of business process summary and data movements in technical way; from BCA to merchants. This file describes the system's process, data flow, and fields that are needed for either BCA or merchants to communicate.

The main concept of online transaction payment to be implemented is using KlikPay account to do payment of all the online shopping transactions that have been made on merchants websites.

II. SYSTEM PROCESS

The system process is separated in three sub processes, namely payment request, redirect forward, and payment flag invocation to merchants.

II. 1. Payment Request



When customers choose the payment using KlikPay on merchant's site, first merchant's will check that total amount of transaction not exceed KlikPay transaction limit. After that, merchant will be request payment. The following table is the data which sent using web service. URL used for payment request are:

- UAT : <https://sandbox.ipay88.co.id/ePayment/WebService/SeamlessPayment/Entry.asmx>
- Production : <https://payment.ipay88.co.id/ePayment/WebService/SeamlessPayment/Entry.asmx>

II. 1. a. Request

Request Format				
No.	Field Name	Type (Size)	Sample	Description
1.	MerchantCode	varchar(20)	ID00001	Merchant code assigned by iPay88
2.	PaymentId	Integer	8	Only support BCA KlikPay (8)
3.	Currency	varchar(5)	IDR	Only support for IDR
4.	RefNo	varchar(20)	12232015104601AM	
5.	Amount	Integer	300000	300000 mean 3000 amount of transaction Amount = FullTransactionAmount +



				installmentTransactions [n].amount + MiscFee
6.	ProdDesc	varchar(100)	Empat hari	
7.	UserName	varchar(100)	Ali	
8.	UserEmail	varchar(100)	test@ipay88.co.id	
9.	UserContact	varchar(20)	60133725811	
10.	Remark	varchar(100)		
11.	Lang	varchar(20)	iso-8859-1	Encoding type "ISO-8859-1" – English "UTF-8" – Unicode "GB2312" – Chinese Simplified "GD18030" – Chinese Simplified "BIG5" – Chinese Traditional
12.	ResponseURL	varchar(200)	<a href="https://store.co.id/res
p.asp">https://store.co.id/res p.asp	
13.	BackendURL	varchar(200)	<a href="https://store.co.id/ba
ckend.asp">https://store.co.id/ba ckend.asp	
14.	Signature	varchar(100)	FCBp9ru3eRHATm0EG lhfmzORDo	
15.	xfield1	varchar(20)		
16.	MiscFee	varchar(12)	50000	50000 mean 500
17.	FullTransactionAmount	List (n)	100000	100000 mean 1000 Amount of full payment product
18.	InstallmentTransactions	List (n)		Only for installment detail with multiple product
	17.1 itemName	string(50)	Book	Name of item installment
	17.2 quantity	Integer(3)	2	Quantity of installment product
	17.3 amount	Integer (12)	150000	150000 mean 1500 amount of installment product
	17.4 tenor	Integer (2)	03	
	17.5 codePlan	string (3)	000	
	17.5 merchantId	Integer (9)	1241	MID from BCA

Example :

- Use Web Service
 - Full Payment
- POST [DOMAIN]/ePayment/WebService/SeamlessPayment/Entry.asmx
HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: application/soap+xml; charset=UTF-8;
SOAPAction="Request"

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-  
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  
  <soap:Body>  
    <Request xmlns="http://www.ipay88.co.id">  
      <MerchantCode>ID00001</MerchantCode>  
      <PaymentId>8</PaymentId>  
      <Currency>IDR</Currency>  
      <RefNo>117201783440AM</RefNo>  
      <Amount>100000000</Amount>  
      <ProdDesc>Product Testing</ProdDesc>  
      <UserName>Ali</UserName>  
      <UserEmail>test@ipay88.co.id</UserEmail>  
      <UserContact>60133725811</UserContact>  
      <Remark></Remark>  
      <Lang></Lang>  
      <ResponseURL>http://www.aa.com/response.asp</ResponseURL>  
      <BackendURL>http://www.aa.com/backend_response.asp</BackendURL>  
    </Request>  
    <Signature>42QMKfAYvkaUUVpP9MT081ootGY=</Signature>  
    <xfield1></xfield1>  
    <FullTransactionAmount>100000000</FullTransactionAmount>  
    <MiscFee>000</MiscFee>  
    <InstallmentTransactions>  
      <installmentTransaction>  
        <itemName></itemName>  
        <quantity></quantity>  
        <amount></amount>  
        <tenor></tenor>  
        <codePlan></codePlan>  
        <merchantId></merchantId>  
      </installmentTransaction>  
    </InstallmentTransactions>  
  </soap:Body>  
</soap:Envelope>
```



o Installment

POST [DOMAIN]/ePayment/WebService/SeamlessPayment/Entry.asmx

HTTP/1.1

Accept-Encoding: gzip, deflate

Content-Type: application/soap+xml; charset=UTF-8;

SOAPAction="Request"

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <Request xmlns="http://www.ipay88.co.id">
      <MerchantCode>ID00001</MerchantCode>
      <PaymentId>8</PaymentId>
      <Currency>IDR</Currency>
      <RefNo>117201783440AM</RefNo>
      <Amount>100000000</Amount>
      <ProdDesc>Buku</ProdDesc>
      <UserName>Ali</UserName>
      <UserEmail>test@ipay88.co.id</UserEmail>
      <UserContact>60133725811</UserContact>
      <Remark></Remark>
      <Lang></Lang>
      <ResponseURL>http://www.aa.com/response.asp</ResponseURL>
      <BackendURL>http://www.aa.com/backend_response.asp</BackendURL>
      <Signature>42QMKfAYvkaUUVpP9MT08l0otGY=</Signature>
      <xfield1></xfield1>
      <FullTransactionAmount></FullTransactionAmount>
      <MiscFee>000</MiscFee>
      <InstallmentTransactions>
        <installmentTransaction>
          <itemName>Buku gambar</itemName>
          <quantity>2</quantity>
          <amount>30000000</amount>
          <tenor>06</tenor>
          <codePlan>000</codePlan>
          <merchantId>12323</merchantId>
        </installmentTransaction>
        <installmentTransaction>
          <itemName>Buku Tulis</itemName>
          <quantity>2</quantity>
          <amount>30000000</amount>
          <tenor>03</tenor>
          <codePlan>000</codePlan>
          <merchantId>12322</merchantId>
        </installmentTransaction>
        <installmentTransaction>
          <itemName>Buku Cetak</itemName>
          <quantity>2</quantity>
          <amount>40000000</amount>
          <tenor>09</tenor>
          <codePlan>000</codePlan>
          <merchantId>12324</merchantId>
        </installmentTransaction>
      </InstallmentTransactions>
    </Request>
  </soap:Body>
</soap:Envelope>
```

- Use PHP

```
<?php
$client = new
SoapClient("https://sandbox.ipay88.co.id/epayment/WebService/Seamless
Payment/Entry.asmx?WSDL");
$client->
__setLocation('https://sandbox.ipay88.co.id/ePayment/WebService/Seam
lessPayment/Entry.asmx');
$params = new stdClass();
$params->MerchantCode = 'ID00001';
$params->PaymentId = '8';
$params->Currency = 'IDR';
$params->RefNo = '117201783440AM';
$params->Amount = '100000000';
$params->ProdDesc = 'Product Testing';
$params->UserName = 'Ali';
$params->UserEmail = 'test@ipay88.co.id';
$params->UserContact = '60133725811';
$params->Remark = '';
$params->Lang = 'iso-8859-1';
$params->ResponseURL = 'http://www.aa.com/response.asp';
$params->BackendURL = 'http://www.aa.com/backend_response.asp';
$params->Signature = '42QMKfAYvkaUUvPp9MT08lootGY=';
$params->xfield1 = '';
$params->MiscFee = '000';
$params->FullTransactionAmount = '50000000';
$installmentTransaction = new stdClass();
    $installmentTransaction->itemName = 'test product';
    $installmentTransaction->quantity = '2';
    $installmentTransaction->amount = '50000000';
    $installmentTransaction->tenor = '03';
    $installmentTransaction->codePlan = '000';
    $installmentTransaction->merchantId = '12322';
$params->InstallmentTransactions[0] = $installmentTransaction;
//approval ipay88
$results = $client->Request($params);
//print_r($results);
$result = $results[0];
    $Amount          = $result->Amount;
    $Currency        = $result->Currency;
    $ErrDesc         = $result->ErrDesc;
    $MerchantCode     = $result->MerchantCode;
    $PaymentId       = $result->PaymentId;
    $RefNo           = $result->RefNo;
    $Remark          = $result->Remark;
    $Status          = $result->Status;
    $TransId         = $result->TransId;
    $Signature        = $result->Signature;
    echo "ErrDesc : " . $ErrDesc . "</br>";
    echo "Status : " . $Status . "</br>";
    echo "TransId : " . $TransId . "</br>";
?>
```



II. 1. b. Request Signature Algorithm

Parameters Used :

- KlikPay Code
- Transaction Time (format: ddMMyyyy)
- Transaction Number
- Total Amount (note : format in integer value, no decimal)
- Currency
- Key Id (current key used for transaction)

Using PHP you can use the following code to perform this:

```
<?php
    include_once "GenerateKey.php"
    $keyId = generateKeyId($clearKey);
    $signature = generateSignature($klikPayCode, $transactionDate,
    $transactionNo, $totalAmount, $currency, $keyId);
?>
```

Note: [GenerateKey.php](#) file has all functions for klikpaybca purpose. iPay88 will provide that file as example to generate signature in PHP.

II. 1. c. Payment Approval

Payment Response Format				
No.	Field Name	Type (Size)	Sample	Description
1.	MerchantCode	varchar(20)	ID00001	Merchant code assigned by iPay88
2.	PaymentId	Integer	8	Only support BCA KlikPay (8)
3.	Currency	varchar(5)	IDR	Only support for IDR
4.	RefNo	varchar(20)	12232015104601AM	
5.	Amount	Integer	300000	300000 mean 3000 amount of transaction
6.	Remark	varchar(100)		
7.	Signature	varchar(100)	FCBp9ru3eRHATm0EGlhfzimORDo	
8.	xfield1	varchar(20)		
9.	TransId	varchar(30)	T0036521900	Transaction to iPay88
10.	Status	Integer(2)	6	Payment Status "6" – Pending "0" – Fail
11.	ErrDesc	varchar(100)	Signature not match	

Example :

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <RequestResponse xmlns="http://www.ipay88.co.id">
      <RequestResult>
        <Status>6</Status>
        <ErrDesc/>
        <MerchantCode>ID00001</MerchantCode>
        <PaymentId>8</PaymentId>
        <RefNo>117201783440AM</RefNo>
        <Amount>100000000</Amount>
        <Currency>IDR</Currency>
        <Remark/>
        <xfield1/>
        <Signature>jvZBU7kxCMKijkhEc1p8CFZnf50=</Signature>
        <TransId>T0000970500</TransId>
        <AuthCode/>
      </RequestResult>
    </RequestResponse>
  </soap:Body>
```

II. 1. d. Payment Approval Signature Algorithm

Parameters Used:

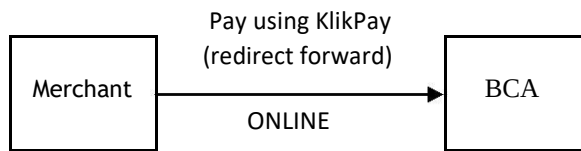
- KlikPay Code
- Transaction Time (format: ddMMyyyy)
- Transaction Number
- Total Amount (note : format in integer value, no decimal)
- Currency
- Key Id (current key used for transaction)

Using PHP you can use the following code to perform this:

```
<?php
include_once "GenerateKey.php"
$keyId = generateKeyId($clearKey);
$signature = generateSignature($klikPayCode, $transactionDate,
$transactionNo, $totalAmount, $currency, $keyId);
?>
```

Note: [GenerateKey.php](#) file has all functions for klikpaybca purpose. iPay88 will provide that file as example to generate signature in PHP.

II. 2. Redirect Forward



Picture 1

After receive payment approval from ipay88, merchant must set redirected forward, so customer will be login in KlikPay. The following table is the data which sent using POST when redirect forward. URL used for redirect forward are :

- Development : <https://sandbox.ipay88.co.id/klikpaybca/purchasing/purchase.asp?action=loginRequest>
- UAT : <https://sb-ipay.u-appspecto.com:8081/purchasing/purchase.do?action=loginRequest>
- Production : <https://klikpay.klikbca.com/purchasing/purchase.do?action=loginRequest>

III. 2. a. Input

The following table specifies input fields and types.

Field#	Name	Description	Type	Length	Format	Mandatory
1.	klikPayCode	KlikPay Code	String	10	(AN)	TRUE
2.	transactionNo	Transaction Number	String	18	(AN)	TRUE
3.	totalAmount	Total Amount	String	12	999999999.00	TRUE
4.	currency	Currency	String	5	(A)	TRUE
5.	payType	Payment Type	String	2	99	TRUE
6.	callback	Call Back URL	String	100	(AN)	TRUE
7.	transactionDate	Transaction Date	String	19	DD/MM/YYYY hh:mm:ss	TRUE
8.	Descp	Description	String	60	(AN)	FALSE
9.	miscFee	Miscellaneous Fee	String	12	999999999.00	FALSE
10.	signature	Signature	String	10	(N)	TRUE

Note:

- (AN) is an alpha-numeric format.
- 2 last digits for totalAmount, miscFee field should be decimal point and decimal point should be valued .00.
- *totalAmount* field value should be total amount for transaction but not including *miscFee*.
- *payType* field may vary with these values :
 - 01 = full
 - 02 = installment
 - 03 = 01 and 02
- *currency* field may vary with these values :
 - IDR

Example :

```
<HTML>
<BODY>
<FORM method="post" name="ePayment"
action="https://klikpay.klikbca.com/purchasing/purchase.do?action=loginRequest">
<INPUT type="hidden" name="klikPayCode" value="BCAPAY0001">
<INPUT type="hidden" name="clearKey" value="ClearKey00000001">
<INPUT type="hidden" name="transactionNo" value="R811184">
<INPUT type="hidden" name="totalAmount" value="3000\">
<INPUT type="hidden" name="Currency" value="IDR">
<INPUT type="hidden" name="payType" value="01">
<INPUT type="hidden" name="transactionDate" value="17/04/2014 11:24:55">
<INPUT type="hidden" name="descp" value="test Pembelian">
<INPUT type="hidden" name="miscFee" value="0.00">
<INPUT type="hidden" name="signature" value="240101082">
<INPUT type="hidden" name="callback"
value="https://dv11.ipay88.co.id/klikpaybca/purchasing/callback.asp?trxno=R811184"
>
<INPUT type="submit" value="Proceed with Payment" name="Submit">
</FORM>
</BODY>
</HTML>
```

II. 2. b. Signature Algorithm

Parameters Used :

- KlikPay Code
- Transaction Time (format: ddMMyyyy)
- Transaction Number
- Total Amount (note : format in integer value, no decimal)
- Currency
- Key Id (current key used for transaction)

Using PHP you can use the following code to perform this:

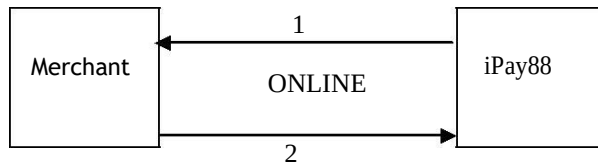
```
<?php
    include_once "GenerateKey.php"
    $keyId = generateKeyId($clearKey);
    $ssSignature = generateSignature($klikPayCode, $transactionDate,
$transactionNo, $totalAmount, $currency, $keyId);
?>
```

Note: [GenerateKey.php](#) file has all functions for klikpaybca purpose. iPay88 will provide that file as example to generate signature in PHP.

II. 3. Payment Flag Invocations to Merchants

From the first process (list inquiry of transactions), after submitting all transactions, BCA will deduct customer's account (debit and/or credit account) and send the payment confirmation to be acknowledged by merchant.

II. 3. a. Transaction Flow



Picture 3

Note:

1 = payment flag from BCA to Merchant

2 = response from Merchant to BCA

II. 3. b. Input

The following table specifies input fields and types.

Field#	Name	Type	Length	Format	Mandatory
1.	klikPayCode	string	10	(AN)	TRUE
2.	transactionDate	string	19	DD/MM/YYYY hh:mm:ss	TRUE
3.	transactionNo	string	18	(AN)	TRUE
4.	currency	string	5	(A)	TRUE
5.	totalAmount	number	12	999999999.00	TRUE
6.	payType	string	2	99	TRUE
7.	approvalCodeFullTransaction	number	6	(N)	FALSE
8.	approvalCodeInstallmentTransaction	number	6	(N)	FALSE
9.	authKey	string	32	(N)	TRUE
10.	additionalData	string	999	(AN)	FALSE

Note:

- (AN) is an alpha numeric format.
- Approval Code for fullTransaction will be filled if customer have a full transaction and choose to pay that full transaction using credit card.
- Approval Code for installmentTransaction will be filled if customer have an installment transaction.
- totalAmount field value should be total amount for transaction, same with total amount at redirect forward.
- *2 last digits for totalAmount field should be decimal point and decimal point should be valued .00.*

- *payType* field may vary with these values :
 - 01 = full
 - 02 = installment
 - 03 = 01 and 02
- *currency* field may vary with these values :
 - *IDR*

II. 3. c. Auth Key Algorithm

AuthKey Flag Transaction :

$$[[A B C D K]^h]^e_K$$

Parameters Used :

A = klikPayCode
B = transactionNo
C = currency
D = transactionDate
K = keyId (current key used for transaction)
 e_K = encrypt using 3DES by keyId
h = hash function

```
<?php
    include_once "GenerateKey.php"

    $keyId = generateKeyId($clearKey);

    $sAuthKey = generateAuthKey($klikPayCode, $transactionNo, $currency,
    $transactionDate, $keyId);
```

II.3.d. Output

The following table specifies output fields and types.

Field#	Name	Type	Length	Format	Nullable
1.	status	string	2	(AN)	FALSE
2.	english	string	50	(AN)	FALSE
3.	indonesian	String	50	(AN)	FALSE



Sample in PHP code:

```
<?php
$vklikPayCode = $_REQUEST['klikPayCode'];
$transactionNo = $_REQUEST['transactionNo'];
$transactionDate = $_REQUEST['transactionDate'];
$vcurrency = $_REQUEST['currency'];
$vtotalAmount = $_REQUEST['totalAmount'];
$vpayType = $_REQUEST['payType'];
$vapprovalCodeFullTransaction =
$_REQUEST['approvalCodeFullTransaction'];
$vapprovalCodeInstallmentTransaction =
$_REQUEST['approvalCodeInstallmentTransaction'];
$vauthKey = $_REQUEST['authKey'];
$vadditionalData = $_REQUEST['additionalData'];

//Do Authkey & Amount checking, write success if correct
echo "00:||:Success:||:Sukses.";
```

III. Key ID For Transaction

Key ID used for transaction :

Uppercase [to String [Hexa[clearKey]]]

Description :

clearKey	= clear key given by BCA
bytes	= array of bytes from clear key
hexArray	= array of char for Hexa character
keyId	= string of hexa from clear key, known as <u>current key used for transaction</u>

APPENDIX

A. Response Codes

The following table specifies codes used for each response from the merchant.

No.	Response Code	Response Code
1.	00	TRX_SUCCESS (*)
2.	01	TRX_REJECTED (*)

B. Reason (Error Message)

The following table specifies an example reason that can be used by merchant.

Condition	Reason (Error Message)
Your transaction has been canceled	Your transaction has been canceled
Your transaction has been paid	Your transaction has been paid
Your transaction has expired or exceeded time limit	Your transaction has expired or exceeded time limit
Technical problem	Your transaction cannot be processed
Transaction data (amount, etc.) from bank is different with transaction data in the merchant	Transaction is invalid

Note :

The reason (error message) will be displayed on KlikPay.

Sample Signature and AuthKey using PHP Script :

```
<?php
function toHexString($string){
    $hex = '';
    for ($i=0; $i<strlen($string); $i++){
        $ord = ord($string[$i]);
        $hexCode = dechex($ord);
        $hex .= substr('0'.$hexCode, -2);
    }
    return strtoupper($hex);
}
function fromHexString($hex){
    $string='';
    for ($i=0; $i < strlen($hex)-1; $i+=2){
        $string .= chr(hexdec($hex[$i].$hex[$i+1]));
    }
    return $string;
}

function generateSignature($klikPayCode, $transactionDate, $transactionNo,
$amount, $currency, $keyId){
    date_default_timezone_set (
        "Asia/Jakarta");
    $tempKey1 = $klikPayCode . $transactionNo . $currency . $keyId;
    $hashKey1 = getHash($tempKey1);
    $tdate = date_create_from_format("d/m/Y G:i:s",$transactionDate);
    $idate = (int)date_format($tdate, 'dmY');
    $iamount = (int)$amount;
    $tempKey2 = ($idate + $iamount);
    $tempKey2 = $tempKey2 . "";
    $hashKey2 = getHash($tempKey2);
    $hash =
        abs($hashKey1+$hashKey2);
    return $hash."";
}
function generateAuthKey($klikPayCode, $transactionNo, $currency,
$transactionDate, $keyId){
    $str1 = str_pad($klikPayCode,10, '0', STR_PAD_RIGHT) .
    str_pad($transactionNo,18, 'A', STR_PAD_RIGHT) . str_pad($currency,5,
    '1',STR_PAD_RIGHT) . str_pad($transactionDate,19, 'C', STR_PAD_LEFT) .
    $keyId;
    $mdstr1 = md5($str1);
    $my_key = fromHexString($keyId);
    $data = fromHexString($mdstr1);
    $my_key .= substr($my_key,0,8);
    $secret = mcrypt_encrypt(MCRYPT_3DES, $my_key, $data,
    MCRYPT_MODE_ECB); return toHexString($secret);
}
function
generateKeyId($st
r){ return
toHexString($str)
;
}

?>
```

Sample Signature and AuthKey using PHP Script (cont.):

```
<?php
function getHash ($value)
{
    $hash = 0;
    for ($i = 0; $i < strlen($value); $i++)
    {
        $hash = (intval($hash) * 31) + ord($value[$i]);
        $hash = intval32bits($hash);
    }
    return (int)$hash;
}
function intval32bits($value)
{
    $value = ($value & 0xFFFFFFFF);
    if ($value & 0x80000000)
        $value = -((~$value & 0xFFFFFFFF) + 1);
    return $value;
}

$clearKey      = "ClearKey000000001";
$klikPayCode   = "BCAPAY0001";
$transactionDate = "11/04/2014 16:15:53"; //format: "d/m/Y G:i:s"
$transactionNo  = "R223248";
$totalAmount   = 3000.00;
$currency       = "IDR";
$keyId = generateKeyId($clearKey);
$ssSignature = generateSignature($klikPayCode, $transactionDate,
    $transactionNo, $totalAmount, $currency, $keyId);
$ssAuthKey = generateAuthKey($klikPayCode, $transactionNo, $currency,
    $transactionDate, $keyId);

echo $keyId . "<br/>" . $ssSignature . "<br/>" . $ssAuthKey;
/*
Result Should be: 436C6561724B65793030303030303031
1375293638
8DAED070984351646CD1D9C8255404F4
*/
?>
```

Sample Signature and AuthKey using C# Script :

```
using System;
using System.Text;
using System.Security.Cryptography;
using System.IO;

public class GenerateKey
{
    private static String HEX_DIGITS = "0123456789ABCDEF";

    private static int getHash(String val)
    {
        int h = 0;
        char[] vals = val.ToCharArray();
        for (int i = 0; i < vals.Length; i++)
        {
            h = (h * 31) + (int)vals[i];
        }
        return h;
    }

    public static String generateSignature(String klikPayCode, String
transactionDate, String transactionNo, String amount, String currency, String
keyId)
    {
        // Generate Key Step 1
        String tempKey1 = klikPayCode + transactionNo + currency + keyId;
        int hashKey1 = getHash(tempKey1);

        // Generate Key Step 2
        // Reformat Input Date
        DateTime date = new DateTime();
        date = DateTime.ParseExact(transactionDate, "dd/MM/yyyyHH:mm:ss", null);

        int transDate = Convert.ToInt32(date.ToString("ddMMyyyy"));
        int amt = (Int32)Convert.ToDouble(amount);

        String tempKey2 = (transDate + amt).ToString();
        int hashKey2 = getHash(tempKey2);

        // Hash the value using integer arithmetic and absolute value
        Decimal hash = Math.Abs(decimal.Add(new Decimal(hashKey1), new
Decimal(hashKey2)));
        return hash.ToString();
    }
}
```

Sample Signature and AuthKey using C# Script (cont.) :

```
private static byte[] fromHexString(String s)
{
    byte[] bytes = new byte[s.Length / 2];
    for (int i = 0; i < s.Length / 2; i++)
    {
        bytes[i] = (byte)(Convert.ToInt32(s.Substring(2 * i, 2), 16));
    }
    return bytes;
}

private static String toHexString(byte[] b)
{
    if ((b == null) || (b.Length == 0))
    {
        return "";
    }
    else
    {
        return toHexString(b, 0, b.Length);
    }
}

/** Convert bytes to HEX string */
private static String toHexString(byte[] b, int off, int len)
{
    StringBuilder s = new StringBuilder();
    for (int i = off; i < off + len; i++)
    {
        s.Append(HEX_DIGITS[(b[i] & 0xff) >> 4]); s.Append(HEX_DIGITS[b[i] & 0xf]);
        // or
        //s.Append((b[i]).ToString("X2"));
    }
    return s.ToString();
}

public static String generateAuthKey(String klikPayCode, String transactionNo,
String currency, String transactionDate, String keyId)
{
    return generateAuthKey(klikPayCode.PadRight(10, '0') + transactionNo.PadRight(18,
'A') + currency.PadRight(5, '1') + transactionDate.PadLeft(19, 'C') + keyId,
keyId);
}

private static String generateAuthKey(String secondValue, String keyId)
{
    MD5 sha = MD5.Create();
    byte[] byteHash = sha.ComputeHash(Encoding.UTF8.GetBytes(secondValue)); ;
    String hashData = toHexString(byteHash);
}
```

Sample Signature and AuthKey using C# Script (cont.) :

```
byte[] byteHashData = fromHexString(hashData);
byte[] byteKey = fromHexString(keyId);

if (!(byteKey.Length == 16 || byteKey.Length == 24))
{
    throw new Exception("Length not valid :" + byteKey.Length);
}

byte[] key = new byte[24];
if (byteKey.Length == 16)
{
    for (int za = 0; za < 16; za++)
    {
        key[za] = byteKey[za];
    }
    for (int za = 0; za < 8; za++)
    {
        key[za + 16] = byteKey[za];
    }
}

if (byteKey.Length == 24)
{
    for (int za = 0; za < 24; za++)
    {
        key[za] = byteKey[za];
    }
}

MemoryStream ms = new MemoryStream();
TripleDESAlg = TripleDES.Create();
alg.Key = key;
alg.Mode = CipherMode.ECB;
alg.Padding = PaddingMode.None;
CryptoStream cs = new CryptoStream(ms, alg.CreateEncryptor(),
    CryptoStreamMode.Write);
cs.Write(byteHashData, 0, byteHashData.Length);
cs.FlushFinalBlock();
byte[] cipherBytes = ms.ToArray();
ms.Close();
cs.Close();

String encryptedHashKey = toHexString(cipherBytes);

return encryptedHashKey;
}

public static String generateKeyId(String clearKey)
{
    byte[] bytes = Encoding.UTF8.GetBytes(clearKey);
    return toHexString(bytes);
}

String keyId =
    BcaKeyGenerator.GenerateKey.generateKeyId(clearKey);
sSignature = BcaKeyGenerator.GenerateKey.generateSignature(klikPayCode,
    transactionDate, transactionNo, totalAmount, currency, keyId);
sAuthKey = BcaKeyGenerator.GenerateKey.generateAuthKey(klikPayCode,
    transactionNo, currency, transactionDate, keyId);
```