

MERCHANT INTEGRATION MANUAL

Merchant Integration Manual

Name	Merchant Integration Manual
Security	Public document
Revision	4.7

Content

1	Introduction.....	3
1.1	Terms and Terminology	3
1.2	Purpose of the Document	3
2	24pay Integration	4
2.1	Configuration Data	4
2.2	Process Flow.....	4
2.3	Graphical elements	5
3	Payment Protocol	6
3.1	Payment Request from Merchant.....	6
3.2	Notification of Payment Processing Status from 24pay	8
3.3	Redirect Client to Merchant.....	9
3.4	Complete / Cancel Pre-authorized Payment	10
3.5	SIGN	11
3.5.1	Security key	12
3.5.2	Checksum	12
3.5.3	Request from Merchant	12
3.5.4	Notification of Payment Processing Status from 24pay	12
3.5.5	Redirect Client to Merchant.....	13
3.5.6	Complete / Cancel Pre-authorized Payment	13
4	Attachments	14
4.1	Example of Sign Creation	14
4.1.1	Payment Request	14
4.1.2	Notification of Payment Processing	15
4.1.3	Complete Pre-authorized Payment	16
4.1.4	Prípád dokončenia predautorizovanej platby	17
4.1.5	Code Samples.....	18
4.2	Payment form	20

1 Introduction

1.1 Terms and Terminology

PSP	payment service provider
24pay	automated payment institution - payment service provider
Merchant	online shop providing goods/services, receiving payments
Client	person purchasing goods/services, making payments
RURL	Redirection Return URL – URL address of the eshop, where the customer is redirected after transaction
NURL	Notification Return URL – URL address, where the notification of change in the payment status is sent via HTTP / HTTPS POST method within the body request.

1.2 Purpose of the Document

The purpose of this document is to describe the communication protocol between the merchant web server and payment system interface 24pay. It serves as a technical manual for the services provided by the 24pay and describes the steps how to correctly connect and communicate with payment interface.

The document is not a guide for creating web pages. Its mission is to list and describe the conditions that merchant web has to meet to successfully realize the payment services.

2 24pay Integration

2.1 Configuration Data

The following section describes a set of data that merchant system and 24pay exchange with each other.

Merchant lists the following information:

- RURL
- NURL

24pay provides merchant the following information:

- Mid
- EshopId
- Key

2.2 Process Flow

The purpose of this section is to outline a model of processing and realization of a payment session showing the interaction between actors: client - merchant - 24pay.

Merchant payment page contains a link to 24pay. Customer who chose 24pay as desired payment method, sends from the merchant system to 24pay request for payment. The request contains a prescribed set of data required for processing and execution of a payment session.

Client is redirected to a payment portal of bank institution. Afterwards the client confirms or cancels the payment. 24pay realizes the necessary steps of processing a payment session, sends a notification message containing the status of the transaction and redirects the client back to the merchant site.

In the case of non-compliant format/content parameters of the requests, the customer is redirected to the 24pay page informing about the unsuccessful execution of a payment.

24pay sends notification of payment result to the merchant address specified by configuration NURL. Merchant serverside has an obligation to respond HTTP status 200 OK, acknowledging receipt of the response. The merchant system can run additional steps relating to the transaction. In case of result PENDING in notification, additional notification message with result OK/FAIL is send to NURL after processing.

24pay redirects client via method GET on the merchant's site specified by configuration RURL. Return addresses contain the string parameter informing about the payment processing result on which merchant system can inform the client about successful or

unsuccessful processing. Return addresses are for information purposes only, on that basis, it is not possible to make any decisions.

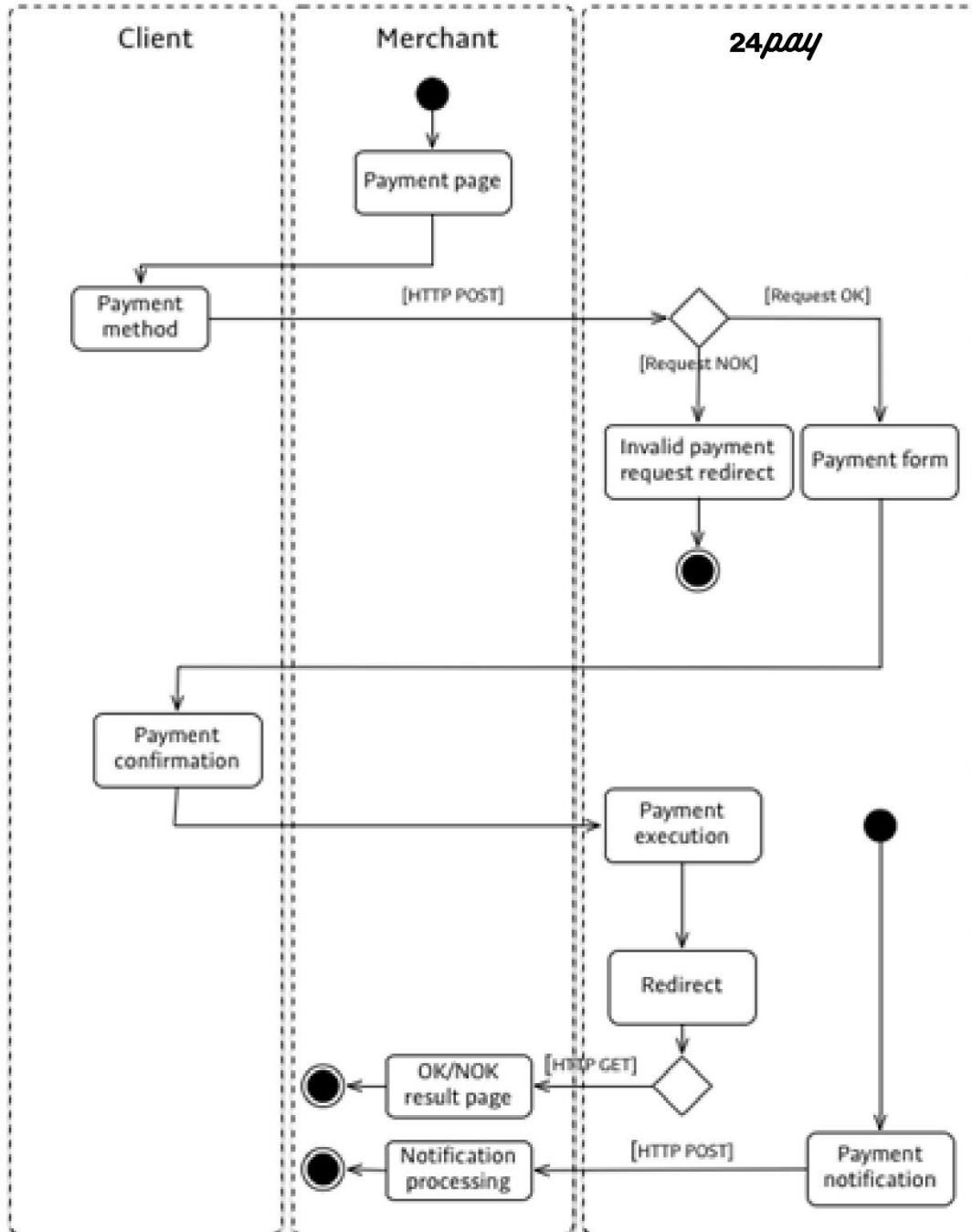


Image 1. Process model

2.3 Graphical elements

To view the payment button and logo on the page, use the logo 24pay, which is referred on: <http://www.24-pay.sk/na-stiahnutie/>

3 Payment Protocol

3.1 Payment Request from Merchant

To send a new payment request there must be a form which redirects the customer to the 24pay payment gateway **paymentRequestGateURL** on merchant web site.

URL 24pay payment gateway:

https://admin.24-pay.eu/pay_gate/paygt

The condition is the creation of HTTPS POST method request. The data will be coded as application/x-wwwform-urlencoded. The list of parameters is given in the table below:

Parameter	Mandatory	Format	Length	Description	Example
Mid	●	Alpha-numeric	8	Merchant identifier (case sensitive)	1a2B3c4D
EshopId	●	Numeric	1..10	E-shop identifier	135
MsTxnId	●	Alpha-numeric	1..32	Merchant system transaction identifier – unique identifier of payment provided by merchant (variable symbol).	1234567890
Amount	●	#0.00	1..10,2	The amount of payment. Decimal separator is a dot. Decimal portion is represented by two digits.	1.00
CurrAlphaCode	●	AAA	3	Currency of payment ISO 4217	EUR
ClientId	●	Alpha-numeric	3..10	Client identifier in merchant system (case sensitive)	12345
FirstName	●	Alphabetic	2..50	Client – first name	Jožko
FamilyName	●	Alphabetic	2..50	Client – family name	Mrkvička
Email	●	email	6..128	Client – email address	jozko.mrkvicka@demo.com
Country	●	AAA	3	Client – country of residence ISO 31661	SVK

Timestamp	●	yyyy-MM-dd HH:mm:ss	19	Timestamp of payment request. Date and time separator is a space character! Timestamp and MsTxnId have to form unique combination.	2014-12-01 13:00:00
Sign	●	Alpha- numeric	32	Checksum of transmitted parameters	
LangCode		aa	2	ISO 639-1 language code. sk, cs, en, de, hu, es, fr, it, pl. Default sk..	sk
RURL		URL	256	Redirection Return URL – URL address, where client is redirected after transaction. It overlays configured item RURL if it is presented.	http://mojobchod.sk/ rurl
NURL		URL	256	Notification Return URL – URL address, where notifications about change of payment status are sent via HTTP/HTTPS POST method within the body request. It overlays configured item NURL if it is presented.	http://mojobchod.sk/ nurl
NotifyEmail		email	6..128	Email address, where additional notifications about change of payment status are sent.	platby@mojobchod.s k
RedirectSign		true/false	4/5	Option for adding sign to redirection.	false
PreAuthProvided		true/false	4/5	Option for preauthorization of payment (only for cards).	false
Phone		Alph a- numeric	8..25	Client – phone	0901 000 001
Street		Alph a- numeric	3..50	Client – street	Kvetná 123
City		Alph abetic	2..50	Client – city	Bratislava
Zip		Alph a- numeric	1..10	Client – zip	821 08

3.2 Notification of Payment Processing Status from 24pay

After completion of the payment process 24pay notifies about the payment processing status. Message is sent within the HTTP POST request addressed to NURL.

Data relating to the payment are transmitted as a structure having an XML format as the value of the parameter params.

Example of notification:

```
<?xml version="1.0" encoding="UTF-8"?>
<Response sign="21f22ef2af21d3819cd0cff06ef55943">
  <Transaction>
    <Identification>
      <MsTxnId>1234567890</MsTxnId>
      <PspTxnId>0987654321</PspTxnId>
    </Identification>
    <Presentation>
      <Amount>1.00</Amount>
      <Currency>EUR</Currency>
    </Presentation>
    <Customer>
      <Contact>
        <Email> jozko.mrkvicka@demo.com</Email>
        <Phone>0901 000 001</Phone>
      </Contact>
      <Address>
        <Street>Kvetná 123</Street>
        <Zip>821 08</Zip>
        <City>Bratislava</City>
        <Country>SVK</Country>
      </Address>
      <Name>
        <Given>Jožko</Given>
        <Family>Mrkvička</Family>
      </Name>
    </Customer>
    <Processing>
      <Timestamp>2014-12-01 13:01:00.548</Timestamp>
      <Result>OK</Result>
      <Reason code="00">Successful Processing</Reason>
      <PSPCategory>2</PSPCategory>
      <CreditCard/>
    </Processing>
  </Transaction>
</Response>
```

<Result> refers the status of your payment. It can have the following values:

- **OK** – payment successful.
- **FAIL** – payment failed.
- **PENDING** – payment was sent for processing. After processing the payment new notification is sent, where the **<Result>** is either OK or FAIL.
- **AUTHORIZED** – pre-authorization of payment was successful. You can complete

or cancel payment within 7 days.

<PSPCategory> indicates the category of payment method that client used for payment.

- 1 – credit card
- 2 – instant payment
- 3 – bank transfer
- 4 – other

3.3 Redirect Client to Merchant

After the payment is complete, client is redirected back to merchant to **RURL** given by merchant. Redirection is done via HTTP GET request, the query string contains parameters carrying information about successful or unsuccessful result of payment processing.

It is important to realize that **RURL** is only for **informative** purposes. Based on the data received in redirection it is not possible to make any decisions. The list of parameters sent in the query string is as follows:

Parameter	Format	Lenght	Description	Example
MsTxnId	Numeric	1..256	Merchant system transaction identifier – unique identifier of payment provided by merchant (variable symbol).	1234567890
Amount	#0.00	1..10,2	The amount of payment. Decimal separator is a dot. Decimal portion is represented by two digits.	1.00
CurrCode	AAA	3	Currency of payment ISO 4217	EUR
Result	OK/ FAIL/ PENDING/ AUTHORIZE D	2/4/7	OK- payment successful FAIL – payment failed PENDING – payment sent for processing	OK
Sign	Alpha-numeric	32	Checksum of transmitted parameters. Send only if request contains 'RedirectSign=true'.	21f22ef2af21d 3819cd0cff06 ef55943

Redirection example:

<http://mojobchod.sk/rurl?MsTxnId=1234567890&Amount=1.00&CurrCode=EUR&Result=OK>

3.4 Complete / Cancel Pre-authorized Payment

Payments can be completed or cancelled only when they are based as pre-authorized and are in a state AUTHORIZED.

The condition is the creation of HTTPS POST method request. The data will be coded as application/x-www-form-urlencoded.

https://admin.24-pay.eu/pay_gate/auth

The list of parameters is given in the table below:

Parameter	Mandatory	Format	Length	Description	Example
Mid	•	Alpha-numeric	8	Merchant identifier (case sensitive)	1a2B3c4D
EshopId	•	Numeric	1..10	E-shop identifier	135
MsTxnId	•	Alpha-numeric	1..32	Merchant system transaction identifier – unique identifier of payment provided by merchant (variable symbol).	1234567890
PspTxnId	•	Alpha-numeric	1..32	Transaction identifier – unique identifier of payment provided by °, sent in notification message after pre-authorization.	0987654321
Amount	•	#0.00	1..10,2	The amount of payment. Decimal separator is a dot. Decimal portion is represented by two digits. On completion of the preauthorization value must be equal to or less than the amount of preauthorization. In case of cancellation value must be equal to the amount of preauthorization.	1.00
CurrAlpha Code	•	AAA	3	Currency of payment ISO 4217	EUR

Timestamp	•	yyyy-MM-dd HH:mm:ss	19	Timestamp of payment request. Date and time separator is a space character!	2014-12-01 13:00:00
Target	•	OK/FAIL	2/4	OK - complete payment FAIL - cancel payment	OK
Sign	•	Alpha-numeric	32	Checksum of transmitted parameters	
LangCode		aa	2	ISO 639-1 language code. sk, cs, en, de, hu, es, fr, it, pl. Default sk..	sk
NURL		URL	256	Notification Return URL - URL address, where notifications about change of payment status are sent via HTTP/HTTPS POST method within the body request. It overlays configured item NURL if it is presented.	http://mojobchod.sk/nurl

Merchant obtain the following information in response JSON format:

```
{ "MsTxnId": "1234567890",
  "PspTxnId": "0987654321",
  "Amount": "1.00",
  "CurrCode": "EUR",
  "Target": "OK",
  "Status": "OK" }
```

This response acknowledges acceptance for processing. Acknowledgment of the transaction status change is sent to NURL in notification message (section 4.2), however, only if Status is OK or FAIL, in case ERROR no notification message is sent, since there is no change in payment status.

3.5 SIGN

For each request for payment from merchant and notification of the payment processing status by 24pay is created checksum. Through the checksum you can verify the integrity and authenticity of data.

The accuracy of generated signature is possible to verify in the 24pay interface https://admin.24-pay.eu/sup_gui/pages/PayReqSimulation.jsf.

3.5.1 Security key

Security key is generated for each merchant - key length 32B (B 32 = 256 bits). Merchant gets the **key** as a string representing the hexadecimal notation - string of 64 characters.

Initialization vector **IV** is needed to calculate checksum besides security key. The initialization vector is formed by concatenating the parameter **Mid** with its reverse form. In this way, the resulting sequence of 16 characters represents the initialization vector **IV**.

3.5.2 Checksum

During the communication there is created checksum, respectively security signature as follows:

- a) Concatenation of signature protected parameters in the specific order creates MESSAGE, the content of which will be subject to encryption.
- b) Created string is converted to HASH/MD (message digest) of fixed length (20 B = 160 bits) using the SHA1 hash function.
- c) The resulting "fingerprint" MD is then encrypted using the AES¹ algorithm:
 - a. initialization vector **IV**
 - b. defined security **key**
- d) The output is a security signature length 32 B = 256 bits. First 16 B of signature is converted to a string equivalent to hexadecimal notation of this section signature. The original plaintext MD is in this way transformed into cipher text representing security signature of length 32 characters.

3.5.3 Request from Merchant

Merchant sends security signature in communication as the parameter value **SIGN**.

The subject of chaining are the following parameters:

MESSAGE = Mid ⊕ Amount ⊕ CurrencyAlphaCode ⊕ MsTxnId ⊕ FirstName ⊕ FamilyName ⊕ Timestamp
--

3.5.4 Notification of Payment Processing Status from 24pay

Merchant forms security signature from parameters of notification in the same way and compares it to the received parameter value **SIGN**.

The subject of chaining are the following parameters:

¹ Block symmetric cryptographic algorithm; key-size 256bits; block-size 128 bits; mode AES/CBC/PKCS7Padding.

MESSAGE =
Mid ⊕ Amount ⊕ Currency ⊕ PspTxnId ⊕ MsTxnId ⊕ Timestamp ⊕ Result

3.5.5 Redirect Client to Merchant

Send only if request contains parameter 'RedirectSign=true'

Merchant forms security signature from parameters of redirection in the same way and compares it to the received parameter value SIGN.

The subject of chaining are the following parameters:

MESSAGE =
MsTxnId ⊕ Amount ⊕ CurrCode ⊕ Result

3.5.6 Complete / Cancel Pre-authorized Payment

Merchant sends security signature in communication as the parameter value **SIGN**.

The subject of chaining are the following parameters:

MESSAGE =
Mid ⊕ Amount ⊕ CurrencyAlphaCode ⊕ MsTxnId ⊕ PspTxnId ⊕ Target ⊕ Timestamp

4 Attachments

4.1 Example of Sign Creation

4.1.1 Payment Request

Key	123456781234567812345678123456781234567812345678123456781234567812345678
IV	{0x58, 0x32, 0x34, 0x35, 0x6e, 0x53, 0x4f, 0x33, 0x33, 0x4f, 0x53, 0x6e, 0x35, 0x34, 0x32, 0x58}
Mid	DemoOMED
Amount	1.00
CurrencyAlphaCode	EUR
MsTxnId	1234567890
FirstName	Jožko
FamilyName	Mrkvička
Timestamp	2014-12-01 13:00:00
Sign	2b817107edb88129d9aa8316f8758270

4.4.1	hexKey 1234567812345678123456781234567812345678123456781234567812345678	=
	length 64 characters	
4.4.1	byte[] keyBytes = {0x12, 0x34, 0x56, 0x78, 0x12, 0x34, 0x56, 0x78, , 0x34, 0x56, 0x78}	
	length 32B = 256 bits	
4.4.1	txtIV = DemoOMEDDEMOomeD	
	length 16 characters	
4.4.1	byte[] IV= {0x44, 0x65, 0x6D, 0x6F, 0x4F, 0x4D, 0x45, 0x44, 0x44, 0x45, 0x4D, 0x4F, 0x6F, 0x6D, 0x65, 0x44}	
	length 16B = 128 bits	
4.4.2 a	MESSAGE = DemoOMED1.00EUR1234567890JožkoMrkvička2014-12-01 13:00:00	
4.4.2 b	byte[] hash/md = SHA-1(message) = {0x78, 0xF7, 0xDA, 0x5C, 0x9D, 0x06, 0xEB, 0x02, 0x5A, 0x55, 0x7D, 0xBA, 0xB9, 0x41, 0x31, 0x83, 0x32, 0xA7, 0x2F, 0xB1}	
	length 20B = 160bits	
4.4.2 c	byte[] signBytes = {0x2B, 0x81, 0x71, 0x07, 0xED, 0xB8, 0x81, 0x29, 0xD9, 0xAA, 0x83, 0x16, 0xF8, 0x75, 0x82, 0x70, 0x31, 0x71, 0x5D, 0xAF, 0x1F, 0x70, 0xB6, 0x7A, 0x6F, 0x92, 0x0A, 0xF7, 0xB7, 0x19, 0x13, 0x72}	
	length 32B = 256 bits	
4.4.2 d	sign = 2b817107edb88129d9aa8316f8758270	

4.1.2 Notification of Payment Processing

Key	123456781234567812345678123456781234567812345678123456781234567812345678
IV	{0x58, 0x32, 0x34, 0x35, 0x6e, 0x53, 0x4f, 0x33, 0x33, 0x4f, 0x53, 0x6e, 0x35, 0x34, 0x32, 0x58}
Mid	DemoOMED
Amount	1.00
Currency	EUR
PspTxnId	0987654321
MsTxnId	1234567890
Timestamp	2014-12-01 13:01:00
Result	OK
Sign	21f22ef2af21d3819cd0cff06ef55943

4.1.3 Complete Pre-authorized Payment

4.4.1	hexKey 1234567812345678123456781234567812345678123456781234567812345678	=
	length 64 characters	
4.4.1	byte[] keyBytes = {0x12, 0x34, 0x56, 0x78, 0x12, 0x34, 0x56, 0x78,, 0x34, 0x56, 0x78}	
	length 32B = 256 bits	
4.4.1	txtIV = DemoOMEDDEMOomeD	
	length 16 characters	
4.4.1	byte[] IV= {0X44, 0X65, 0X6D, 0X6F, 0X4F, 0X4D, 0X45, 0X44, 0X44, 0X45, 0X4D, 0X4F, 0X6F, 0X6D, 0X65, 0X44}	
	length 16B = 128 bits	
4.4.2 a	message = DemoOMED1.00EUR098765432112345678902014-12-01 13:00:00OK	
4.4.2 b	byte[] hash/md = SHA-1(message) = {0XC4, 0X77, 0X06, 0X33, 0X7F, 0X91, 0XAB, 0X96, 0XEE, 0X20, 0X6A, 0XEA, 0X35, 0XFD, 0X2A, 0X8E, 0X74, 0X57, 0XED, 0XBF}	
	length 20B = 160bits	
4.4.2 c	byte[] signBytes = {0X21, 0XF2, 0X2E, 0XF2, 0XAF, 0X21, 0XD3, 0X81, 0X9C, 0XD0, 0XCF, 0XF0, 0X6E, 0XF5, 0X59, 0X43, 0X57, 0X67, 0X14, 0XC1, 0XB0, 0XD1, 0X95, 0X67, 0X99, 0X12, 0XF9, 0XDE, 0X38, 0X72, 0X38, 0XCE}	
	length 32B = 256 bits	
4.4.2 d	sign = 21f22ef2af21d3819cd0cff06ef55943	

4.1.4 Prípád dokončenia predautorizovanej platby

Key	123456781234567812345678123456781234567812345678123456781234567812345678
IV	{0x58, 0x32, 0x34, 0x35, 0x6e, 0x53, 0x4f, 0x33, 0x33, 0x4f, 0x53, 0x6e, 0x35, 0x34, 0x32, 0x58}
Mid	DemoOMED
Amount	1.00
CurrencyAlphaCode	EUR
MsTxnId	1234567890
PspTxnId	0987654321
Target	OK
Timestamp	2014-12-01 13:00:00
Sign	34087afa7367d29507f2d3561bd63171

4.4.1	hexKey 123456781234567812345678123456781234567812345678123456781234567812345678	=
	length 64 characters	
4.4.1	byte[] keyBytes = {0x12, 0x34, 0x56, 0x78, 0x12, 0x34, 0x56, 0x78, , 0x34, 0x56, 0x78}	
	length 32B = 256 bits	
4.4.1	txtIV = DemoOMEDDEMOomeD	
	length 16 characters	
4.4.1	byte[] IV= {0x44, 0x65, 0x6D, 0x6F, 0x4F, 0x4D, 0x45, 0x44, 0x44, 0x45, 0x4D, 0x4F, 0x6F, 0x6D, 0x65, 0x44}	
	length 16B = 128 bits	
4.4.2 a	MESSAGE = DemoOMED1.00EUR12345678900987654321OK2014-12-01 13:00:00	
4.4.2 b	byte[] hash/md = SHA-1(message) = {0XDF, 0XBE, 0X53, 0X2A, 0X00, 0XA8, 0XA9, 0X44, 0XAF, 0X9F, 0XA4, 0X49, 0XE1, 0X7D, 0X25, 0X4B, 0X39, 0X9D, 0X05, 0X7C}	
	length 20B = 160bits	
4.4.2 c	byte[] signBytes = {0X34, 0X08, 0X7A, 0XFA, 0X73, 0X67, 0XD2, 0X95, 0X07, 0XF2, 0XD3, 0X56, 0X1B, 0XD6, 0X31, 0X71, 0X19, 0X20, 0X8A, 0X93, 0XB7, 0XE0, 0X09, 0X89, 0X5D, 0X87, 0XF8, 0XCB, 0XDE, 0X28, 0XF6, 0X86}	
	length 32B = 256 bits	
4.4.2 d	sign = 34087afa7367d29507f2d3561bd63171	

4.1.5 Code Samples

a) PHP

```
public function computeSIGN($mid, $key, $message){
    $hash = hash("sha1", $message, true);
    $iv = $mid . strrev($mid);
    $key = pack('H*', $key);
    $crypted = openssl_encrypt( $hash, 'AES-256-CBC', $key, 1, $iv );
    $sign = strtoupper(bin2hex(substr($crypted, 0, 16)));
    return $sign;
}
```

b) Java

```
public String generateSign(String message, String key, String iv) {
    try {
        Security.addProvider(new BouncyCastleProvider());
        byte[] keyBytes = Hex.decodeHex(key.toCharArray());
        byte[] ivBytes = iv.getBytes();

        SecretKeySpec secretKeySpec = new SecretKeySpec(keyBytes, "AES");
        IvParameterSpec ivSpec = new IvParameterSpec(ivBytes);
        Cipher encryptCipher = Cipher.getInstance("AES/CBC/PKCS7Padding");
        encryptCipher.init(Cipher.ENCRYPT_MODE, secretKeySpec, ivSpec);

        byte[] sha1Hash = DigestUtils.sha1(message);
        byte[] encryptedData = encryptCipher.doFinal(sha1Hash);
        return Hex.encodeHexString(encryptedData).substring(0,32);
    } catch (Exception e) {
        Logger.error("ERROR!", e);
        return null;
    }
}
```

c) .NET framework 3.5 (C#)

```
public static string AesEncrypt( string message, byte[] Key, byte[] IV, PaddingMode
paddingMode , CipherMode cipherMode)
{
    byte[] hash = GetSha1(message);
    AesManaged aes= new AesManaged();
    aes.Key = Key;
    aes.IV = IV;
    aes.Mode = cipherMode;
    aes.Padding = paddingMode;
    ICryptoTransform encryptor = aes.CreateEncryptor(aes.Key, aes.IV);

    byte[] encrypted = null;

    using (MemoryStream ms = new MemoryStream()) {
        using (var cs = new CryptoStream(ms, encryptor, CryptoStreamMode.Write))
        {
            cs.Write(hash, 0, hash.Length);
        }
        encrypted = ms.ToArray();
    }

    return ConvertByteArrayToHexString(encrypted);
}
```

d) .NET framework 3.5 (VB)

```
Public Shared Function AesEncrypt(message As String, Key As Byte(), IV As Byte(),
paddingMode As PaddingMode, cipherMode As CipherMode) As String
    Dim hash As Byte() = GetSha1(message)

    Dim aes As New AesManaged()
    aes.Key = Key
    aes.IV = IV
    aes.Mode = cipherMode
    aes.Padding = paddingMode

    Dim encryptor As ICryptoTransform = aes.CreateEncryptor(aes.Key, aes.IV)
    Dim encrypted As Byte() = Nothing

    Using ms As New MemoryStream()
        Using cs = New CryptoStream(ms, encryptor, CryptoStreamMode.Write)
            cs.Write(hash, 0, hash.Length)
        End Using
        encrypted = ms.ToArray()
    End Using

    Return ConvertByteArrayToHexString(encrypted)
End Function
```

4.2 Payment form

The screenshot displays the 24pay payment interface. At the top right, there is a 'Language' dropdown menu. The main payment area is divided into three sections: 'Card payment' with logos for Mastercard and VISA; 'Instant payment' with logos for Tatra Pay, sporopay, VUB PLATBY, poštová banka, VIAMO, and UniPlatba; and 'Bank transfer' with a 'PAY by square' logo. To the right of the payment methods, a white box contains the following information: Amount: 123,46 €, Order number: 598426, and Merchant: Demo obchod www.demoobchod.sk. The 24pay logo is centered at the bottom of the interface.