

Mobile SDK Specifications for Android
(Version 4.1.1)
Last updated Dec 15, 2014

MOLPay Sdn Bhd (948015-X)
B-13-3A, Jalan Multimedia 7/AH,
CityPark, i-City, 40000 Shah Alam,
Selangor Darul Ehsan, Malaysia.



+603-55218438



+603-55218437



sales@molpay.com



www.molpay.com

Table of Contents

1. PREREQUISITE	2
2. SYSTEM REQUIREMENTS	2
3. MOLPAY ANDROID PAYMENT LIBRARY	2
3.1 How the Library Works	3
3.2 Declaring Permissions in AndroidManifest.xml	3
3.3 Adding the Library and Importing Classes	3
3.4 Listing Payment Channels	3
3.5 Initiate the payment (sending payment request)	4
3.6 Payment Result (Auto-return Payment Status)	5
3.7 Payment Result (Manual Payment Status Request)	6
3.8 Transaction Details Using OrderId.....	7
3.9 MOLPay SDK Screenshots.....	9

1. Prerequisite

Please provide below information to verify merchant identity from MOLPay Server in order for merchant's app to use the payment channel:

- merchant ID
- verify key
- appname
- username
- password

All the values above are in one word without whitespace.

The information should be submitted to MOLPay support team and the email address is support@molpay.com

2. System Requirements

- Minimum required android OS version: 2.3
- Internet Connection is essential for the MOLPay SDK to successfully connected with the MOLPay API

3. MOLPay Android Payment Library

This section explains details about the library API, and it explains the instructions and example to be follow to develop an Android apps that using mobile payment channel.

The information should be submitted to MOLPay support team and the email address is support@molpay.com

3.1 How the Library Works

The flow of the library is as below:

1. Library is in JAR format;
2. Initiate the library;
3. Merchant app to pass buyer's information to the SDK library and will be sent to MOLPay server;
4. After payment is completed, MOLPay will return the result for transaction id and payment status;
5. Library will return the result back to application.

3.2 Declaring Permissions and Activity in AndroidManifest.xml

To use the payment channel, Internet Permission is required for the application to avoid any error during payment make.

3.2.1 Declaring Permission

```
<uses-permission android:name="android.permission.INTERNET"/>
```

3.2.1 Declaring Activity

```
<activity  
    android:name="com.molpay.molpaylib.MOLPayActivity"  
    android:configChanges="keyboardHidden|orientation|screenSize">  
</activity>
```

3.3 Adding the Library, Files and Importing Classes

Assuming that user is using the eclipse as development tools for the android application, the steps will be as follow :

1. Create folder name's libs and copy the molpay.jar into the folder.
2. Right click on your project and select "Properties"
3. Select "Java Build Path"
4. Select the "Libraries" tab
5. Select the "Add External Jars..." button
6. Choose the "molpay.jar", "gson.jar", "universal-image-loader-1.9.2.jar" and / or "android-support-v13" files from MOLPay SDK folder and click "open"
7. Select "Order and Export" and tick all newly add jars and click "ok"
8. Copy res folder and paste in the project res folder
9. Copy all layout in the layout folder and paste in the project layout folder

*Please be careful when does copy and paste steps so that your project files not be replaced by MOLPay sdk files.

*If any errors occur, try clean the project.

You are required to import appropriate classes into your application classes as example show below:

```
import com.molpay.molpaylib.MOLPayActivity;  
import com.molpay.molpaylib.settings.MerchantInfo;
```

3.4 Listing Payment Channels

There are 2 ways for buyer to select the payment channel. First is to list out all available channels in this SDK. Second is to pass a specific channel to the SDK.

3.4.1 Listing all the channels

```
bundle.putString("Channel", "multi"); Type: String
```

3.4.2 Specific channel

```
bundle.putString("Channel", "MB2u"); Type: String
```

* **Note:** Please refer to last page of this document for the full lists of “Channel” value.

3.5 Initiate the Payment Activity

MOLPayActivity is to start payment activity. For example:

```
public static final int REQUEST_CODE=1; //Any number

Intent intent = new Intent(this, MOLPayActivity.class);
startActivityForResult(intent, REQUEST_CODE);
```

To pass the payment information to the activity, please refer to following example:

```
Intent intent = new Intent(this, MOLPayActivity.class);
Bundle bundle = new Bundle();

//Merchant Info
bundle.putString("MerchantId", merchantId) Type: String
bundle.putString("AppName", appName); Type: String
bundle.putString("VerifyKey", verifykey); Type: String
bundle.putString("Username", username); Type: String
bundle.putString("Password", password); Type: String

//Buyer Info
bundle.putString("OrderId", orderid ); Type: String
bundle.putString("BillName", bill_Name); Type: String
bundle.putString("BillDesc", bill_Desc); Type: String
bundle.putString("BillMobile", bill_Mobile); Type: String
bundle.putString("BillEmail", bill_Email); Type: String
bundle.putString("Channel", Channel); Type: String
bundle.putString("Currency", Currency); Type: String
bundle.putString("Country", Country); Type: String
bundle.putFloat("Amount", Amount); Type: Float
bundle.putBoolean("debug", false); Type: Boolean //Make this true to
enable debugging
bundle.putBoolean("editable", false); Type: Boolean //Make this true to
make the sdk fields editable
intent.putExtras(bundle);
startActivityForResult(intent, REQUEST_CODE);
```

* **Note:** Please refer to MOLPay API for the full lists of “Channel” value.

To Start the MOLPay SDK as the example below:

```
Intent intent = new Intent(this, MOLPayActivity.class);
Bundle bundle = new Bundle();
```

```
intent.putExtras(bundle);  
startActivityForResult(intent, REQUEST_CODE);
```

- **Note:** Hitting the back button while MOLPay SDK is running will cause the payment page to exit after a warning dialog box

3.6 Payment Result (Auto-return Payment Status)

You can either implement auto-return payment result or manual request for the payment result.

To receive the result automatically from library, you need to call `onActivityResult`.

```
public class MainActivity extends FragmentActivity implements  
@Override  
protected void onActivityResult(int requestCode, int resultCode, Intent  
data) {  
    if (resultCode == RESULT_OK && requestCode == REQUEST_CODE) {  
        Bundle bundle = data.getExtras().getBundle("bundle"); String  
        amount = bundle.getString(MerchantInfo.PAY_AMOUNT); String  
        transaction_id = bundle.getString(MerchantInfo.TXN_ID);  
        String transaction_status = bundle.getString(MerchantInfo.STATUS_CODE);  
        String error_desc = bundle.getString(MerchantInfo.ERR_DESC);  
        String currency = bundle.getString(MerchantInfo.CURRENCY);  
        String orderId = bundle.getString(MerchantInfo.ORDER_ID);  
        String payDate = bundle.getString(MerchantInfo.PAYDATE);  
        String bill_name = bundle.getString(MerchantInfo.BILL_NAME);  
        String bill_email = bundle.getString(MerchantInfo.BILL_EMAIL);  
        String bill_mobile = bundle.getString(MerchantInfo.BILL_MOBILE);  
        String bill_desc = bundle.getString(MerchantInfo.BILL_DESC);  
  
        //transaction_status = 00 (SUCCESS), 11(FAIL) & 22(PENDING)  
    }  
}
```

3.7 Payment Result (Manual Payment Status Request)

To use manual approach, developer needs to pass the request status information to the fragment, please refer to following example:

```
String merchant_id = MerchandId (String);
String txn_id = TransactionId (String);
float amount = Amount (float);
String verifyKey = VerifyKey (String);

String [] request = {merchant_id, txn_id, ""+amount, verifyKey};
new RequestResult().execute(request);

private class RequestResult extends AsyncTask<String, Void, Boolean> {

    String HTMLResult;

    @Override
    protected Boolean doInBackground(String... param) {
        String merchant_id = ""+param[0];
        String txn_id = ""+param[1];
        float amount = Float.parseFloat(param[2]);
        String verifyKey = ""+param[3];

        HTMLResult = JSONNString.requestResult(merchant_id, txn_id, amount,
        verifyKey);
    }
}
```



```

return true;
}

@Override
protected void onPostExecute(Boolean result) {
    super.onPostExecute(result);
    if (result)
    {
        try {
            JSONObject jsonObj = new JSONObject(HTMLResult);
            String transactionID = jsonObj.getString("txn_ID");
            String amount = jsonObj.getString("amount");
            String status_code = jsonObj.getString("status_code");
            String app_code = jsonObj.getString("app_code");
            String pay_date = jsonObj.getString("paydate");

        }
        catch (JSONException e) {
            e.printStackTrace();
        }
    }
}
}
}

```

If multiple or duplicated order ID found in MOLPay system, the latest transaction status will be returned.

3.8 Transaction Details Using OrderId (Manual Payment Status Request)

The SDK allows grabbing the transaction details using the orderId after the transaction already created, please refer to following example:

```

String merchant_id = MerchandId (String);
String order_id = OrderId (String);
float amount = Amount (float);
String verifyKey = VerifyKey (String);

```

```

String [] request = {merchant_id, order_id, String.valueOf (amount),
verifyKey};
new RequestResult().execute(request);

```

```

private class RequestResult extends AsyncTask<String, Void, Boolean> {

    String HTMLResult;

    @Override
    protected Boolean doInBackground(String... param) {

        String merchant_id = ""+param[0];
        String order_id = ""+param[1];
        float amount = Float.parseFloat(param[2]);
        String verifyKey = ""+param[3];
    }
}

```

```
HTMLResult = JSONnString.GetTxnIdManually(merchant_id, order_id,  
amount, verifyKey);
```



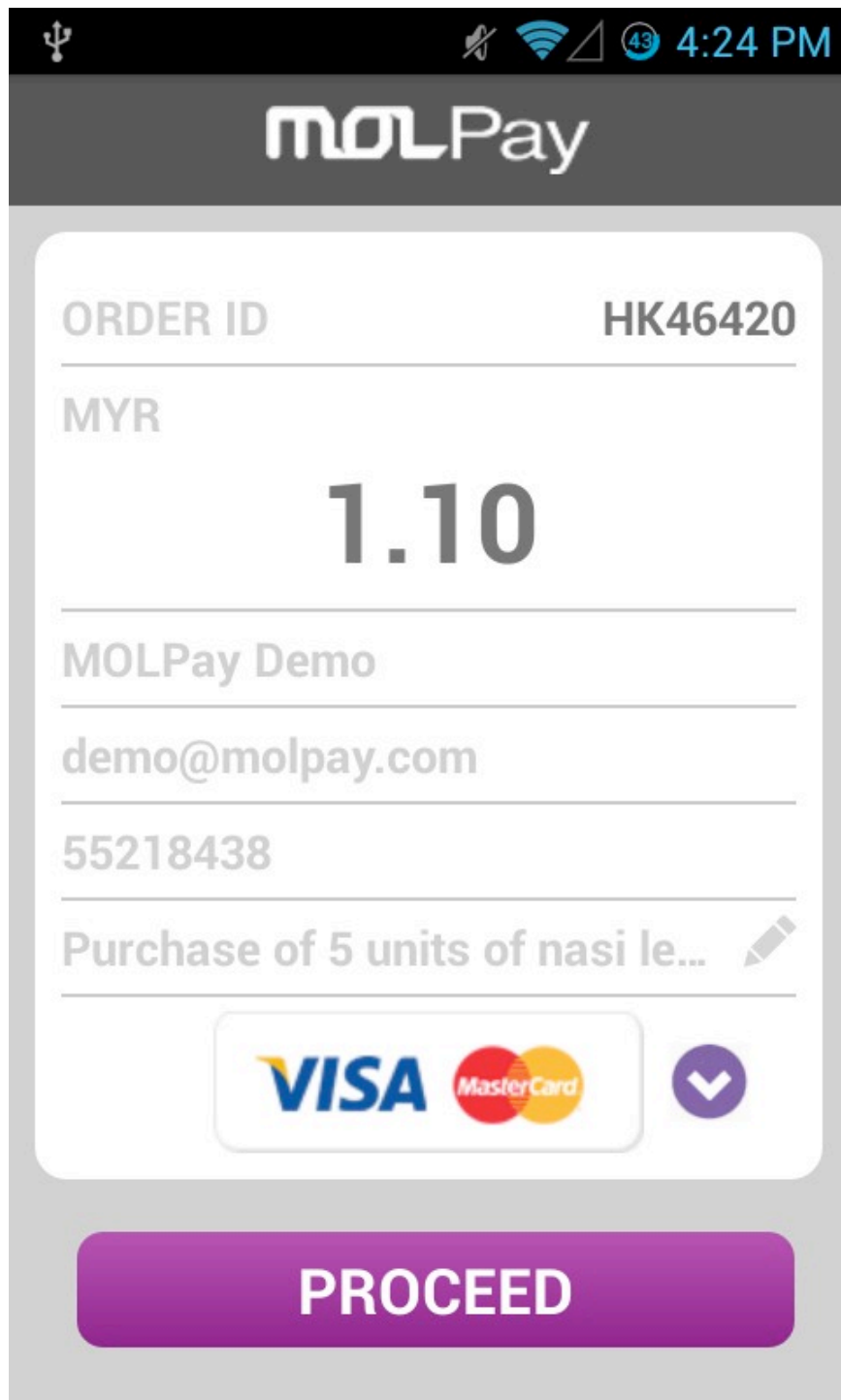
```
return true;
}

@Override
protected void onPostExecute(Boolean result) {
super.onPostExecute(result);
    if (result)
    {
        try {
            JSONObject jsonObj = new JSONObject(HTMLResult);
            String transactionID = jsonObj.getString("txn_ID");
            String amount = jsonObj.getString("amount");
            String status_code = jsonObj.getString("status_code");
            String app_code = jsonObj.getString("app_code");
            String pay_date = jsonObj.getString("paydate");

        }
        catch (JSONException e) {
            e.printStackTrace();
        }
    }
}
```

3.9 MOLPay SDK Screenshots

Order details



A screenshot of the MOLPay SDK interface on a mobile device. The screen displays order details for a transaction. At the top, the status bar shows a USB icon, signal strength, Wi-Fi, and a battery level of 43% at 4:24 PM. The MOLPay logo is prominently displayed at the top of the app interface. Below the logo, the order details are presented in a form-like structure with horizontal dividers. The fields include: ORDER ID (HK46420), MYR (currency), a large amount of 1.10, the merchant name MOLPay Demo, the email address demo@molpay.com, a reference number 55218438, and a description of the purchase: 'Purchase of 5 units of nasi le...'. Below the description, there are logos for VISA and MasterCard, and a dropdown arrow icon. At the bottom of the screen, there is a large purple button labeled 'PROCEED'.

ORDER ID HK46420

MYR

1.10

MOLPay Demo

demo@molpay.com

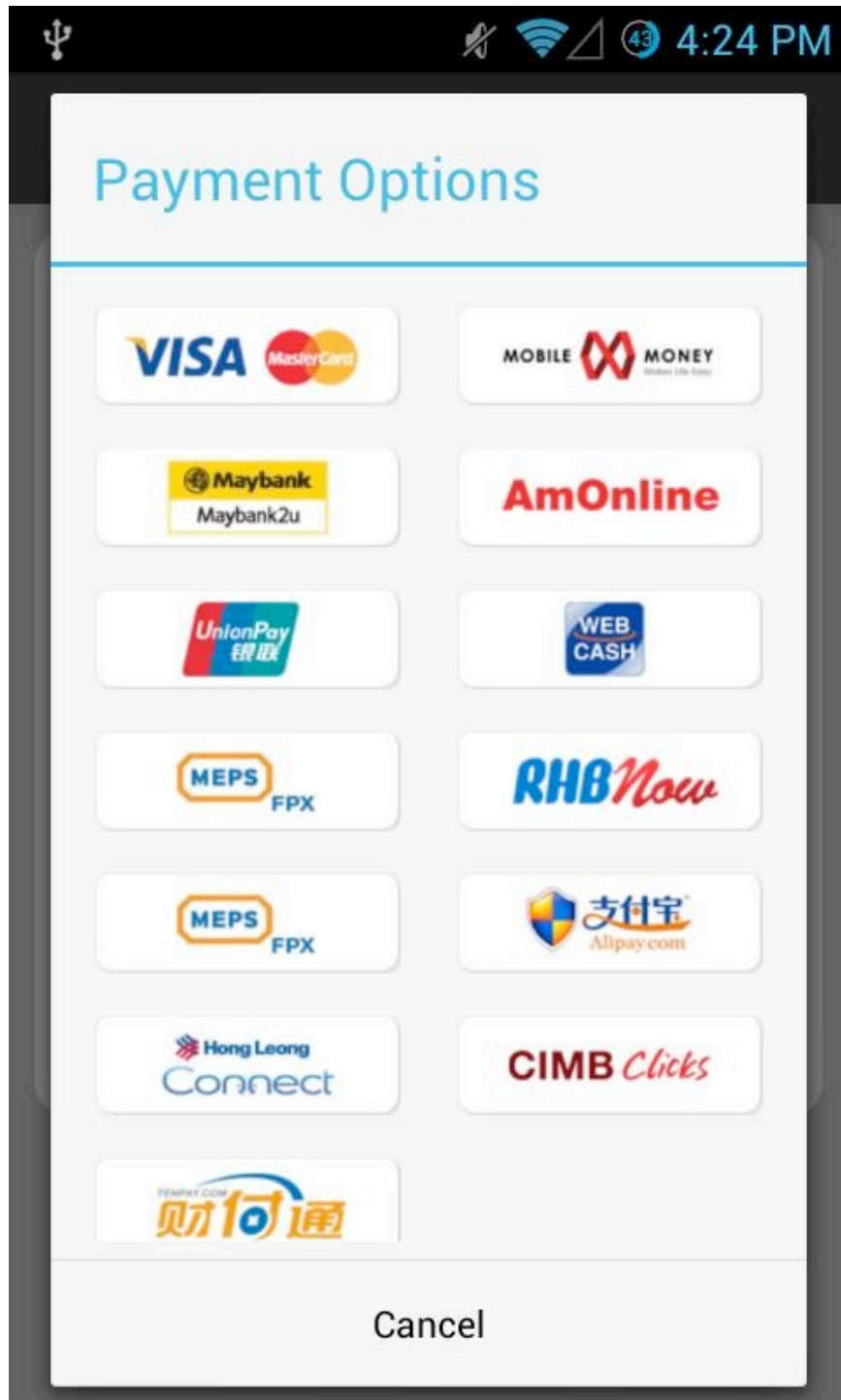
55218438

Purchase of 5 units of nasi le...

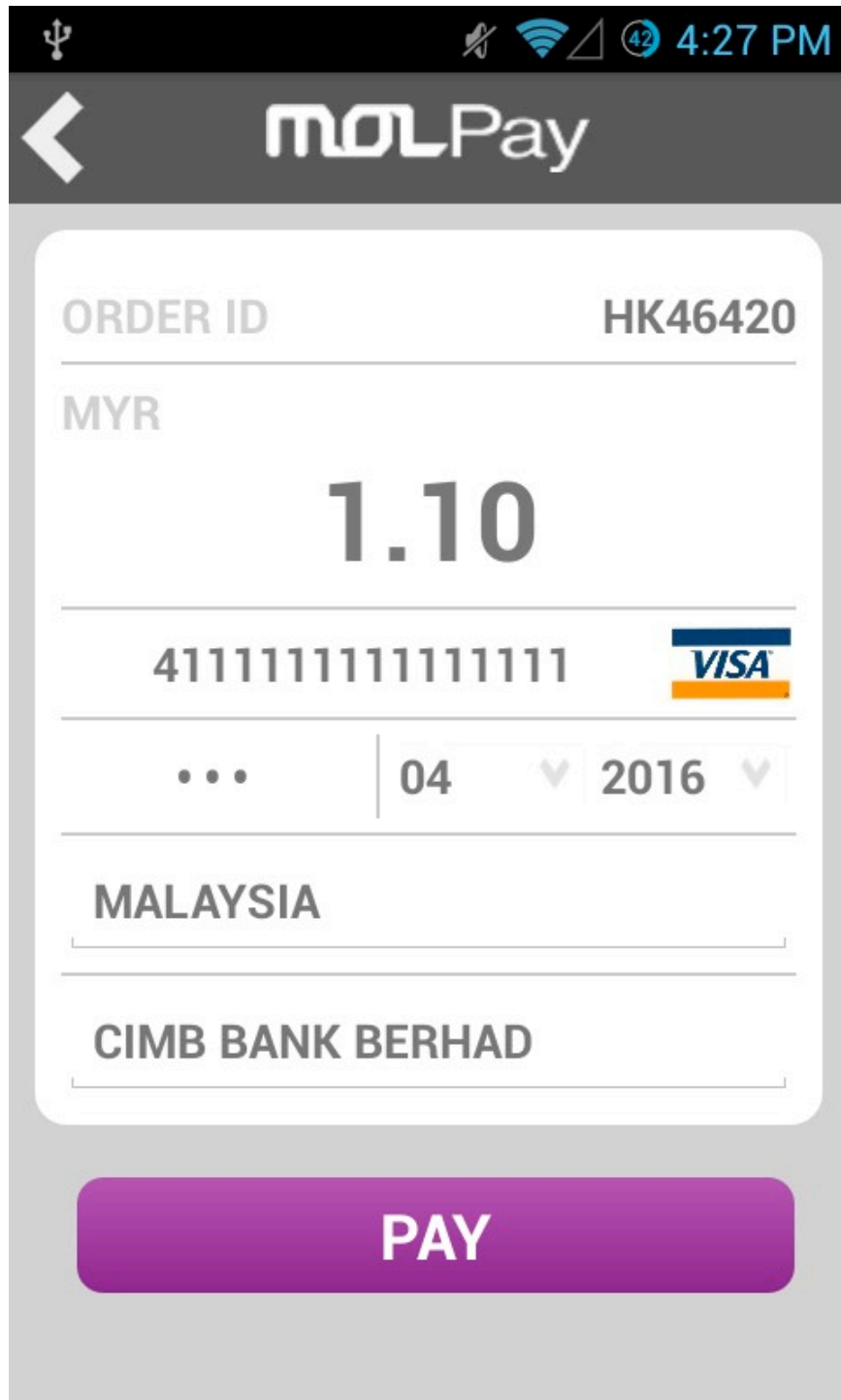
VISA MasterCard

PROCEED

List of Channels



Credit / Debit Card




The image shows a mobile app interface for MOLPay. At the top, there's a status bar with icons for USB, signal, Wi-Fi, and battery, along with the time 4:27 PM. Below this is a dark header with a back arrow and the MOLPay logo. The main content area is a white card with rounded corners. It displays the ORDER ID HK46420, the currency MYR, and the amount 1.10. Below the amount is the card number 4111111111111111 and the VISA logo. The cardholder's name is masked with three dots, followed by the expiry date 04/2016. The country is MALAYSIA and the bank is CIMB BANK BERHAD. At the bottom of the card is a large purple button labeled PAY.

ORDER ID HK46420

MYR

1.10

4111111111111111 

... 04 2016

MALAYSIA

CIMB BANK BERHAD

PAY

Payment Page



Secure ePay Code has been sent to your registered mobile phone number **+6012xxx1111**. Please enter the Secure ePay Code to authenticate this payment.

Merchant Name : Netbder Netbuilder
Amount : MYR 1.10
Transaction Date : Fri Apr 11 2014 16:32:32 GMT+0800
CIMB Bank MasterCard No. : xxxx xxxx xxxx 1111
SecureCode™ Secure ePay Code :

Submit **Cancel**

If you do not receive Secure ePay Code within the next few minutes, please click on "Resend Secure ePay Code" button for a new secure ePay Code.

Resend Secure ePay Code

This information is **not shared** with the Merchant.

 Please contact our Customer Service Hotline at the back of your card for assistance.

List of Channels Code and Supported Currency Code

No	Channel Code	Channel Name	Currency Code
1.	credit	Visa / Mastercard	MYR
2.	mobilemoney	Mobile Money	MYR
3.	maybank2u	Maybank2u	MYR
4.	amb	AmOnline	MYR
5.	alb	Aliance Online	MYR
6.	rhb	RHB Now	MYR
7.	paymentasia	China Online Banking	MYR, RM, RMB and CNY
8.	webcash	WEBCASH	MYR
9.	paypal	PayPal	USD, AUD, GBP, CAD, CZK, DKK, EUR, HKD, HUF, ILS, JPY, MYR, MXN, NZD, NOK, PHP, PLN, SGD, SEK, CHF, TWD and THB
10.	alipay	Alipay	MYR, RM, USD, TWD and RMB
11.	hlb	Hong Leong Online	MYR
12.	cimb	CIMB Clicks	MYR
13.	fpx	MEPS FPX	MYR
14.	dragonpay	DragonPay	USD and PHP
15.	enetsD	eNets Debit	SGD
16.	polipayment	Poli Payment	MYR, RM and AUD
17.	tenpay	Tenpay	MYR, RM and USD
18.	fuiou	Fuiou	MYR, RM and USD
19.	cash	7-Eleven Cash Retail	MYR
20.	esapay	Esapay Cash Retail	MYR
21.	senheng	Senheng Cash Retail	MYR
22.	singpost	SingPost	SGD