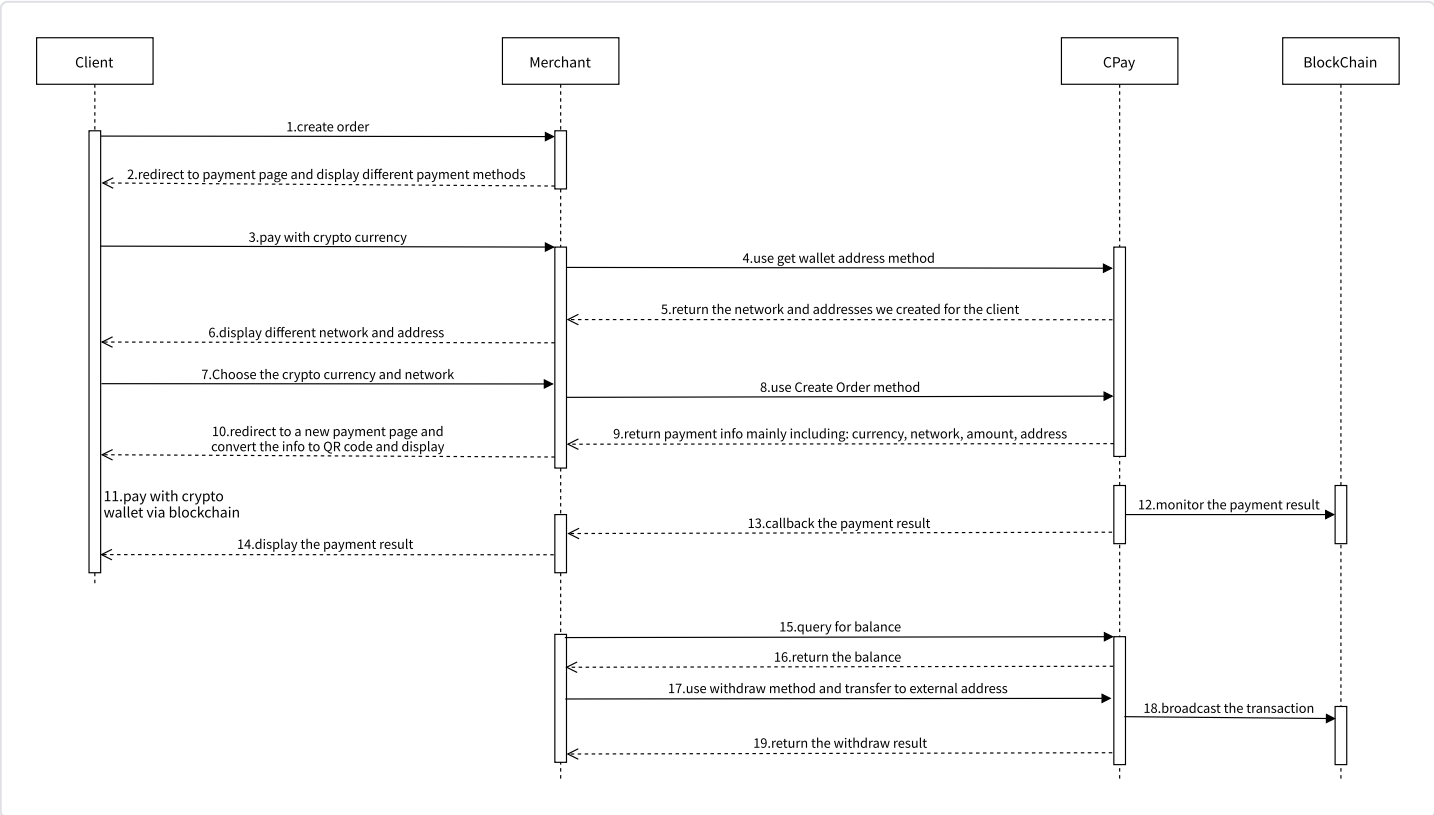
C.Pay Open API

C-Pay
Interface Description

1. Flowchart

UML of our main service: payment and settlement based on crypto currency.



2. Env config

domain test : http://8.142.157.45:9075/

domain prod: https://api.cpay.ltd/

3. Get WalletAddress

Description: query for the deposit address of your users.

Note: If the user want

to deposit crypto directly from his external wallet, your system needs to use this method ,so we can generate a new address for the client ,otherwise he can’ t deposit .

Request URL:
https://domain/openapi/v1/getWalletAddress

Request method: GET

- **Request parameters**

Name	Type	Mandatory	Description
merchantId	Long	Yes	The unique ID generated by C-Pay for merchant
userId	String(64)	Yes	User’ s unique ID in merchant’ s system
sign	String(32)	Yes	See API Signature

- **Request Example**

```
1 http://8.142.157.45:9075/openapi/v1/getWalletAddress?merchantId=20002299&userId=te
  st3450283&sign=8c4807355d1547630d3381ae331a9c6fa59d1fde6199133acb58545a94bcd79e
```

- **Response parameters**

Name	Type	Description
currency	String	Currency
address	String	Address
network	String	Network

- **Response example**

```
{
    "code": 0,
    "msg": "ok",
    "data": [
        {
            "currency": "BTC",
            "address": "12FhMVSJYw3NMsMbfHqhyJX6jAgFESoCrA",
            "network": "Bitcoin"
        },
        {
            "currency": "ETH",
            "address": "0x3a4776e3bd49273a5bce7f8aa77c51421cbb83a7",
            "network": "Ethereum(ERC-20)"
        },
        {
            "currency": "USDT",
            "address": "0x3a4776e3bd49273a5bce7f8aa77c51421cbb83a7",
            "network": "BSC(BEP-20)"
        },
        {
            "currency": "USDT",
            "address": "TBkF1gJsvGhJ68BKFvYJNywezKEinyiHXQ",
            "network": "TRON(TRC-20)"
        }
    ]
}
```


4. Withdraw

Description: Use this method when your clients want to withdraw crypto coins to their external wallet address.

Request URL:
https://domain/openapi/v1/withdraw

Request method: POST

- **Request parameters**

Name	Type	Mandatory	Description
merchantId	Long	Yes	The unique ID generated by C-Pay for merchant

userId	String(64)	Yes	User’ s unique ID in merchant’ s system
merchantTradeNo	String(64)	Yes	The unique order number generated by merchant
createTime	Long	Yes	The Order time (ms) generated by merchant
cryptoCurrency	String(16)	Yes	The currency user wants to withdraw. e.g. BTC, USDT, ETH
network	String(16)	Yes	Network e.g. Bitcoin, Ethereum(ERC-20), TRON(TRC-20), BSC(BEP-20)
amount	String(128)	Yes	The amount user wants to withdraw. Support 8 decimals, e.g. 66.12345678
toAddress	String(128)	Yes	The address user fills in to receive the withdraw
sign	String(32)	Yes	See API Signature

- **Request Example**

```
http://8.142.157.45:9075/openapi/v1/purchaseCryptoCurrency?merchantId=20000092&FiatCurrency=USD&cryptoCurrency=USDT&purchaseType=1&amount=100&sign=1d102274f2e98ad0bbfdf97c42110a748105e96182a2350a39f2d998443dabfb&merchantTradeNo=100000230&createTime=1653669353000&userId=2345
```

- **Response parameters**

Name	Type	Description
fee	String	Transaction fee amount. The fee currency is the same as withdrawing currency.

- **Response Example**

```
{
    "code": 0,
    "msg": "ok",
```

```
    "data": {
        "fee": "0.0123"
    }
}
```

5. Query ExchangeRate

Description: query the exchange rate for "Fiat to crypto" and "crypto to crypto"

Request URL: https://domain/openapi/v1/getExchangeRate

Request method: GET

- **Request parameters**

Name	Type	Mandatory	Description
merchantId	Long	Yes	The unique ID generated by C-Pay for merchant
sourceCurrency	String(16)	Yes	The currency user has
targetCurrency	String(16)	Yes	The currency user wants to get
purchaseType	int	Yes	0:by sourceCurrency amount 1:by targetCurrency amount
amount	String(128)	Yes	If purchaseType=0, it means “I want to spend XXX to buy crypto”; If purchaseType=1,it means “I want to get XXX crypto coin”
sign	String(32)	Yes	See API Signature

- **Request Example**

```
http://8.142.157.45:9075/openapi/v1/getExchangeRate?merchantId=20000092&sourceCurrency=USDT&targetCurrency=BTC&purchaseType=1&amount=10&sign=98c6d3376078e9d95d6386fd967530535af729ee178bafa1773ab34e5bed19e9
```

- **Response parameters**

Name	Type	Description
exchangeRate	String(128)	Real time exchange rate

- **Response Example**

```
{
    "code": 0,
    "msg": "ok",
    "data": {
        "exchangeRate": "2999.342"
    }
}
```

> Note：sourceCurrency:targetCurrency=1:exchangeRate
>
> e.g. :sourceCurrency is USDT, targetCurrency is BTC,
>
> exchangeRate=40000.000000

6. Query ExchangeRate For Credit Card

Description: query the exchange rate for "Fiat to crypto" and "crypto to crypto"

Request URL: https://domain/openapi/v1/getExchangeRateForCreditCard

Request method: GET

- **Request parameters**

Name	Type	Mandatory	Description
merchantId	Long	Yes	The unique ID generated by C-Pay for merchant

sourceCurrency	String(16)	Yes	The currency user has
targetCurrency	String(16)	Yes	The currency user wants to get
purchaseType	int	Yes	0:by sourceCurrency amount 1:by targetCurrency amount
amount	String(128)	Yes	If purchaseType=0, it means “I want to spend XXX to buy crypto”; If purchaseType=1,it means “I want to get XXX crypto coin”
sign	String(32)	Yes	See API Signature

- **Request Example**

```
http://8.142.157.45:9075/openapi/v1/getExchangeRateForCreditCard?merchantId=20000092&sourceCurrency=USD&targetCurrency=BTC&purchaseType=0&amount=200&sign=adc81a604c4cf1bdd9a2e1c7476ddcd91aa5b13303fe4f49769a6add3f62e926
```

- **Response parameters**

Name	Type	Description
exchangeRate	String(128)	Real time exchange rate
amountExchanged	String(128)	If the purchaseType=0, it means “when I spend Fiat, XXX crypto I will get” If the purchaseType=1, it means “if I want to get crypto ,how much Fiat I need to pay”

- **Response Example**

```
{
    "code": 0,
    "msg": "ok",
    "data": {
```

```
        "exchangeRate": "2999.342",
        "amountExchanged": "0.03334064604"
    }
}
```

> Note：sourceCurrency:targetCurrency=1:exchangeRate
>
> e.g. :sourceCurrency is USDT, targetCurrency is BTC,
>
> exchangeRate=40000.000000

7. Query Order Detail

Description: query for the card purchase order details

Request URL:
https://domain/openapi/v1/getOrderDetail

Request method: GET

- **Request parameters**

Name	Type	Mandatory	Description
merchantId	Long	Yes	The unique ID generated by C-Pay for merchant
merchantTradeNo	String(64)	Yes	
sign	String(32)	Yes	See API Signature

- **Request Example**

```
http://8.142.157.45:9075/openapi/v1/getOrderDetail?merchantId=20000092&merchantTradeNo=100000230&sign=1234
```

- **Response parameters**

Name	Type	Description
orderId	String	CPay orderId
status	int	0:PENDING 14:COMPLETED 15:CLOSED
depositMethod	String	VISA or MasterCard
sourceCurrencyAmount	String	
sourceCurrency	String	
targetCurrencyAmount	String	
targetCurrency	String	
merchantTradeNo	String	

- **Response Example**

```
{
    "code": 0,
    "msg": "ok",
    "data": {
        "orderId ": "3548135216138444487",
        "status": "10",
        "depositMethod ": "VISA",
        "sourceCurrencyAmount ": "1004.5",
        "sourceCurrency ": "USD",
        "targetCurrencyAmount ": "1000",
        "targetCurrency ": "USDT",
        "walletAddress ": "TP33EoRzymVVnv5Wd7pRS5vCqhhTKZLV3x",
        "merchantTradeNo ": "63fdf2ee-b2c3-4ca3-aefe-6272cd579e54"
    }
}
```

8. Exchange

Description: query for exchange between two crypto coins

Request URL:
https://domain/openapi/v1/exchange

Request method: POST

- **Request parameters**

Name	Type	Mandatory	Description
merchantId	Long	Yes	The unique ID generated by C-Pay for merchant
userId	String(64)	Yes	User’ s unique ID in merchant’ s system
merchantTradeNo	String(64)	Yes	The unique order number generated by merchant
createTime	Long	Yes	The Order time (ms) generated by merchant
sourceCurrency	String(16)	Yes	The currency user has
sourceAmount	String(128)	Yes	Amount of source currency
targetCurrency	String(16)	Yes	The currency user wants to get
sign	String(32)	Yes	See API Signature

- **Request example**

```
curl -X POST 'https://domain/openapi/v1/exchange' -d 'merchantId=200001111&userId=xxx-1001&merchantTradeNo=100&createTime=1655899200000&sourceCurrency=USDT&sourceAmount=9.12345678&targetCurrency=BTC&sign=xxxxxxxxxxxxxxxx'
```

- **Response parameters**

Name	Type	Description
exchangeRate	String	The actual exchange rate between source currency and target currency

targetAmount	String	The actual amount of target currency the user gets after the exchange.
targetCurrency	String	The currency user wants to get
fee	String	Transaction fee amount. The fee currency is the same as the target currency.

- **Response example**

```
{
    "code": 0,
    "msg": "ok",
    "data": {
        "targetCurrency": "BTC",
        "targetAmount": "0.997",
        "exchangeRate": "40000.000000",
        "fee": "0.003"
    }
}
```

9. Purchase Crypto Currency

Description: query for generate the order of Visa/MC credit/debit card purchase crypto coins

Request URL:
https://domain/openapi/v1/purchaseCryptoCurrency

Request method: POST

- **Request parameters**

Name	Type	Mandatory	Description
merchantId	Long	Yes	The unique ID generated by C-Pay for merchant
userId	String(64)	Yes	User’ s unique ID in merchant’ s system

merchantTradeNo	String(64)	Yes	The unique order number generated by merchant
createTime	Long	Yes	The Order time (ms) generated by merchant
fiatCurrency	String(16)	Yes	The currency user has
purchaseType	Int	Yes	0:by fiatCurrency amount 1:by cryptoCurrency amount
amount	String(128)	Yes	If purchaseType=0, it means “I want to spend XXX Fiat to buy crypto”; If purchaseType=1,it means “I want to get XXX crypto coin”
cryptoCurrency	String(16)	Yes	The currency user wants to get
sign	String(32)	Yes	See API Signature

- **Request example**

```
http://8.142.157.45:9075/openapi/v1/purchaseCryptoCurrency?merchantId=20000092&fiatCurrency=USD&cryptoCurrency=USDT&purchaseType=1&amount=100&sign=1d102274f2e98ad0bbfdf97c42110a748105e96182a2350a39f2d998443dabfb&merchantTradeNo=100000230&createTime=1653669353000&userId=2345
```

- **Response parameters**

Name	Type	Mandatory	Description
orderId	String(64)		C-Pay orderId
status	Int		0:PENDING 14:COMPLETED 15:CLOSED
sourceCurrencyAmount	Int		
sourceCurrency	String(64)		
targetCurrencyAmount	Int		

targetCurrency	String(64)		
merchantTradeNo	String(64)		
redirectUrl	String(512)		

- **Response example**

```
{
    "code": 0,
    "msg": "ok",
    "data": {
        "orderId ": "3548135216138444487",
        "status ": "10",
        " depositMethod": "",
        "sourceCurrencyAmount ": "1004.5",
        "sourceCurrency": "USD",
        "targetCurrencyAmount ": "1000",
        "targetCurrency": "USDT",
        "merchantTradeNo ": "63fdf2ee-b2c3-4ca3-aefe-6272cd579e54",
        "redirectUrl": "www.advcash.com"
    }
}
```

10. Create Order

Description:

Request URL: https://domain/openapi/v1/createOrder

Request method: POST

- **Request parameters**

Name	Type	Mandatory	Description

merchantId	String(64)	Yes	The unique ID generated by C.Pay for merchant
merchantTradeNo	String(64)	Yes	The unique order number generated by merchant
createTime	Long	Yes	The Order time (ms) generated by merchant
userId	String(64)	Yes	User’ s unique ID in merchant’ s system
amount	String(128)	Yes	
cryptoCurrency	String(16)	Yes	The currency user wants to get
sign	String(256)	Yes	See API Signature

- **Request example**

```
http://8.142.157.45:9075/openapi/v1/createOrder?merchantId=20000092&merchantTradeNo=test0001&userId=83453&cryptoCurrency=USDT&amount=100&createTime=1658387195&sign=9211e7139623be0acf6fd6b65a234200f5e397efb0d3f17b24732805ee860e12
```

- **Response parameters**

Name	Type	Mandatory	Description
orderId	String(64)		CPay orderId
serviceFee	String(64)		
订单总金额			
到账金额			

- **Response example**

```
{
    "code": 0,
```

```
    "msg": "ok",
    "data": {
        "orderId ": "35481352161384448 7"
    }
}
```

11. Query Merchant Balance

Description:

Request URL: https://domain/openapi/v1/getMerchantBalance

Request method: GET

- **Request parameters**

Name	Type	Mandatory	Description
merchantId	String(64)	Yes	The unique ID generated by C.Pay for merchant
sign	String(256)	Yes	See API Signature

- **Request example**

```
http://8.142.157.45:9075/openapi/v1/getMerchantBalance?merchantId=20000092&sign=5a
aae162fb87530f983e600fb076981f724ebdbaf3ca6e87aae62aa15b2000e1
```

- **Response parameters**

Name	Type	Mandatory	Description
availableBalance	String(64)		

freezeBalance	String(64)		
cryptoCurrency	String(128)		

- **Response example**

```
{
        "code": 0,
        "msg": "ok",
        "data": [{
                "availableBalance ": "100000",
                "freezeBalance ": "10",
                "cryptoCurrency ": "USDT"
        }, {
                "availableBalance ": "103.85",
                "freezeBalance ": "0",
                "cryptoCurrency ": "ETH"
        }]
}
```

12. API Signature

0、API Example:

```
curl https://domain/openapi/v1/getSth?xx=1001&yy=&aa=hello&sign=Vs23424SHW

curl -X POST 'https://domain/openapi/v1/updateSth' -d 'xx=1001&yy=&aa=hello&sign=V
s23424SHW'
```

1、All parameters will be sorted by parameter name and stitched into a character string, e.g.: `aa=hello&xx=1001`

2、Add the secret key we provide at the end of the string: `aa=hello&xx=1001` , you will get: `aa=hello&xx=1001&key=aaaaaaaaaaxxxxxx`

> - secret key will be given through email
> - Given the value of the yy parameter is an empty string, therefore it is ignored here.

3、Encrypt HmacSha256 the string generated in step 2, and convert it to lowercase. Then you obtain a value which is the sign parameter of the API.

> sign=HmacSha256(`aa=hello&xx=1001&key=aaaaaaaaaaxxxxxx`)
>
> sign=sign.ToLower()

Response Code

Code	Description
0	ok
1	fail

Java demo

```
1
2  package com.lianjing.controller;
3
4  import javax.crypto.Mac;
5  import javax.crypto.spec.SecretKeySpec;
6  import java.util.*;
7
8  public class CheckSign {
9      public static void main(String[] args) {
10         String key = "*iu6hufsi%^54fyt";
11         Map<String, Object> params = new HashMap<>();
12         params.put("amount","32");
13         params.put("merchantTradeNo","1018163312113640372232");
14         params.put("merchantId","20000092");
15         params.put("createTime","20220726100218");
16         params.put("fiatCurrency","USD");
17         params.put("userId","10181633");
18         params.put("cryptoCurrency","USDT");
19         params.put("purchaseType","1");
20         Collection keySet = params.keySet();
21         List list = new ArrayList(keySet);
22         Collections.sort(list);
23         StringBuilder sb = new StringBuilder();
```

```
        for (int i = 0; i < list.size(); i++) {
            sb.append(list.get(i) + "=" + params.get(list.get(i))+"&");
        }
        sb.append("key="+key);
        System.out.println(sb.toString());
        try {
            System.out.println(HMACSHA256(sb.toString(),key));
        } catch (Exception e) {
            throw new RuntimeException(e);
        }

    }
    public static String HMACSHA256(String data, String key) throws Exception {

        Mac sha256_HMAC = Mac.getInstance("HmacSHA256");

        SecretKeySpec secret_key = new SecretKeySpec(key.getBytes("UTF-8"), "Hmac

        sha256_HMAC.init(secret_key);

        byte[] array = sha256_HMAC.doFinal(data.getBytes("UTF-8"));

        StringBuilder sb = new StringBuilder();

        for (byte item : array) {

            sb.append(Integer.toHexString((item & 0xFF) | 0x100).substring(1, 3))

        }

        return sb.toString().toLowerCase();

    }
}
```