Paylot Integration Documentation

Release latest

Aug 17, 2021

Table of Contents

This is the official documentation for the integration of Paylot.

For integration of the available SDKs and libraries, check out the following.

Android SDK

Woocommerce Plugin

PHP SDK

Other SDKs and libaries coming soon. . .

CHAPTER 1

Introduction

This is the documentation for integration of Paylot.

NB: Before you can start integrating Paylot, you will need a Paylot account. Create a free account now if you haven't already done so: https://paylot.co/signup.

After that, you can proceed to add a merchant/business at https://paylot.co/businesses/create

Once you have added your business details, you would be required to add at least one payment processor to accept payments. This involves adding a currency you would be collecting (**BTC, ETH, LTC, BCH, BNB, BUSD**) and what you prefer to store it as (**NGN, USDT, USDC, PAX, BUSD**).

Once you are done. You are good to go.

CHAPTER 2

Integration

There are 2 ways to integrate Paylot into your web or mobile applications. These are as follows:

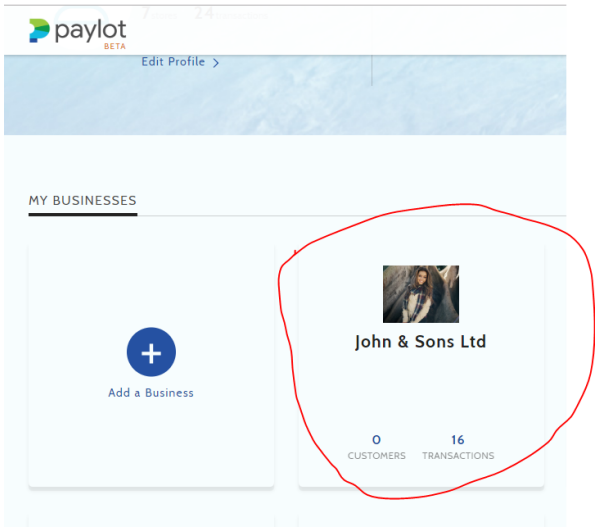
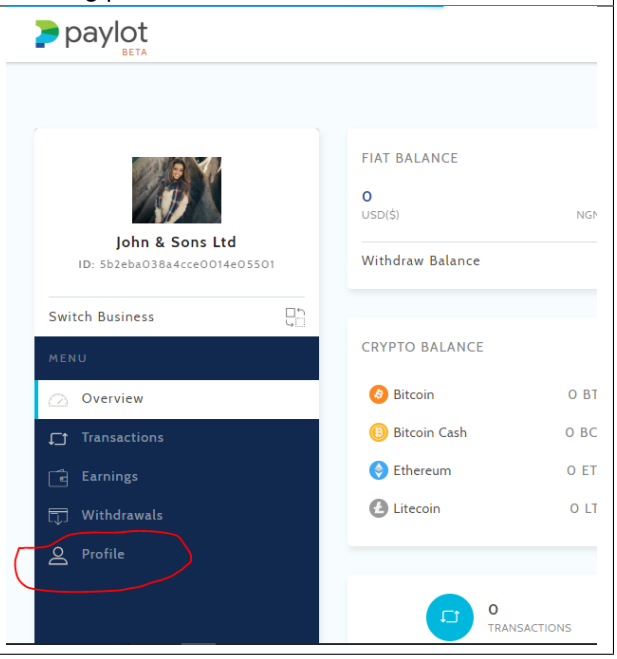
- **Paylot Inline:** This provides a Javascript library that you can include in any webpage to easily to use our payment widget with minimal code.
- **Paylot Standard:** This is necessary when you want a fully customized experience. It provides an API endpoint you can use to generate a wallet address, memo (BEP32 tokens) and the amount the customer is required to send.

2.1 1. Paylot Inline JS Client

Paylot inline javascript client offers a simple, secure and convenient payment flow for web and mobile. It can be integrated with a line of code thereby making it the easiest way to start accepting payments. It also makes it possible to start and end the payment flow on the same page, thus combating redirect fatigue.

Here is a sample code that calls Paylot and also handles outcome.

NB: Please, note that the key used is your merchant key. To get this key, go to your merchant profile by clicking one of your businesses on your dashboard @ https://paylot.co/dashboard and then clicking profile on the sidebar.

Clicking Business	Clicking profile
	

2.2 1.1. Integration Code

This HTML code shows a simple way to integrate Paylot into your webpage.

```
<form >
    <script src="https://js.paylot.co/v1/inline.min.js"></script>
    <button type="button" onclick="pay()"> Pay </button>
</form>

<script>
  function pay(){
    paylot({
        amount: 10000,
        key: 'pyt_pk-6efec0d34c8147eba4de783714c6eae7',
        reference: Date.now(),
        currency: 'NGN',
        payload: {
            type: 'payment',
            subject: 'Test payment',
            email: 'john.doe@gmail.com',
            sendMail: true
        },
        onClose: function(){
            console.log('I just closed the payment modal');
        }
    }, (err, tx) => {
        if(err){
            console.log('An error has occured');
        }else{
            //Transaction was successful
            console.log(tx);
```

(continues on next page)

(continued from previous page)

```
        }
    });
  }
</script>
```

2.3 1.2. Configuration options

(* indicates required)

Parameter	Description
amount *	The amount to be paid (number)
key *	The merchant public key (string)
reference *	A unique reference that identifies your transaction. If not found, a random reference would be generated (string)
currency*	The base currency (NGN, USD, BTC, ETH, LTC & BCH allowed) (string)
subaccount	The reference for the subaccount if available (for split payments).
payload.email	The email of the customer (string)
payload.type	The type of payment (string)
payload.sendMail	Determines whether an email should be sent to the user or not (boolean)
payload.callback	The callback URL where the webhook notification would be sent. (string)
onClose	Function called when popup is closed

2.4 1.3. Callback Parameters

The paylot function has the following signature.

```
function paylot(options, callback);
```

Options specifies the Configuration options as highlighted above while callback takes the form of normal javascript callbacks i.e. accepts a function with the following signature.

```
function callback(error, data);
```

Here, in the absence of errors, the data parameter will contain the transaction details and is an object with the following properties stated below.

Parameter	Description
reference	The transaction reference. Pay attention to this if you didn't create a reference manually. (string)
sent	Specifies if payment was made successfully (boolean)
confirmed	Specifies if the payment has been confirmed on the blockchain (boolean)
amount	Specifies the intended amount in the currency selected during payment (number)
amountSent	Specifies the actual amount that was sent to the blockchain (number)

NB: These are the same parameters posted to the call back url which can be set in the business profile.

2.5 2. Paylot Standard

This is necessary when you want a fully customized experience. We provides an API endpoint you can use to generate a wallet address, memo (BEP32 tokens) and the amount the customer is required to send. You are free to customize your interface the interface used to display the wallet address, amount and memo. To initialize the transaction, you are expected to use the following:

URL:

```
POST https://api.paylot.co/transactions/initialize
```

2.5.1 Request

The expected request is a JSON object of the format stated below.

```
{
  "currency": "BTC",
  "reference": "1234567",
  "key": "pyt_pk_12345678901234567890",
  "subaccount": "1111321",
  "email": "doz****@qa.team",
  "sendMail": true,
  "data": {
     "amount": 10000,
     "currency": "NGN"
   }
}
```

Property	Description
currency *	This specifies the currency you would like to accept options: (BTC,ETH,LTC,BNB, etc)
reference *	A unique reference for the transaction.
key *	This is the merchant's public key.
subaccount *	The reference for the subaccount if available (for split payments).
email *	This is the customer's email.
sendMail	This specifies if we should send a mail to the customer on successful transaction
data.amount *	This amount you want to charge
data.currency *	This ISO code for the currency you are charging in. eg. NGN, USD, etc

2.5.2 Response

The expected response is a JSON object of the format stated below.

```
{
     "amount": 0,
     "address": "string",
     "memo": "string",
     "currency": "string",
     "reference": "string"
 }
```

Property	Description
currency *	This specifies the currency you would like to accept options: (BTC,ETH,LTC,BNB, etc)
reference *	A unique reference for the transaction.
amount *	This is the amount to charge customers (fee inclusive).
address *	This is the generated wallet address.
memo (BEP32 only)	This is the generated wallet. memo (BEP32 currencies only).

NB: It is recommended that all wallet addresses are valid for a maximum of 15 minutes. Due to the volatility of the market, this is required to ensure that customers send the coins using the latest market rate. After 15 minutes, there's a probability that the transaction won't be picked up by us.

2.6 3. Webhook Notification (Optional)

Once we receive the notification that a transaction is marked as sent, we sent a POST request to the callback URL you have specified. Specifying a callback URL is not mandatory since the inline JS client provides a mechanism to notify you when we detect that the coins have been sent and also, you can achieve almost the same result using background processes to verify/confirm transactions.

You can specify a callback URL in 2 different ways. Although, the second option overrides the first.

- **Business Settings:** You can specify the callback URL at your business profile page. This can be accessed by selecting a business and clicking **Profile** on the left menu.
- **Inline JS Client:** You can specify a callback URL while initializing a transaction on the inline JS client.

The body of the webhook notification request is a JSON object with the following parameters.

Parameter	Description
reference	The transaction reference. Pay attention to this if you didn't create a reference manually. (string)
sent	Specifies if payment was made successfully (boolean)
confirmed	Specifies if the payment has been confirmed on the blockchain (boolean)
amount	Specifies the intended amount in the currency selected during payment (number)
amountSent	Specifies the actual amount that was sent to the blockchain (number)

CHAPTER 3

Transaction Verification

In a situation where the call back url is not suitable, the developer might require a way to query transactions to know the status of transactions.

We have an API endpoint for checking transaction status with url as stated below:

https://api.paylot.co/transactions/verify/{reference}

reference: This is the reference used while creating a transaction or the reference returned in the callback.

To access this endpoint, the business secret key is required and this secret key can be obtained at the business profile page.

This key should never be exposed to the public and is of the format:

*pyt_sk-123455****

To send a request, use bearer authentication with your secret key as token i.e. add this to your request header.

*Authorization: Bearer pyt_sk-123455****

The API returns an object with the parameters described below.

Parameter	Description
reference	The transaction reference. Pay attention to this if you didn't create a reference manually. (string)
sent	Specifies if payment was made successfully (boolean)
confirmed	Specifies if the payment has been confirmed on the blockchain (boolean)
amount	Specifies the intended amount in the currency selected during payment (number)
amountSent	Specifies the actual amount that was sent to the blockchain (number)

CHAPTER 4

Split Payments

Split payments are necessary in cases where the merchant wants to split payments with another entity or bank account. To enable split payments, we use the concept of subaccounts. Sub accounts are bank accounts with the split percentage specified. The following describe how to manage subccounts on paylot.

NB: To manage your subaccounts, it's required that you have your API Secret key which can be accessed from your merchant profile on the dashboard.

4.1 Requirements

API Url: `https://api.paylot.co`

All requests accept and return **JSON**.

For all stated requests, it's required that you specify your **secret key** in the authorization header i.e

```
Authorizaion:  Bearer SECRET_KEY
```

This authenticates the specific merchant and ensures everything is done in the confines of the 1 merchant.

NB: ** specifies required properties while * specifies properties required for fiat only.

4.2 Create Sub Accounts

URL: `POST /subAccounts`

This creates a new sub account.

4.2.1 Request

The expected request is a JSON object of the format stated below.

```
{
  "currency": "NGN",
  "accountName": "JOHN & SONS Ltd.",
  "accountNumber": "0050505022",
  "bankName": "ACCESS BANK",
  "schedule": "auto",
  "percentage": 70
}
```

Property	Description
currency **	This specifies the currency of the account. (Possible options: NGN)
accountNumber **	This specifies the bank account number or for digital currencies, the wallet address
bankName *	This specifies the bank name or for digital currencies, the wallet name (optional for digital currencies)
accountName *	This specifies the name of the bank account or optional for digital currencies
schedule	This specifies how often the subaccount should receive payouts. The dafault is auto (24 hours interval). Other options are coming soon.
percentage **	This specifies the the percentage of payment they are expected to receive

4.2.2 Response

The expected response is a JSON object of the format stated below.

```
{
  "accountName": "JOHN & SONS Ltd.",
  "accountNumber": "0050505022",
  "bankName": "ACCESS BANK",
  "schedule": "auto",
  "percentage": 70,
  "reference": "1876187618761876"
}
```

The *reference* is a uniquely generated code for each subaccount and it's required for initializing split payments.

4.3 Update Sub Accounts

URL: `PUT /subAccounts/{id}`

This updates a specified subaccount by it's id.

4.3.1 Request

The expected request is a JSON object of the format stated below.

```
{
  "currency": "NGN",
```

(continues on next page)

(continued from previous page)

```
  "accountName": "JOHN & SONS Ltd.",
  "accountNumber": "0050505022",
  "bankName": "ACCESS BANK",
  "schedule": "auto",
  "percentage": 70
}
```

NB: Same as in Create sub accounts above

4.3.2 Response

The expected response is a JSON object of the format stated below.

```
{
  "accountName": "JOHN & SONS Ltd.",
  "accountNumber": "0050505022",
  "bankName": "ACCESS BANK",
  "schedule": "auto",
  "percentage": 70,
  "reference": "1876187618761876"
}
```

4.4 Get All Sub Accounts

URL: `GET /subAccounts`

This gets all the subaccounts of a specific merchant.

4.4.1 Response

The expected response is a JSON object of the format stated below.

```
[{
  "id": "1",
  "currency": {
      symbol: "NGN",
      name: "Naira"
  },
  "accountName": "JOHN & SONS Ltd.",
  "accountNumber": "0050505022",
  "bankName": "ACCESS BANK",
  "schedule": "auto",
  "percentage": 70,
  "reference": "1876187618761876"
},{
  "id": "2",
  "currency": {
      symbol: "NGN",
      name: "Naira"
  },
  "accountName": "JOHN & SONS Ltd.",
  "accountNumber": "0050505022",
```

(continues on next page)

(continued from previous page)

```
  "bankName": "ACCESS BANK",
  "schedule": "auto",
  "percentage": 70,
  "reference": "1876187618761876"
}]
```

4.5 Get Sub Account

URL: `GET /subAccounts/{id}`

This fetches a subaccount by it's id.

4.5.1 Response

The expected response is a JSON object of the format stated below.

```
{
  "id": "1",
  "currency": {
      symbol: "NGN",
      name: "Naira"
  },
  "accountName": "JOHN & SONS Ltd.",
  "accountNumber": "0050505022",
  "bankName": "ACCESS BANK",
  "schedule": "auto",
  "percentage": 70,
  "reference": "1876187618761876"
}
```

URL: `GET /subAccounts/ref/{reference}`

This fetches a subaccount by it's reference.

4.5.2 Response

The expected response is a JSON object of the format stated below.

```
{
  "currency": {
      symbol: "NGN",
      name: "Naira"
  },
  "accountName": "JOHN & SONS Ltd.",
  "accountNumber": "0050505022",
  "bankName": "ACCESS BANK",
  "schedule": "auto",
  "percentage": 70,
  "reference": "1876187618761876"
}
```

4.6 Usage

To split payments with a subaccount, it is required that you specify the reference of the subaccount while initializing a transaction. Check the integration section for more.