verifone

# MobilePay

## MobilePay Transactions

### Resources

| |
|---|
| MobilePay Direct API Reference |
| API Authentication |

These are the steps for processing a MobilePay transaction with the API:

1. Generate a unique `merchant_reference` that will be linked to this transaction. This reference needs to be used to link the order of the customer with the transaction `id` generated by Verifone.
2. Perform the create transaction API call and set the `redirect_url` to include the `merchant_reference` so you can display the transaction result when they return to your server. The `redirect_url` can include the reference as part of the url ( `/order/1234` ) or as a query parameter ( `/order?merchant_reference=1234` ).
3. Redirect the shopper to the `redirect_url` provided by MobilePay in the response of the API call. When the customer has completed the payment on their telephone, they are returned to the `redirect_url` you provided in the request body.
4. When the customer visits the `redirect_url` you initially provided, there are two options for retrieving the transaction status:
    - Use the notification functionality and wait until the webhook is received that the transaction has been completed ( `"eventType":"TxnSaleApproved"` )
    - Query the GET transaction endpoint using the `id` returned in the API call from step 3

## Required fields

| parameters | Description |
|---|---|
| `payment_provider_contract` | In the Payment Provider Contracts section in Verifone Central, set the *Payment Type* to *MobilePay*, select your contract and copy the Payment Provider Contract ID. Please note this value is different in Sandbox and in Production. |
| `redirect_url` | Generate a unique link and include the `merchant_reference` either as a query parameter or as part of the URL ( `https://merchant.com/order/1234` or `https://merchant.com/order?merchant_reference=1234` ) |
| `amount` | Amount of the transaction |
| `customer` | Customer ID |
| `merchant_reference` | Unique UUID you generate and can link the transaction to when the customer returns |
| `currency_code` | Accepted values are EUR, NOK, SEK and DKK |

## 1. Generate a unique merchant_reference

The `merchant_reference` needs to be unique to identify the shopper when they are redirected to your server by MobilePay. In the next step, you will be creating a transaction through the API, the transaction will return an `id` that needs to be stored safely with the reference. When the shopper returns, you can use this reference to confirm either through the webhook or through the GET transaction API call if the shopper has successfully completed the transaction.

## 2. Create Transaction API call

POST the following example to the MobilePay endpoint: `/oidc/api/v2/transactions/mobilepay`.

Unsure what URL you should be using? The full list can be found here.

### Request

#### Headers

| | |
|---|---|
| URL | List of available servers |
| Endpoint | /oidc/api/v2/transactions/mobilepay |
| x-vfi-api-idempotencyKey | Unique UUID to identify the transaction |
| Authorization | Basic {{your encoded user id and api secret}} |
| Content-Type | application/json |

#### Body

```
{
"payment_provider_contract": "{{replace with your own payment_provider_contract}}",
"redirect_url": "https://yourwebsite.com/order/{{replace with your merchant_reference}}",
"amount": 100,
"customer": "{{replace with your own customer object}}",
"merchant_reference": "{{generate a unique merchant_reference}}",
"currency_code": "EUR"
}
```

### Response

> The `redirect_url` in the request is the site where the shopper is sent to **after** completing the transaction. The `redirect_url` that is returned in the response is the **MobilePay URL** the shopper must visit to complete the payment.

From the response, the `id` of the transaction should now be stored along with the `merchant_reference` from the first step. When the shopper returns to your environment, you can confirm the `merchant_reference` by cross referencing it with the transaction status using the `id`.

You should now use the `redirect_url` from the **response** to redirect the shopper to MobilePay's website.

```
{
"amount": 100,
"blocked": false,
"created_at": "2022-03-23T13:00:53.191575Z",
"customer": "{{customer_id}},
"details": {
"auto_capture": true
},
"merchant_reference": "{{merchant_reference}}",
"status": "INITIATED",
"created_by": "{{created_by}}",
"country_code": "FI",
"id": "4cbcf8af-b36e-451a-8874-c6fa9976362c",
"redirect_url": "https://sandprod-products.mobilepay.dk/remote-website/index.html?page=request&id={{id}}",
"processor": "NETS",
"payment_product": "CARD",
"payment_product_type": "Unknown"
}
```


## 3. Redirect the shopper

Redirect the shopper to the `redirect_url` from the previous step. The shopper will now complete the MobilePay payment. After completion or cancellation, the shopper will be sent back to the `redirect_url`.

## 4. Retrieving the transaction status

When the customer is returned to your environment, you have two options to confirm the status of the transaction:

1. Confirm the transaction through the notification service with webhooks/emails
2. Query the GET transaction endpoint `/oidc/api/v2/transaction/{{id}}` using the `id` from the response in step 2

### 4.1 Retrieving the transaction status using the notification service

Following the steps in the notification service documentation, set up a webhook to be sent for the event type `TxnSaleApproved` and `TxnSaleDeclined` for your organization. When Verifone receives a notification that the transaction has been completed or declined, the webhook will be sent. Here is an example of a webhook:


```
{
"eventId": "1",
"eventDateTime": "2022-03-23T11:07:28Z",
"recordId": "{{transaction id}}",
"eventType": "TxnSaleApproved"
}
```


The `recordId` field will contain the transaction ID, and the `eventType` field can be parsed to view the outcome of the transaction. For the response in step 2 of this guide, you stored the combination of the `merchant_reference` and transaction `id` transaction.

### 4.2 Retrieving the transaction status by querying /oidc/api/v2/transaction/{{id}}

Alternatively, the direct API can be used to query the transaction status. After the shopper returns to your site, you can do a GET request to search for the `status` of the transaction:

### Request

#### Headers

| | |
|---|---|
| URL | List of available servers |
| Endpoint | /oidc/api/v2/transactions/{{id}} |
| Authorization | Basic {{your encoded user id and api secret}} |
| Content-Type | application/json |

### Response

The response will have the `status` field which can be used to determine the outcome of the transaction.


```
{
"id": "46bf6f96-0103-4f53-a538-33444cea86ee",
"amount": "1.00",
"currency_code": "EUR",
"created_at": "2022-03-23T14:24:45.241Z",
"customer": "{{customer id}}",
"shipping_information": {
"phone": "633344455",
"address": "Singel 250",
"city": "Amsterdam",
"state": "Noord-Holland",
"postal_code": "1012 AB",
"country": "FI"
},
"merchant_reference": "{{merchant reference}}",
"payment_product": "CARD",
"payment_product_type": "MASTERCARD",
"status": "SALE AUTHORISED",
"processor_reference": "MCC0124B60323",
"arn": "231424000518",
"rrn": "231424000518",
"cvv_present": true,
"authorization_code": "026257",
"reason_code": "0000",
"shopper_interaction": "ecommerce",
"pos_device_id": "40000564",
"stan": "518",
"masked_card_number": "541303******0005"
}
```