



# Centiglobe system certification [payin]

Centiglobe certification of member integration aims to improve system stability and the ability to recover from system failures and outages.

Centiglobe delivers a set of self-service scenarios, which also is the base of the certification.

<b>Introduction</b>	<b>2</b>
Connection endpoint	2
<b>Concepts</b>	<b>2</b>
Payment steps	2
Recovery	2
paymentRef vs paymentId	2
<b>Description of payment instruction</b>	<b>3</b>
<b>Scenarios/Certification</b>	<b>4</b>
Standard payment	5
Peer-to-Peer (P2P) payment	6
Prepare sanction screening rejected	7
Payment confirmed aborted	8
Payment error aborted	9
Payment error refunded	10
Payment refresh accepted	11
Payment refresh refunded	12
Payment manual accepted	13
Payment manual refunded	14
Payment timeout accepted delayed reply	15
Refund reverse, to payin currency	16
Refund reverse, to stable token	17
Refund retry	18
Ledger issue member token	19
Ledger redeem member tokens	20
Ledger transfer tokens	21



# Introduction

To aid system stability, members need to comply with a set of Certification criteria. The certification is based on scenarios that can happen in real production setups, e.g., pending payments and how to resolve them.

## Connection endpoint

Scenarios are executed towards a proxy service whose endpoint is defined in the API documentation <https://doc.system.centiglobe.com/>.

# Concepts

## Payment steps

Payments are processed in two steps. The first step is validation of payment viability. If payments are rejected during the first step, no money has been transferred, and the debtor can be notified to e.g. amend the payment. After a successful preparation phase, the payment phase begins.

## Recovery

### paymentRef vs paymentId

To enable recovery of payments that never got any reply there is a concept of paymentRef and paymentId.

- paymentRef is created and sent from pay-in members and unique in the context of the pay-in members account.
- paymentId is created by Centiglobe and is the unique id from the blockchain

Using the paymentRef to query for a paymentId is how the pay-in member can recover a lost payment.



## Description of payment instruction

To execute the scenario 'Prepare sanction screening rejected' the name of the scenario is used in the serviceAccount field.

```
{
  "timestamp": "2022-04-13T07:43:45Z",
  "payin": {
    "memberName": "<certifying-member-name>",
    "currency": "EUR"
  },
  "payout": {
    "memberName": "payout-fake",
    "currency": "SEK",
    "service": "scenario-based",
    "amount": "0.034"
  },
  "amlInformation": {
    "$schema": "https://schema.centiglobe.com/aml/id-test-3-20220307.json",
    "fullName": "Elvis Presley",
    "serviceAccount": "prepare-sanction-screening-rejected"
  }
}
```

Send payment to the scenario proxy endpoint defined in API documentation.

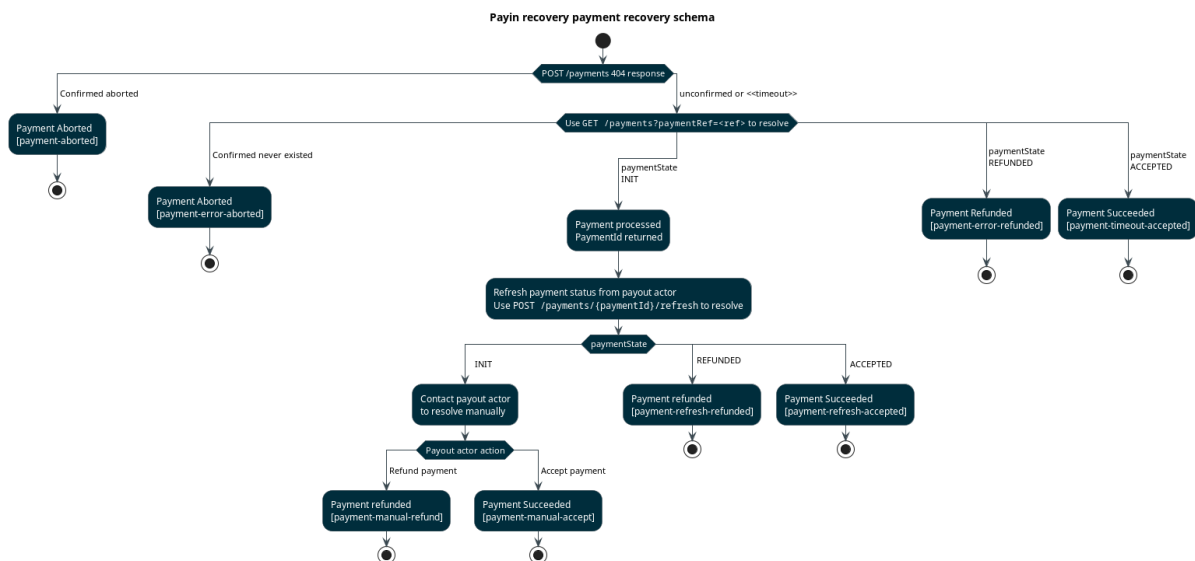
Field	Description
payin.memberName	Name of certifying member. Created at setup.
payin.currency	One of the currencies configured for certifying the member.
payout.memberName	Payout member name used for certification which is <i>payout-fake</i> .
payout.currency	Available currencies (and implied) SEK (USD.FCAT) USD (USD.FCAT)
payment.service	Name of service which in this context is ' <i>scenario-based</i> '
amlInformation	Payment aml information need to comply with schema <a href="https://schema.centiglobe.com/aml/id-test-3-20220307.json">https://schema.centiglobe.com/aml/id-test-3-20220307.json</a>
amlInformation. serviceAccount	Use this field to specify which scenario to use



# Scenarios/Certification

Certification is grouped into the following areas

- Standard payments
- Payment recovery
- Refund
- Ledger operations





## Standard payment

Standard payment which gets accepted by payout.

```
{
  "timestamp": "2022-04-13T07:43:45Z",
  "payin": {
    "memberName": "<certifying-member-name>",
    "currency": "EUR"
  },
  "payout": {
    "memberName": "payout-fake",
    "currency": "SEK",
    "service": "scenario-based"
    "amount": "0.034"
  },
  "amlInformation": {
    "$schema": "https://schema.centiglobe.com/aml/id-test-3-20220307.json",
    "fullName": "Elvis Presley",
    "serviceAccount": "standard-payment"
  }
}
```

### Send

POST /payment-statements  
***standard-payment***

POST /payments

### Expect

200 - Success

201 - Created  
state: ACCEPTED

- ☐ Completed
- ☐ Exempted

### Note



## Peer-to-Peer (P2P) payment

Peer-to-Peer payment to net imbalances. Certifying member sends fully collateralized asset tokens (FCAT) to payout and expects corresponding fiat currency on account.

```
{
  "timestamp": "2022-04-13T07:43:45Z",
  "payin": {
    "memberName": "<certifying-member-name>",
    "currency": "USD.FCAT"
  },
  "payout": {
    "memberName": "payout-fake",
    "currency": "USD",
    "service": "scenario-based",
    "amount": "0.034"
  },
  "amlInformation": {
    "$schema": "https://schema.centiglobe.com/aml/id-test-3-20220307.json",
    "fullName": "Elvis Presley",
    "serviceAccount": "standard-payment"
  }
}
```

### Send

### Expect

POST /payment-statements  
*standard-payment*  
payinCurrency: USD.FCAT  
payoutCurrency: USD

200 - Success

POST /payments

201 - Created  
state: ACCEPTED

- ☐ Completed
- ☐ Exempted

---

### Note



## Prepare sanction screening rejected

Payout members will perform a sanction screening and this scenario simulates a rejection.

```
{
  "timestamp": "2022-04-13T07:43:45Z",
  "payin": {
    "memberName": "<certifying-member-name>",
    "currency": "<CURRENCY>"
  },
  "payout": {
    "memberName": "payout-fake",
    "currency": "<CURRENCY>",
    "service": "scenario-based"
  },
  "amlInformation": {
    "$schema": "https://schema.centiglobe.com/aml/id-test-3-20220307.json",
    "fullName": "Elvis Presley",
    "serviceAccount": "prepare-sanction-screening-rejected"
  }
}
```

### Send

POST /payment-statements  
*prepare-sanction-screening-rejected*

### Expect

404 - Not found  
Error code: SANCTION\_SCREENING\_REJECTED

The upstream system can be notified about pre validation of payment failed.

- ☐ Completed
- ☐ Exempted

---

### Note



## Payment confirmed aborted

An error occurred early in the payment flow and the payment was confirmed aborted. No tokens transferred in blockchain.

### Send

POST /payment-statements  
*payment-confirmed-aborted*

POST /payments

### Expect

200 - Success

404 - Not found  
Error code: PAYMENT\_ERROR  
confirmedAbort: true

Upstream system can be notified about the aborted payment.

- ☐ Completed
- ☐ Exempted

---

### Note





## Payment error aborted

An error occurred during payment. Recovery concluded no payment was created on the blockchain, which implies no tokens have been transferred.

### Send

POST /payment-statements  
*payment-error-aborted*

POST /payments  
Store sending time

GET /payments?paymentRef=<ref>

### Expect

200 - Success

404 - Not found  
Error code: PAYMENT\_ERROR  
confirmedAbort: false

200 - Success  
Expect empty array

### Note

Sending time of payment *must* be within `finalitySpan` before the empty array can be interpreted as non-existing payment.  
(see payment-timeout-accepted-delayed-reply)

Upstream system can be notified about the aborted payment.

☐ Completed

☐ Exempted

---

### Note



## Payment error refunded

An error occurred during payment. Recovery concluded that payment was created on the blockchain and later refunded.

Send	Expect
POST /payment-statements <i>payment-error-refunded</i>	200 - Success
POST /payments	404 - Not found Error code: PAYMENT_ERROR confirmedAbort: false
GET /payments?paymentRef=<ref>	200 - Success Expect array to contain payment state: REFUNDED
POST /payments/{id}/reverse	200 - Success state: REFUND_RESOLVED  Upstream system can be notified about the completed refund.

- ☐ Completed
- ☐ Exempted

---

**Note**



## Payment refresh accepted

An error occurred during payment and payment could have been created on the blockchain (confirmedAbort: false). Recovery reveals the payment was created on the blockchain and the payout member has processed the payment.

### Send

POST /payment-statements  
*payment-refresh-accepted*

POST /payments

GET /payments?paymentRef=<ref>

POST /payments/{id}/refresh

### Expect

200 - Success

404 - Not found  
Error code: PAYMENT\_ERROR  
confirmedAbort: false

200 - Success  
PaymentId returned  
state: INIT

200 - Success  
state: ACCEPTED

Upstream system can be notified about the successful payment.

- ☐ Completed
- ☐ Exempted

---

### Note



## Payment refresh refunded

An error occurred during payment and payment could have been created on the blockchain (confirmedAbort: false). Recovery reveals the payment was created on the blockchain but the payout member never processed the payment, hence a refund.

### Send

POST /payment-statements  
*payment-refresh-refunded*

POST /payments

GET /payments?paymentRef=<ref>

POST /payments/{id}/refresh

POST /payments/{id}/reverse

### Expect

200 - Success

404 - Not found  
Error code: PAYMENT\_ERROR  
confirmedAbort: false

200 - Success  
PaymentId returned  
state: INIT

200 - Success  
state: REFUNDED

200 - Success  
state: REFUND\_RESOLVED

Upstream system can be notified about the completed refund.

☐ Completed

☐ Exempted

---

### Note



## Payment manual accepted

An error occurred during payment and payment could have been created on the blockchain (confirmedAbort: false). Recovery reveals nothing, the payment still pending. Payin member need to contact payout member to manually resolve the situation. In this case Payout will accept payment.

### Send

POST /payment-statements

*payment-manual-accepted*

POST /payments

GET /payments?paymentRef=<ref>

POST /payments/{id}/refresh

*simulated contact with payout member*

GET /payments/{paymentId}

### Expect

200 - Success

404 - Not found

Error code: PAYMENT\_ERROR  
confirmedAbort: false

200 - Success

PaymentId returned  
state: INIT

200 - Success

state: INIT

After a two minute timeout the payment is refunded

Before the simulated timeout  
state: INIT

After the simulated timeout  
state: ACCEPTED

Upstream system can be notified about the completed payment.

☐ Completed

☐ Exempted

---

### Note



## Payment manual refunded

An error occurred during payment and payment could have been created on the blockchain (confirmedAbort: false). Recovery reveals nothing, the payment still pending. Payin member need to contact payout member to manually resolve the situation. In this case Payout will refund payment.

### Send

POST /payment-statements

*payment-manual-refunded*

POST /payments

GET /payments?paymentRef=<ref>

POST /payments/{id}/refresh

*simulated contact with payout member*

GET /payments?state=REFUNDED

POST /payments/{id}/reverse

### Expect

200 - Success

404 - Not found

Error code: PAYMENT\_ERROR

confirmedAbort: false

200 - Success

PaymentId returned  
state: INIT

200 - Success

state: INIT

After a two minute timeout the payment is refunded

After the simulated timeout, the list returned contains the payment with state REFUNDED.

200 - Success

state: REFUND\_RESOLVED

Upstream system can be notified about the completed refund.

☐ Completed

☐ Exempted

### Note



## Payment timeout accepted delayed reply

Payment service call times out, payment is still in-flight. Scenario simulates a delay until payment created on blockchain and the use of finality span to resolve the situation.

### Send

POST /payment-statements  
*payment-timeout-accepted-delayed-reply*

POST /payments  
[sending time]

GET /payments?paymentRef=<ref>

Wait until [sending time]  
within finaliySpan

GET /payments?paymentRef=<ref>

POST /payments/{id}/refresh

### Expect

200 - Success

<<timeout>>

200 - Success  
Expect empty array

**Note**  
Sending time of payment *must* be within `finaliySpan` before the empty array can be interpreted as non-existing payment.

**Note**  
Current grace period in the Centiglobe system is 5 minutes.

200 - Success  
PaymentId returned in array  
state: INIT

200 - Success  
state: ACCEPTED

Upstream system can be notified about the accepted payment.

- ☐ Completed
- ☐ Exempted

### Note



## Refund reverse, to payin currency

A payment ends up in a refund state. Resolve the situation by reversing payment quotes back to source currency.

### Send

POST /payment-statements  
*payment-error-refunded*  
[payin.currency]

POST /payments

GET /payments?paymentRef=<ref>

POST /payments/{id}/reverse  
reverseToCurrency=[payin.currency]

### Expect

200 - Success

404 - Not found  
Error code: PAYMENT\_ERROR  
confirmedAbort: false

200 - Success  
Expect array to contain payment  
state: REFUNDED

200 - Success  
state: REFUND\_RESOLVED

Upstream system can be notified about the  
completed refund.

- ☐ Completed
- ☐ Exempted

---

### Note





## Refund reverse, to stable token

A payment ends up in a refund state. Resolve the situation by reversing payment quotes back to stable tokens.

Send	Expect
POST /payment-statements <i>payment-error-refunded</i>	200 - Success
POST /payments	404 - Not found Error code: PAYMENT_ERROR confirmedAbort: false
GET /payments?paymentRef=<ref>	200 - Success Expect array to contain payment state: REFUNDED
POST /payments/{id}/reverse reverseToCurrency=[stable token]	200 - Success state: REFUND_RESOLVED  Upstream system can be notified about the completed refund.

- ☐ Completed
- ☐ Exempted

---

**Note**



## Refund retry

A payment ends up in a refund state. Resolve the situation by retrying the payment.

Send	Expect
POST /payment-statements <i>payment-error-refunded</i>	200 - Success
POST /payments	404 - Not found Error code: PAYMENT_ERROR confirmedAbort: false
GET /payments?paymentRef=<ref>	200 - Success Expect array to contain payment state: REFUNDED paymentId: [id]
POST /payment-statements <i>standard-payment</i>	200 - Success
POST /payments refundPaymentId=[id]	201 - Created  Upstream system can be notified about the completed payment.

- ☐ Completed
- ☐ Exempted

---

### Note



## Ledger issue member token

Members will issue their own member tokens.

### Send

```
POST ledger-service
/member-tokens/{memberTokenName}
/issue
```

### Expect

200 - Success

```
GET ledger-service
/member-tokens/{memberTokenName}
```

200 - Success

Expect issuedAmount and balance to have increased

- ☐ Completed
- ☐ Exempted

---

### Note



## Ledger redeem member tokens

Members will redeem their own member tokens. To redeem tokens the member needs to have sufficient balance.

### Send

```
POST ledger-service
/member-tokens/{memberTokenName}
/redeem
```

### Expect

200 - Success

```
GET ledger-service
/member-tokens/{memberTokenName}
```

200 - Success

Expect issuedAmount and balance to have decreased

☐ Completed

☐ Exempted

### Note



## Ledger transfer tokens

Members will transfer tokens to another account.

### Send

```
POST ledger-service
/member-tokens/{memberTokenName}
/transfer
accountName=payout-fake
```

### Expect

200 - Success

```
GET ledger-service
/member-tokens/{memberTokenName}
```

200 - Success

Expect balance to have decreased.  
IssuedAmount should remain intact.

- ☐ Completed
- ☐ Exempted

---

### Note