ANASTASIA LABS

HOW TO USE

Payment Subscription Smart Contract

Project Number	1100025
Project Manager	Jonathan Rodriguez
Project Name:	Anastasia Labs X Maestro - Plug ‘n Play 2.0
URL:	Catalyst Proposal

Contents

How to Use the Payment Subscription Smart Contract via Maestro

1. Introduction

The Payment Subscription smart contract is designed to manage recurring payments on the Cardano blockchain. It automates subscriber-to-merchant interactions such as service creation, account registration, subscription initiation, extension, cancellation, and fund withdrawals. This guide assists developers in integrating these functions into their applications via Maestro's API endpoints.

Key features include:

- Initiating subscriptions with customizable terms
- Extending or terminating subscriptions
- Automated recurring payments
- Secure withdrawal of funds for both merchants and subscribers
- Seamless integration with popular wallet applications and RESTful API endpoints

2. Prerequisites

Before you begin, ensure that you have:

- **Access Credentials:**
  - Maestro API Key (Get from **Maestro Getting Started**)
  - Cardano wallet address with sufficient funds for either the merchant or subscriber
- **Environment Setup:**
  - Basic familiarity with REST APIs and JSON data formats.
  - A command-line environment and ability to execute **cURL** commands in a terminal/ command-line environment.

3. Smart Contract Overview

The Payment Subscription smart contract is built using Aiken and comprises three key components:

1. **Service Contract:**

   Responsible for creating a service by minting a CIP-68 compliant Service NFT, managing service metadata, and handling service deactivation.

2. **Account Contract:**

   Handles the creation of subscriber accounts via minting a CIP-68 compliant Account NFT, updating subscriber metadata, and account removal.

3. **Payment Contract:**

   Manages the locking of prepaid subscription fees, subscription initiation, extension, unsubscription, and fund withdrawals. A linear vesting mechanism gradually releases funds to merchants as per the subscription schedule.

Each function is accompanied by detailed onchain and offchain documentation. This guide focuses on how each API endpoint maps to these smart contract functionalities.

4. Merchant Operations

Merchant operations primarily include creating a new service and withdrawing subscription fees. These actions allow merchants to configure their service offerings and claim funds that accumulate from subscriber payments.

To use these operations effectively:

- Replace **`${API_KEY}`**, with your Maestro **`API_KEY`** and fill in the required parameters.
- Remember to enter the correct **NETWORK: preview | preprod | mainnet**
- Your addresses should match the chosen Network.

4.1. Create a Service

This endpoint registers a new service by minting a Service NFT along with its reference NFT. It captures essential parameters such as service fee, penalty fee, subscription interval, and service status.

Endpoint:

```
POST https://preprod.gomaestro-api.org/v1/contracts/subscription/createService
```

Required Parameters:

- **`merchant_address`**: Address of the merchant.
- **`selected_out_ref`**: UTxO **`tx_hash`** and **`output_index`** used to derive token names.
- **`service_fee_policyid`**: Policy ID governing the service fee asset.
- **`service_fee_assetname`**: Asset name for the service fee.
- **`service_fee`**: Fee amount for the service.
- **`penalty_fee_policyid`**: Policy ID for the penalty fee asset.
- **`penalty_fee_assetname`**: Asset name for the penalty fee.
- **`penalty_fee`**: Penalty fee amount (for early cancellations).
- **`interval_length`**: Duration of one subscription interval (in milliseconds).
- **`num_intervals`**: Total number of intervals in the subscription period
- **`is_active`**: Boolean flag indicating the service's active state.

cURL Example:

```
# subscription: create service

curl --location 'https://preprod.gomaestro-api.org/v1/contracts/subscription/
createService' \
--header 'Content-Type: application/json' \
--header 'api-key: ${API_KEY}' \
--data '{
    "merchant_address":
"addr_test1qzngfvxfk2grdas3daahzku733639z0lacyergaerf382rcungt5wwjcw2ef63324vchv7k9ryds4030h3hfw2us2vs
    "selected_out_ref": {
        "tx_hash": "9bae7f98ee3c363f7d8dd47e57b5458d272e4d0ed1c3f21d14662f42aac31776",
        "output_index": 2
    },
    "service_fee_policyid": "",
    "service_fee_assetname": "",
    "service_fee": 1000,
    "penalty_fee_policyid": "",
    "penalty_fee_assetname": "",
    "penalty_fee": 10,
    "interval_length": 2592000000,
    "num_intervals": 1,
    "is_active": true
}'
```

4.2. Withdraw Fees

This merchant operation permits withdrawal of accumulated subscription fees from the Payment Contract. It verifies that the request complies with the vesting rules defined in the contract and uses the Payment NFT along with the Service NFT to authorize and execute the withdrawal.

Endpoint:

```
POST https://preprod.gomaestro-api.org/v1/contracts/subscription/withdrawFees
```

Required Parameters:

- **`merchant_address`**: Address of the merchant.
- **`service_nft_tn`**: The token name for the Service NFT.
- **`subscriber_nft_tn`**: The token name for the Subscriber NFT.
- **`merchant_nft_tn`**: The token name proving merchant ownership.
- **`payment_nft_tn`**: The Payment NFT token involved.
- **`current_time`**: Current Unix timestamp to validate withdrawal timing.

cURL Example:

```
# subscription: merchant withdrawal

curl --location 'https://preprod.gomaestro-api.org/v1/contracts/subscription/withdrawFees'
\
--header 'Content-Type: application/json' \
--header 'api-key:  ${API_KEY}' \
--data '{
    "merchant_address":
"addr1qxccwptmx6r523vxcrplvfhtrpdut8s0hht0fyt9f8vy8385chtg2dupkyqu7pgqawju7awrwfg94skstmaves6hwaks6qlg
    "service_nft_tn": "000643b0006d5fd6a8ebe94e12ea74d46ee2bd5621a8e8d55df0bfa9419ed7ed",
    "subscriber_nft_tn":
"000de14002cfa3e99ad2d48c991d02db0adedf4b8b08ac72746e5db6a3938c57",
    "merchant_nft_tn": "000de140006d5fd6a8ebe94e12ea74d46ee2bd5621a8e8d55df0bfa9419ed7ed",
    "payment_nft_tn": "subscription",
    "current_time": 4288320000
}'
```

5. Subscriber Operations

Subscriber operations cover account creation, subscription initiation, and unsubscription. These endpoints enable users to register for services, lock funds, and later cancel subscriptions according to the contract rules.

5.1. Create User Account

This endpoint creates a subscriber account by minting an Account NFT along with its reference NFT. It binds subscriber details (such as **`email`** or **`phone`**) to ensure account integrity before locking funds for subscriptions.

Endpoint:

```
POST https://preprod.gomaestro-api.org/v1/contracts/subscription/createUserAccount
```

Required Parameters:

- **`subscriber_address`**: Address of the subscriber.
- **`selected_out_ref`**: UTxO with **`tx_hash`** and **`output_index`**.
- **`email`**: Subscriber's email address as a string
- **`phone`**: Subscriber's phone number as a string.

cURL Example:

```
# subscription: create user account

curl --location 'https://preprod.gomaestro-api.org/v1/contracts/subscription/
createUserAccount' \
--header 'Content-Type: application/json' \
--header 'api-key: ${API_KEY}' \
--data '{
    "subscriber_address":
"addr_test1qpp9s0ml0z9lg8ym2ueh0jsay6dg7dj0aslx2lnlda9sukk6ryucc4hthndrvllxq584xw6r2q8hynv6sslrammpmd5
    "selected_out_ref": {
        "tx_hash": "e3dd79c0e578aabdac13f1f35a2ed07ad3a0e82e0e4d6eec8ee4c3a2919846c8",
        "output_index": 1
    },
    "email": "business@web3.ada",
    "phone": "288-481-2686"
}'
```

5.2. Initiate Subscription

Locks funds by minting a Payment NFT and setting up a Payment datum that includes the subscription start time, service reference token, and subscriber reference token. This action creates a unique Payment Token that signifies the start of a subscription.

Endpoint:

```
POST https://preprod.gomaestro-api.org/v1/contracts/subscription/initSubscription
```

Required Parameters:

- **`service_nft_tn`**: Combined token string for the service NFT.
- **`subscriber_nft_tn`**: Combined token string for the subscriber NFT.
- **`subscription_start`**: Unix timestamp marking the start of the subscription.

cURL Example:

```
# subscription: initiate subscription

curl --location 'https://preprod.gomaestro-api.org/v1/contracts/subscription/
initSubscription' \
--header 'Content-Type: application/json' \
--header 'api-key: ${API_KEY}' \
--data '{
    "service_nft_tn": "000643b0006d5fd6a8ebe94e12ea74d46ee2bd5621a8e8d55df0bfa9419ed7ed",
    "subscriber_nft_tn":
"000de14002cfa3e99ad2d48c991d02db0adedf4b8b08ac72746e5db6a3938c57",
    "subscription_start": 1696320000
}'
```

5.3. Unsubscribe

This subscriber operation allows a user to cancel an active subscription. The endpoint processes the unsubscription request, calculates any applicable penalty fees (if the service is still active), and updates the contract state.

Endpoint:

```
POST https://preprod.gomaestro-api.org/v1/contracts/subscription/unsubscribe
```

Required Parameters:

- **`subscriber_address`**: Address of the subscriber initiating the unsubscription.
- **`service_nft_tn`**: The token string for the Service NFT.
- **`subscriber_nft_tn`**: The token string for the Subscriber NFT.
- **`current_time`**: Unix timestamp used for validating penalty application.

cURL Example:

```
# subscription: unsubscribe

curl --location 'https://preprod.gomaestro-api.org/v1/contracts/subscription/unsubscribe'
\
--header 'Content-Type: application/json' \
--header 'api-key: 2eg9m9dyrj491urTQ74qsd6k7PwfHaz4' \
--data '{
    "subscriber_address":
"addr_test1qpp9s0ml0z9lg8ym2ueh0jsay6dg7dj0aslx2lnlda9sukk6ryucc4hthndrvllxq584xw6r2q8hynv6sslrammpmd5
    "service_nft_tn": "000643b0006d5fd6a8ebe94e12ea74d46ee2bd5621a8e8d55df0bfa9419ed7ed",
    "subscriber_nft_tn":
"000de14002cfa3e99ad2d48c991d02db0adedf4b8b08ac72746e5db6a3938c57",
    "current_time": 1696320000
}'
```

Note: Early termination may incur penalties if defined in the service contract.

6. Conclusion

This guide has provided a comprehensive walkthrough for using the Payment Subscription smart contract via Maestro's API endpoints. By following these instructions, developers can easily:

- Register and configure services via the Service Contract.
- Enable subscribers to create accounts, initiate subscriptions, and cancel subscriptions.
- Facilitate secure fund withdrawals for merchants in accordance with the subscription vesting rules.

Before going live, replace any placeholder values (e.g., **`${API_KEY}`** and wallet addresses) with actual data, and test each endpoint in a sandbox environment. For more detailed technical references, please consult the the associated documentation provided by the project.

7. Links

- Payment Subscription Smart Contract
- Payment Subscription Offchain SDK