ANASTASIA LABS

# Project Design Specification

| | |
|---|---|
| **Project Number** | 1000013 |
| **Project manager** | Philip DiSarro |
| **Date Started** | February 24, 2024 |
| **Date Completed** | ... |

# Contents

# Payments Subscription Smart Contract

## 1. Overview

This Payments Subscription Smart Contract is developed using Aiken, to facilitate automated recurring payments between Subscribers and Merchants on the Cardano blockchain. This smart contract will enable users to set up, manage, and cancel subscriptions directly from their wallets.

## 2. Architecture

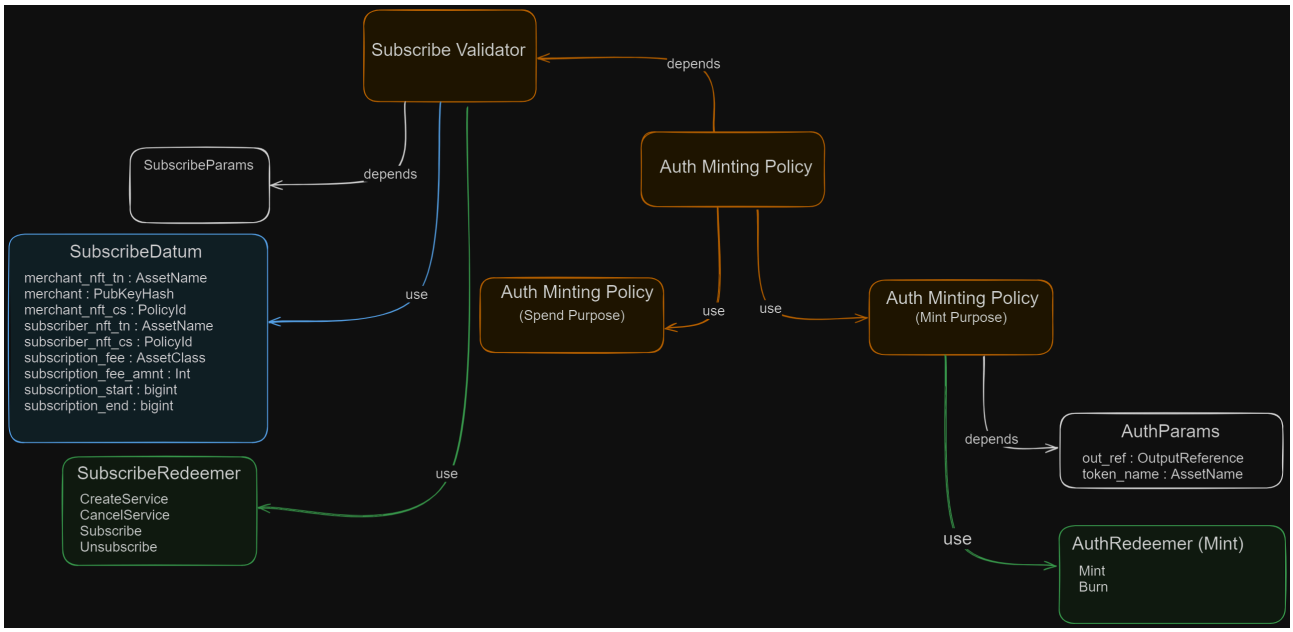


Figure 1: Contract Architecture

There are two contracts in this subscription system.

- **Subscribe Contract:** Is responsible for creating services they provide, holding the prepaid subscription fees for the respective service awaiting merchant withdrawal.
- **Auth Minting Policy:** Responsible for creating initial service, minting and burning MerchantNFT and SubscriberNFT as well as tracking and updating the respective tokens.

# 3. Specification

## 3.1. System Actors

- **Merchant:** An entity who interacts with the Subscribe Contract in order to create a service and receives subscription payments for the respective service/ services.
- **Subscriber:** An entity who wants to interact with the Subscribe Contract to deposit/withdraw prepaid subscription fees.

## 3.2. Tokens

- **Merchant NFT:** Can only be minted by a merchant when creating a subscription service and burned when merchant removes their service/services from the system. Datum is updated when a subscription is paid or fees are withdrawn from Subscribe Contract.
  - Policy Id: Auth Minting Policy
  - TokenName: Defined in Auth Minting Policy parameters (e.g. "MERCHNFT")
- **Subscriber NFT:** Can only be minted when a subscription fee is paid to Subscribe Contract and burned when subscriber exits the system. Datum is updated when fees are withdrawn from Subscribe Contract.
  - Policy Id: Auth Minting Policy
  - TokenName: Defined in Auth Minting Policy parameters (e.g. "SUBNFT")

## 3.3. Smart Contract

### 3.3.1. Subscribe Validator

Subscribe validator is responsible for validating subscriptions and holding "Subscriber Requests" funds and subscription details.

#### 3.3.1.1. Parameters

None

#### 3.3.1.2. Datum

- **`merchantNftTn`:** Merchant's token name encoding UTXO to be consumed when minting the token.
- **`merchant`:** Who should recieve the subscription fees.
- **`merchant_nft_id`:** Policy Id of the Merchant's NFT.

- **`subscriberNftTn`:** Subscriber's token name encoding UTXO to be consumed when minting the token.
- **`subscriber_nft_id`:** Policy Id of the Subscriber's NFT.
- **`subscriptionFee`:** AssetClass type for the subscription fee.
- **`subscriptionFeeAmnt`:** Amount of the subscription fee.
- **`subscriptionStart`:** Start of the subscription.
- **`subscriptionEnd`:** Expiry time of the subscription.

#### 3.3.1.3. Redeemer

- CreateService
- CancelService
- Subscribe
- Unsubscribe

#### 3.3.1.4. Validation

- **CreateService:** The redeemer will allow an entity to create a new subscription service.
  - validate that transaction has to be executed by a payment sinature.
  - validate the transaction mints one token of name MerchantNFT.
- **CancelService:** The redeemer allows any to spend Subscribe UTxO to get back locked funds.
  - validate payment signature is equal to merchant payment signature.
  - validate that there is a single UTxO in the transaction input and contains a single merchentNFT token.
  - validate that unlocked funds are sent to `merchant`
  - validate that transaction burns one token MerchantNFT.
- **Subscribe:** This redeemer allows spending Subscribe UTxO to subscribe to a service.
  - validate that the transaction must execute only after `subscriptionStart`
  - validate that transaction contains `subscriptionFee` fee amount equal or more than the amount in service datum.
  - validate that transaction contains the correct service owner / `merchant` to withdraw the funds to.
  - validate that transaction mints or updates only one token of name SubscriberNFT.
  - validate that transaction mints or updates only one token of name MerchantNFT.
- **Unsubscribe:** The redeemer will allow anyone with a subscriberNFT to spend Subscribe UTxO to unlock funds back to their address.
  - validate that there is a single UTxO in the transaction input and contains a single subscriberNFT token.
  - validate that transaction burns one token of name SubscriberNFT.

#### 3.3.2. Auth Minting Policy

Auth Minting Policy is responsible for creating and burning Subscriber and Merchant NFT Tokens

##### 3.3.2.1. Parameter

- **out_ref:** is a Reference of an Unspent Transaction Output, which will only be spent on Subscribe and CreateService redeemer to make sure this redeemer can only be called once.
- **token_name:** is identified as the service name prefixed to the token name of the merchant or subscriber.

##### 3.3.2.2. Minting Purpose

###### 3.3.2.2.1. Redeemer

- Mint
- Burn

###### 3.3.2.2.1.1. Validation

- **Mint:** The redeemer allows creating of a new subscriprion sevice and a new subscription by minting only one unique
  - validate that out_ref must be present in the Transaction Inputs
  - validate that the redeemer only mints:
    - a single CIP68 compliant Merchant Token
    - a single CIP68 compliant Subscriber Token
- **Burn:**
- validate that the redeemer only burns:
  - a single CIP68 compliant Merchant Token
  - a single CIP68 compliant Subscriber Token

#### 3.3.2.3. Spend Purpose

##### 3.3.2.3.1. Datum

##### 3.3.2.3.2. Redeemer

- Update
- Remove

###### 3.3.2.3.2.1. Validation

- **Update:** updates the metadata of the Reference NFT token and sends the token back to the address entity executing it.
- **Remove:** removes all the Reference NFT tokens to another external address.