ANASTASIA LABS

Proof of Achievement - Milestone 1

Payment Subscription Smart Contract

Project Number 1100025
Project Manager Jonathan Rodriguez

Contents

Payment Subscription Smart Contract

1. Introduction

This document presents comprehensive evidence of the successful implementation and testing of the Payment Subscription Smart Contract addressing the effortless management of recurring payments

Each section provides detailed insights into the functionality, security, and usability of the smart contract, demonstrating its readiness for real-world application.

Our rigorous testing suite demonstrates the contract's ability to manage recurring payments effectively and with ease.

2. Test Suite Details

The test suite for the Payment Subscription Smart Contract consists of thirteen critical test cases, each designed to verify specific aspects of the contract's functionality.

2.1. Test Execution Results

```
Testing ...

┌ payment_subscription/tests/account_multi_validator ──────
PASS [mem: 347249, cpu: 137903361] succeed_create_account
PASS [mem: 206854, cpu:  79875639] succeed_delete_account
PASS [mem: 477264, cpu: 179987133] succeed_update_account
PASS [mem: 289679, cpu: 112928610] succeed_remove_account
└──────────────────────────────── 4 tests | 4 passed | 0 failed

┌ payment_subscription/tests/payment_multi_validator ──────────
PASS [mem: 719170, cpu: 277679281] succeed_initiate_subscription
PASS [mem: 358259, cpu: 134424664] succeed_terminate_subscription
PASS [mem: 886708, cpu: 338587008] succeed_extend_subscription
PASS [mem: 712513, cpu: 273599426] succeed_unsubscribe
PASS [mem: 765639, cpu: 289918106] succeed_merchant_withdraw
PASS [mem: 598455, cpu: 229728929] succeed_subscriber_withdraw
└──────────────────────────────────── 6 tests | 6 passed | 0 failed

┌ payment_subscription/tests/service_multi_validator ──────
PASS [mem: 416801, cpu: 163250864] success_create_service
PASS [mem: 560773, cpu: 210655896] success_update_service
PASS [mem: 596384, cpu: 230611796] success_remove_service
└──────────────────────────────── 3 tests | 3 passed | 0 failed

Summary 13 checks, 0 errors, 0 warnings
```

Figure 1: All Payment Subscription Tests

This test validates the contract's ability to initiate a new subscription. It demonstrates:

- Correct setup of subscription parameters
- Proper creation of the Payment Datum
- Accurate handling of inputs and outputs
- Successful minting of the Payment NFT

3. Managing Recurring Payments Tests

This process comprises of six checks:

- succeed_initiate_subscription
- succeed_terminate_subscription
- succeed_extend_subscription
- succeed_unsubscribe
- succeed_merchant_withdraw
- succeed_subscriber_withdraw

3.1. Test Case: Initiating a Subscription (succeed_initiate_subscription)

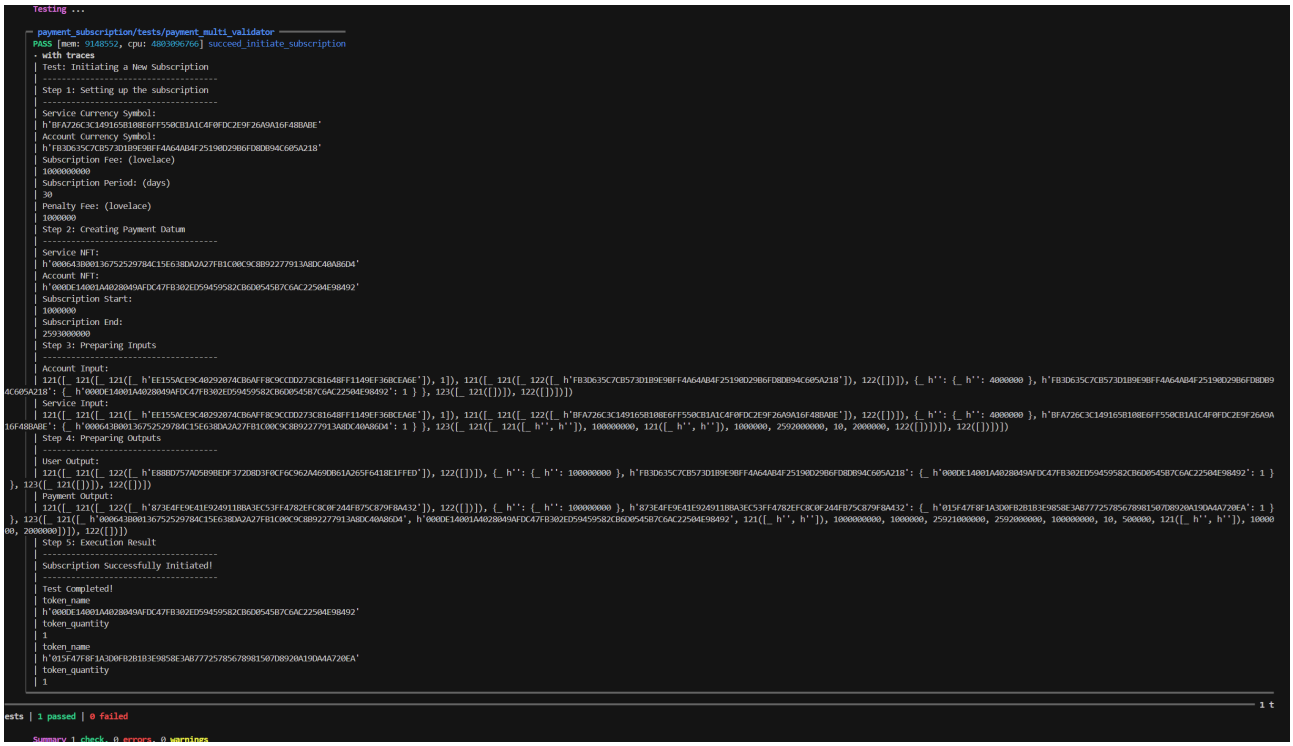

Figure 2: Succeed Initialize Subscription Test

This test validates the contract's ability to initiate a new subscription. It demonstrates:

- Correct setup of subscription parameters
- Proper creation of the Payment Datum
- Accurate handling of inputs and outputs
- Successful minting of the Payment NFT

3.2. Test Case: Terminate Subscription (succeed_terminate_subscription)

Figure 3: Succeed Terminate Subscription Test

This test verifies the contract's ability to handle early termination, applying appropriate refunds and penalties.

3.3. Test Case: Extend Subscription (succeed_extend_subscription)

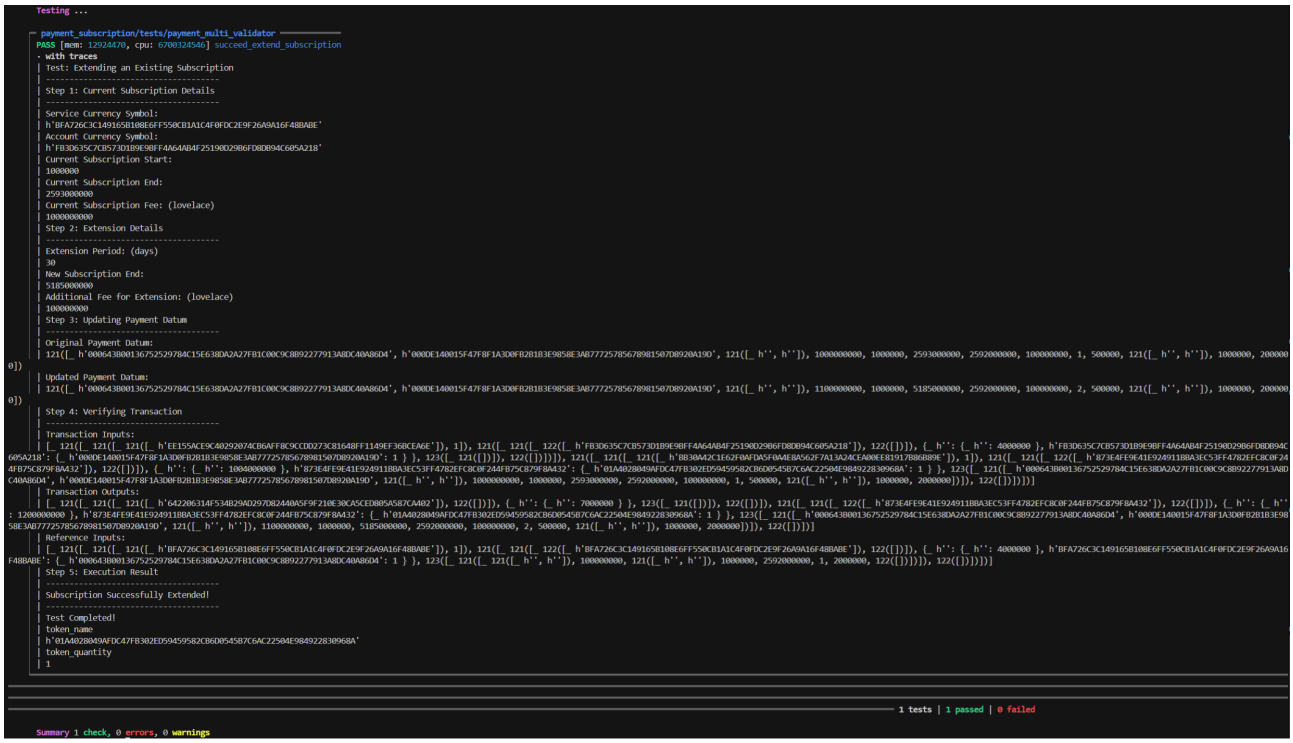

Figure 4: Succeed Extend Subscription Test

This test demonstrates the contract's ability to extend an existing subscription, showcasing the flexibility offered to subscribers. It shows:

- Accurate calculation of the new subscription end date
- Correct fee adjustment for the extension
- Proper updating of the Payment Datum
- Successful execution of the extension transaction

3.4. Test Case: Unsubscribe (succeed_unsubscribe)

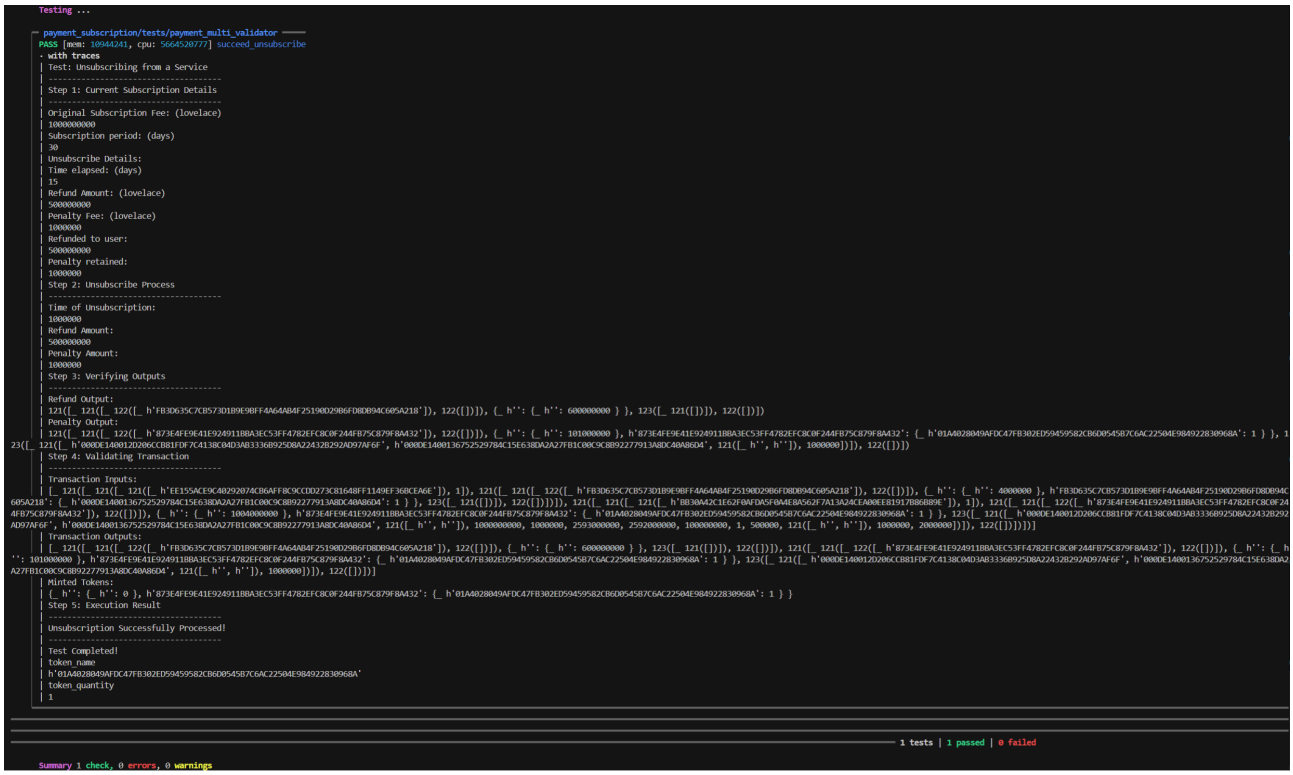

Figure 5: Succeed Unsubscribe Test

This test verifies the contract's ability to process an unsubscription. It demonstrates:

- Accurate calculation of refund and penalty amounts
- Proper distribution of funds (refund to subscriber, penalty to designated UTxO)
- Correct burning of the Payment NFT

3.5. Test Case: Withdrawing Subscription Fees by Merchant (succeed_merchant_withdraw)

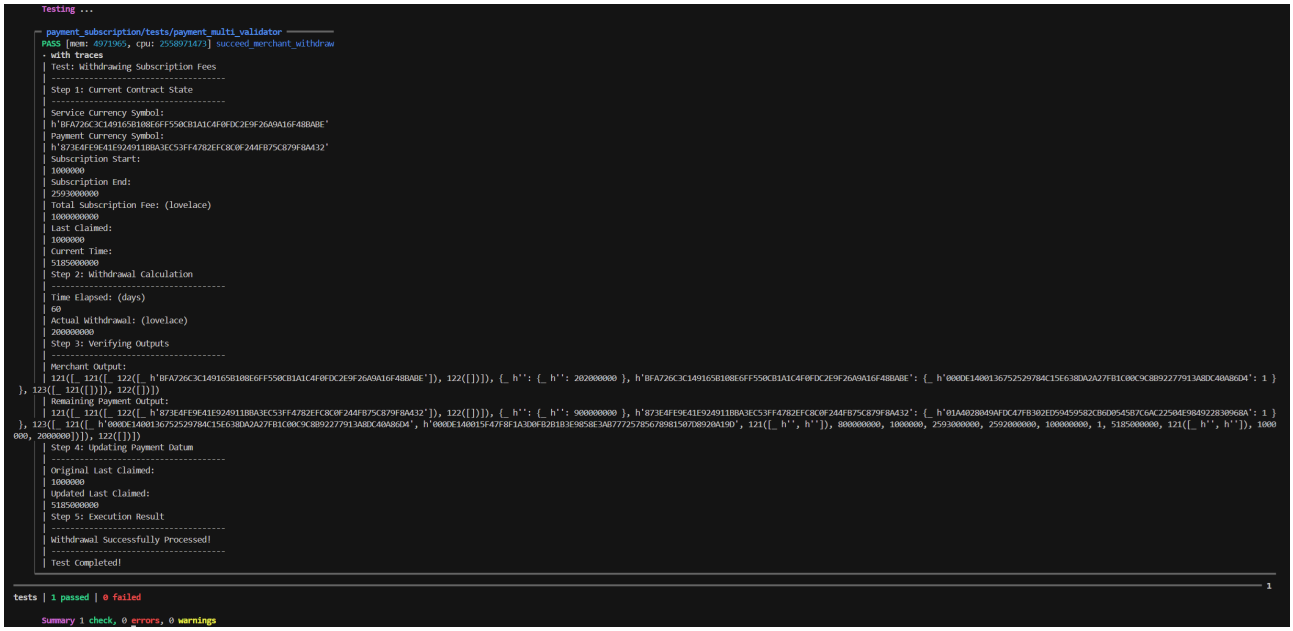

Figure 6: Succeed Unsubscribe Test

This test confirms the contract's ability to process withdrawals of subscription fees by a merchant. It shows:

- Correct calculation of withdrawable amounts based on elapsed time
- Proper distribution of funds to the merchant
- Accurate updating of the Payment Datum with new 'last claimed' time

3.6. Test Case: Withdrawing Subscription Fees by Subscriber (succeed_subscriber_withdraw)

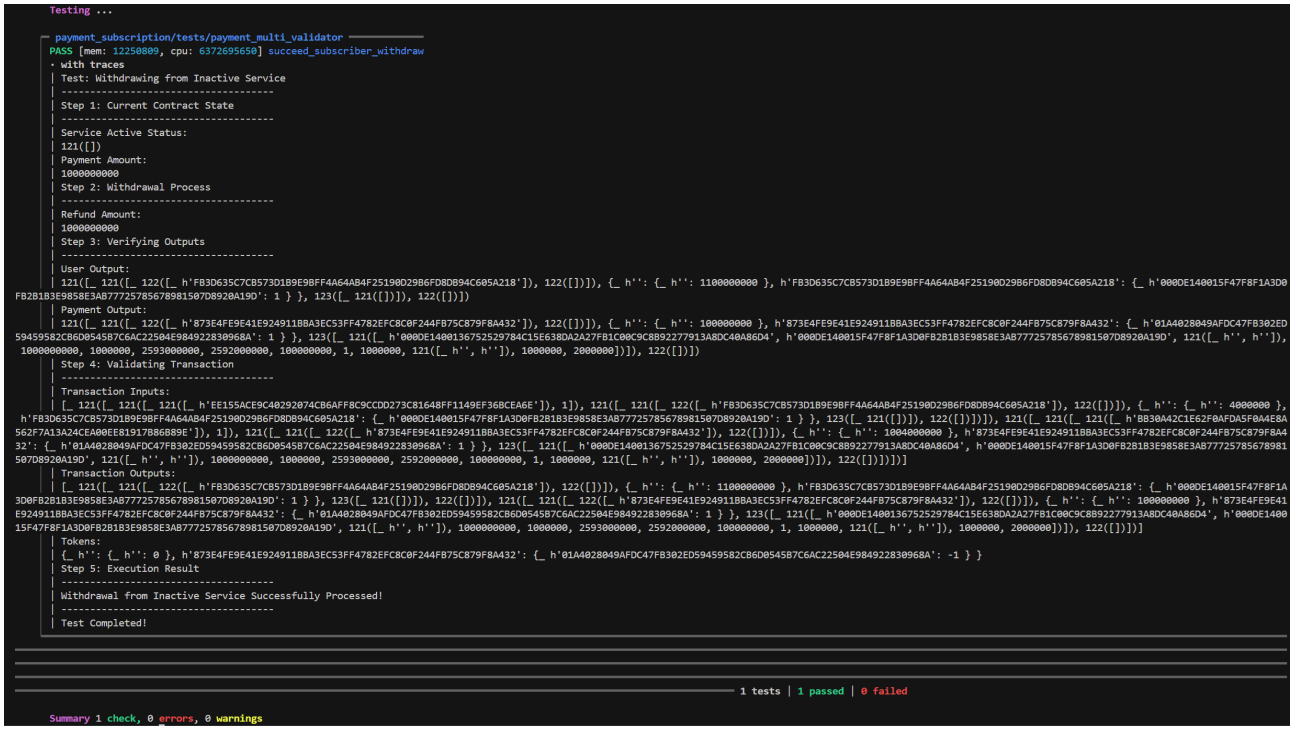

Figure 7: Succeed Unsubscribe Test

This test verifies the contract's ability to process withdrawals of subscription fees by a subscriber when the service becomes inactive. It demonstrates:

- Correct identification of an inactive service
- Full refund of the subscription amount to the subscriber
- Proper burning of the Payment NFT
- Accurate updating of the Payment UTxO

4. User Workflow for Managing Recurring Payments

The following outlines the user workflow for managing recurring payments:

1. **Initiate Subscription:**
   - User selects a service and subscription period
   - Smart contract mints a Payment NFT and locks the subscription fee
   - User receives confirmation of successful subscription
2. **Extend Subscription:**
   - User chooses to extend their subscription
   - Smart contract calculates additional fee and new end date
   - User approves the extension
   - Contract updates the Payment Datum with new details
3. **Unsubscribe:**
   - User requests to end their subscription
   - Contract calculates refund and penalty amounts
   - User receives refund, minus any applicable penalties
   - Payment NFT is burned, ending the subscription
4. **Merchant Withdrawal**
   - Merchant can withdraw accrued fees at any time
   - Contract calculates withdrawable amount based on elapsed time
   - Remaining funds stay locked until the next withdrawal or end of subscription
5. **Subscriber Withdrawal**

- Subscriber can withdraw remaining funds if the service becomes inactive
- Contract verifies the inactive status of the service
- Full remaining subscription amount is refunded to the subscriber
- Payment NFT is burned, finalizing the withdrawal

This workflow demonstrates the ease with which users can manage their recurring payments, from initiation to termination, directly from their wallets.

5. Conclusion

The Payment Subscription Smart Contract demonstrates robust functionality and ease of use. Through comprehensive testing and thoughtful implementation, it effectively manages recurring payments, allowing users to initiate, extend, and terminate subscriptions directly from their preferred wallet applications.

These features collectively ensure that the contract meets the needs of both service providers and subscribers, offering a secure and user-friendly solution for managing subscription-based services on the Cardano blockchain.