ANASTASIA LABS

Proof of Achievement - Milestone 1

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

Payment Subscription Smart Contract

1. Introduction

The Payment Subscription Smart Contract project aims to develop a robust and user-friendly system for managing recurring payments on the Cardano blockchain. This contract enables users to effortlessly set up, manage, and cancel recurring payments directly from their wallets, providing a seamless experience for both subscribers and merchants.

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

This report demonstrates our progress in implementing the contract and meeting the acceptance criteria, focusing on effortless management of recurring payments and integration with wallets.

2. Contract Functionality

The Payment Subscription Smart Contract consists of three main validators:

2.1. Service Contract

A multi-validator responsible for:

- Creating an initial service by minting a single CIP-68 compliant Service NFT asset
- Sending the Service NFT to the user and the reference NFT to the spending endpoint
- Updating the metadata for the user
- Deleting the service by setting it to inactive.

2.2. Account Contract

A multi-validator responsible for:

- Creating an account for the user by minting a CIP-68 compliant Account NFT asset
- Sending the Account NFT to the user and the reference NFT to the spending endpoint
- Updating the metadata for the account
- Deleting the user account by burning the Account NFT

2.3. Payment Contract

This is the core validator and is responsible for:

- Holding prepaid subscription fees for a service
- Renewing a subscription
- Unsubscribing from a service
- Withdrawing subscription fees

The contract incorporates a linear vesting mechanism to gradually release subscription fees to the merchant over the subscription period.

3. Effortlessly Manage Recurring Payments

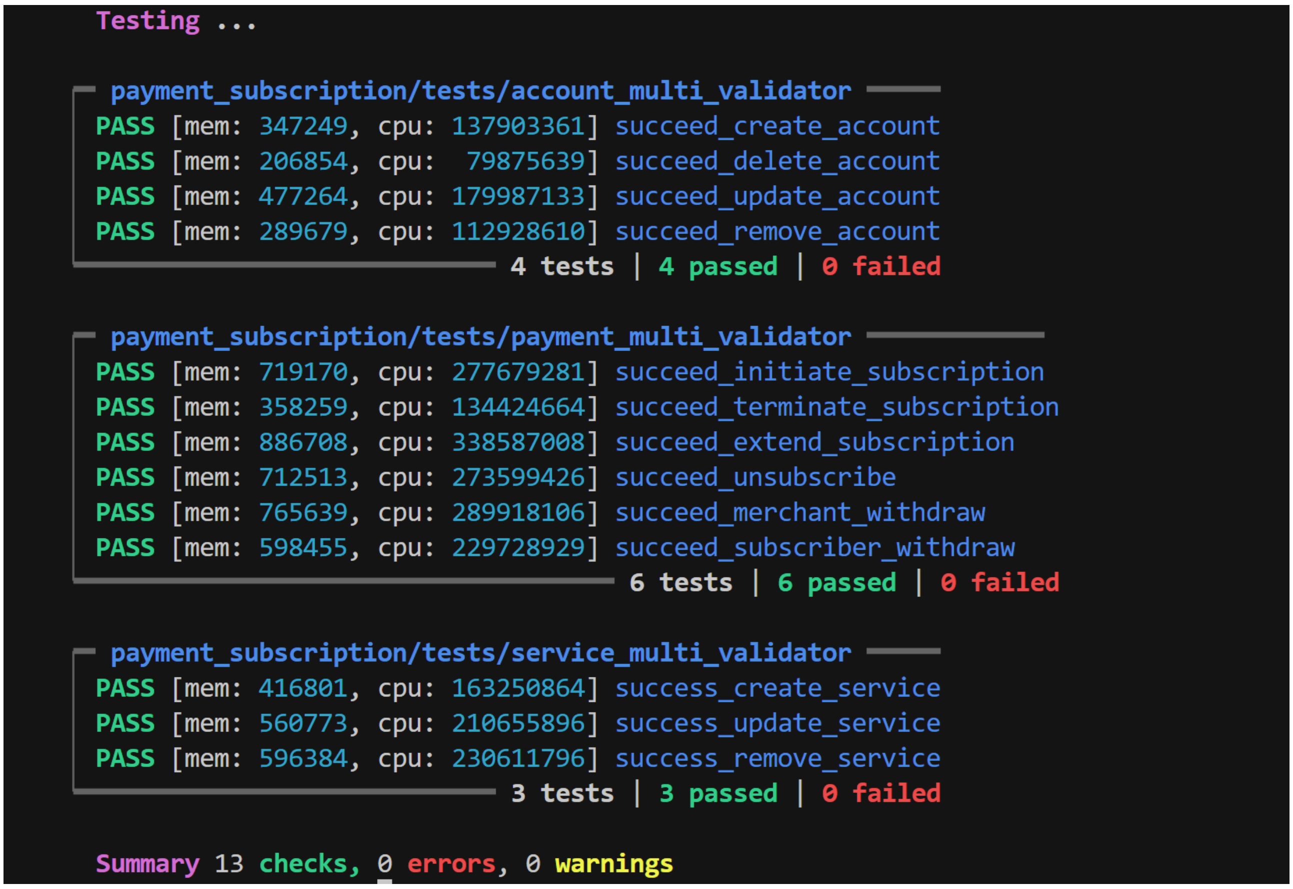
Our smart contract enables users to easily manage their recurring payments through a series of intuitive operations. We've implemented and tested various scenarios to ensure a smooth user experience.

3.1. Test Suite Details

We've developed a comprehensive test suite consisting of thirteen critical test cases, to validate the contract's functionality. These tests cover all aspects of subscription management, from initiation to termination and fund withdrawal.

Here's an overview of the test execution results:

3.2. 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: Payment Subscription Tests Overview

4. Detailed Test Case Scenarios

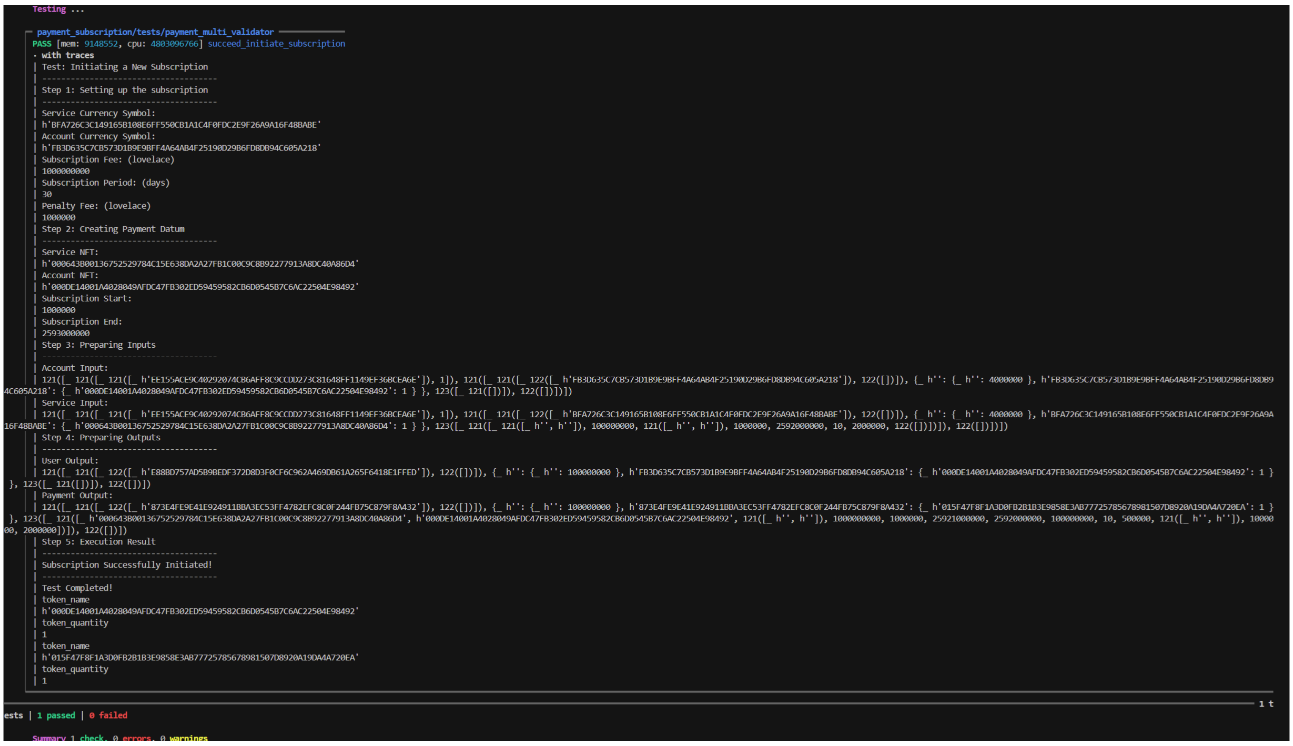
This process comprises of six checks all from the Payments Contract which are:

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

4.1. Initiating a Subscription (succeed_initiate_subscription)

Test Case Code: succeed_initiate_subscription

This test includes setting up a subscription with payment, creating the required datum and redeemer, and ensuring that the subscription is correctly recorded in the contract. It involves preparing inputs and outputs for the transaction and confirming that the subscription is successfully initiated.



```
    Testing ...

  ┍━ payment_subscription/tests/payment_multi_validator ━━━━━━━━
    PASS [mem: 9148552, cpu: 4803096766] succeed_initiate_subscription
    · with traces
    | Test: Initiating a New Subscription
    | ------------------------------------
    | Step 1: Setting up the subscription
    | ------------------------------------
    | Service Currency Symbol:
    | h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE'
    | Account Currency Symbol:
    | h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218'
    | Subscription Fee: (lovelace)
    | 1000000000
    | Subscription Period: (days)
    | 30
    | Penalty Fee: (lovelace)
    | 1000000
    | Step 2: Creating Payment Datum
    | ------------------------------------
    | Service NFT:
    | h'000643B00136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4'
    | Account NFT:
    | h'000DE14001A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E98492'
    | Subscription Start:
    | 1000000
    | Subscription End:
    | 2593000000
    | Step 3: Preparing Inputs
    | ------------------------------------
    | Account Input:
    | 121([_ 121([_ 121([_ h'EE155ACE9C40292074CB6AFF8C9CCDD273C81648FF1149EF36BCEA6E']), 1]), 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 4000000 }, h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB9
4C605A218': {_ h'000DE14001A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E98492': 1 } }, 123([_ 121([])]), 122([])])])
    | Service Input:
    | 121([_ 121([_ 121([_ h'EE155ACE9C40292074CB6AFF8C9CCDD273C81648FF1149EF36BCEA6E']), 1]), 121([_ 121([_ 122([_ h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE']), 122([])]), {_ h'': {_ h'': 4000000 }, h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A
16F48BABE': {_ h'000643B00136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4': 1 } }, 123([_ 121([_ 121([_ h'', h'']), 100000000, 121([_ h'', h'']), 1000000, 2592000000, 10, 2000000, 122([])])]), 122([])])])
    | Step 4: Preparing Outputs
    | ------------------------------------
    | User Output:
    | 121([_ 121([_ 122([_ h'E88BD757AD5B9BEDF372D8D3F0CF6C962A469DB61A265F6418E1FFED']), 122([])]), {_ h'': {_ h'': 100000000 }, h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218': {_ h'000DE14001A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E98492': 1 }
 }, 123([_ 121([])]), 122([])])
    | Payment Output:
    | 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 100000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19DA4A720EA': 1 }
 }, 123([_ 121([_ h'000643B00136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', h'000DE14001A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E98492', 121([_ h'', h'']), 1000000000, 1000000, 25921000000, 2592000000, 100000000, 10, 500000, 121([_ h'', h'']), 10000
00, 2000000])]), 122([])])
    | Step 5: Execution Result
    | ------------------------------------
    | Subscription Successfully Initiated!
    | ------------------------------------
    | Test Completed!
    | token_name
    | h'000DE14001A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E98492'
    | token_quantity
    | 1
    | token_name
    | h'015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19DA4A720EA'
    | token_quantity
    | 1
  ┕━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1 t
ests | 1 passed | 0 failed

    Summary 1 check, 0 errors, 0 warnings
```

Figure 2: Succeed Initialize Subscription Test

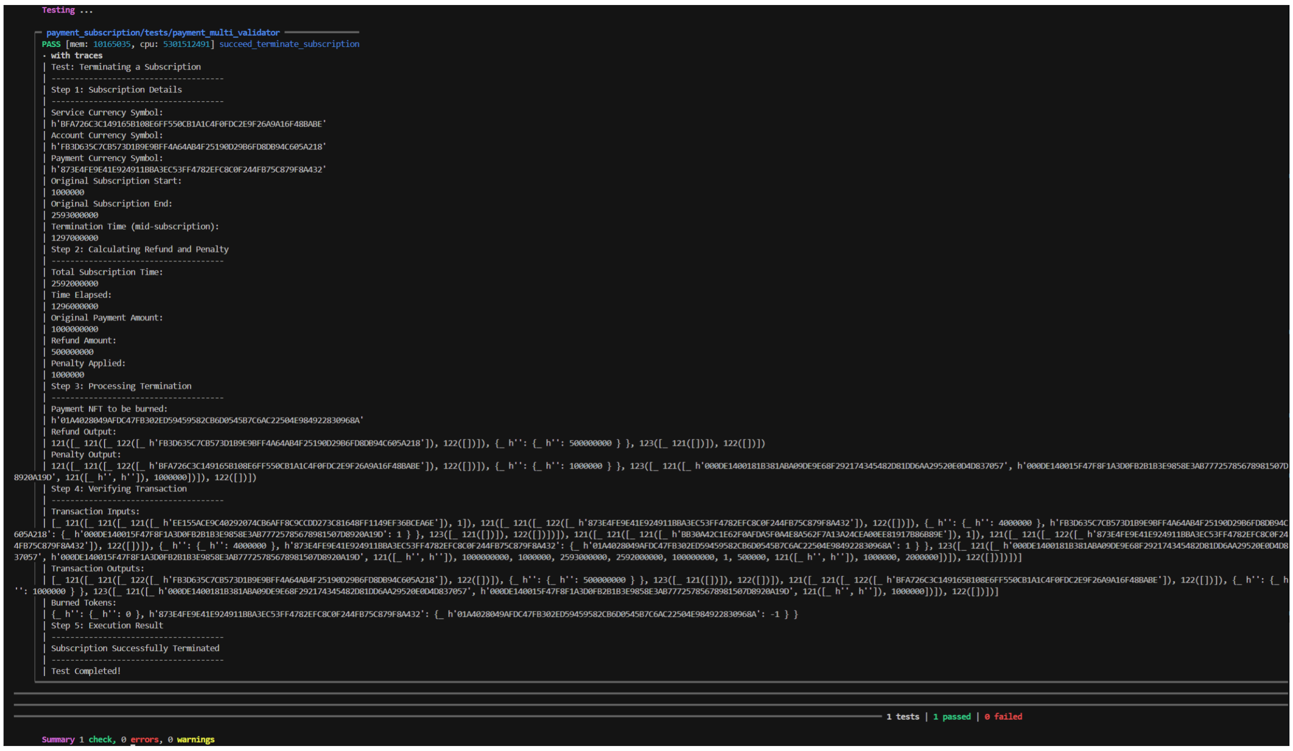
This test validates the contract’s ability to initiate a new subscription. It demonstrates:

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

4.2. Terminate Subscription (succeed_terminate_subscription)

Test Case Code: succeed_terminate_subscription

This scenario covers the case where a subscription is terminated before its end date. It involves calculating and distributing the refund and penalty fees based on the elapsed subscription time and ensuring that the contract correctly reflects the termination and payment of fees.



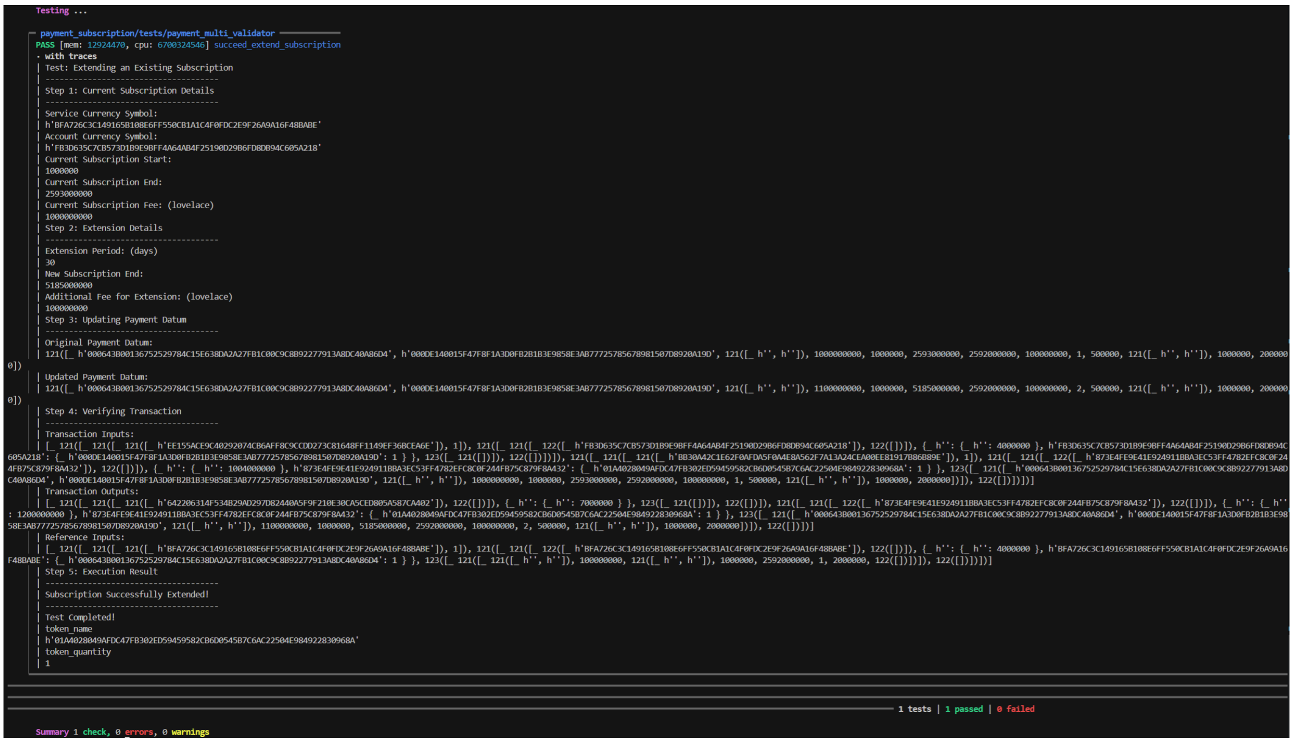
```
Testing ...

┍━ payment_subscription/tests/payment_multi_validator ━━━━━━━━━━
│ PASS [mem: 10165035, cpu: 5301512491] succeed_terminate_subscription
│ · with traces
│ | Test: Terminating a Subscription
│ | ------------------------------------
│ | Step 1: Subscription Details
│ | ------------------------------------
│ | Service Currency Symbol:
│ | h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE'
│ | Account Currency Symbol:
│ | h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218'
│ | Payment Currency Symbol:
│ | h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432'
│ | Original Subscription Start:
│ | 1000000
│ | Original Subscription End:
│ | 2593000000
│ | Termination Time (mid-subscription):
│ | 1297000000
│ | Step 2: Calculating Refund and Penalty
│ | ------------------------------------
│ | Total Subscription Time:
│ | 2592000000
│ | Time Elapsed:
│ | 1296000000
│ | Original Payment Amount:
│ | 1000000000
│ | Refund Amount:
│ | 500000000
│ | Penalty Applied:
│ | 1000000
│ | Step 3: Processing Termination
│ | ------------------------------------
│ | Payment NFT to be burned:
│ | h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A'
│ | Refund Output:
│ | 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 500000000 } }, 123([_ 121([])]), 122([])])
│ | Penalty Output:
│ | 121([_ 121([_ 122([_ h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE']), 122([])]), {_ h'': {_ h'': 1000000 } }, 123([_ 121([_ h'000DE1400181B381ABA09DE9E68F292174345482D81DD6AA29520E0D4D837057', h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D
8920A19D', 121([_ h'', h'']), 1000000])]), 122([])])
│ | Step 4: Verifying Transaction
│ | ------------------------------------
│ | Transaction Inputs:
│ | [_ 121([_ 121([_ 121([_ h'EE155ACE9C40292074CB6AFF8C9CCDD273C81648FF1149EF36BCEA6E']), 1]), 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 4000000 }, h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C
605A218': {_ h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D': 1 } }, 123([_ 121([])]), 122([])])]), 121([_ 121([_ 121([_ h'BB30A42C1E62F0AFDA5F0A4E8A562F7A13A24CEA00EE81917B86B89E']), 1]), 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F24
4FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 4000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 } }, 123([_ 121([_ h'000DE1400181B381ABA09DE9E68F292174345482D81DD6AA29520E0D4D8
37057', h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D', 121([_ h'', h'']), 1000000000, 1000000, 2593000000, 2592000000, 100000000, 1, 500000, 121([_ h'', h'']), 1000000, 2000000])]), 122([])])])]
│ | Transaction Outputs:
│ | [_ 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 500000000 } }, 123([_ 121([])]), 122([])]), 121([_ 121([_ 122([_ h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE']), 122([])]), {_ h'': {_ h
'': 1000000 } }, 123([_ 121([_ h'000DE1400181B381ABA09DE9E68F292174345482D81DD6AA29520E0D4D837057', h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D', 121([_ h'', h'']), 1000000])]), 122([])])]
│ | Burned Tokens:
│ | {_ h'': {_ h'': 0 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': -1 } }
│ | Step 5: Execution Result
│ | ------------------------------------
│ | Subscription Successfully Terminated
│ | ------------------------------------
│ | Test Completed!
┕━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1 tests | 1 passed | 0 failed

Summary 1 check, 0 errors, 0 warnings
```

Figure 3: Succeed Terminate Subscription Test

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

4.3. Extend Subscription (succeed_extend_subscription)

Test Case Code: succeed_extend_subscription



```
    Testing ...

  ┌ payment_subscription/tests/payment_multi_validator ──────────
    PASS [mem: 12924470, cpu: 6700324546] succeed_extend_subscription
    · with traces
    | Test: Extending an Existing Subscription
    | ------------------------------------
    | Step 1: Current Subscription Details
    | ------------------------------------
    | Service Currency Symbol:
    | h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE'
    | Account Currency Symbol:
    | h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218'
    | Current Subscription Start:
    | 1000000
    | Current Subscription End:
    | 2593000000
    | Current Subscription Fee: (lovelace)
    | 1000000000
    | Step 2: Extension Details
    | ------------------------------------
    | Extension Period: (days)
    | 30
    | New Subscription End:
    | 5185000000
    | Additional Fee for Extension: (lovelace)
    | 100000000
    | Step 3: Updating Payment Datum
    | ------------------------------------
    | Original Payment Datum:
    | 121([_ h'000643B00136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D', 121([_ h'', h'']), 1000000000, 1000000, 2593000000, 2592000000, 100000000, 1, 500000, 121([_ h'', h'']), 1000000, 200000
0])
    | Updated Payment Datum:
    | 121([_ h'000643B00136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D', 121([_ h'', h'']), 1100000000, 1000000, 5185000000, 2592000000, 100000000, 2, 500000, 121([_ h'', h'']), 1000000, 200000
0])
    | Step 4: Verifying Transaction
    | ------------------------------------
    | Transaction Inputs:
    | [_ 121([_ 121([_ 121([_ h'EE155ACE9C40292074CB6AFF8C9CCDD273C81648FF1149EF36BCEA6E']), 1]), 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 4000000 }, h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C
605A218': {_ h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D': 1 } }, 123([_ 121([])]), 122([])])]), 121([_ 121([_ 121([_ h'BB30A42C1E62F0AFDA5F0A4E8A562F7A13A24CEA00EE81917B86B89E']), 1]), 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F24
4FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 1004000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 } }, 123([_ 121([_ h'000643B00136752529784C15E638DA2A27FB1C00C9C8B92277913A8D
C40A86D4', h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D', 121([_ h'', h'']), 1000000000, 1000000, 2593000000, 2592000000, 100000000, 1, 500000, 121([_ h'', h'']), 1000000, 2000000])]), 122([])])])]
    | Transaction Outputs:
    | [_ 121([_ 121([_ 121([_ h'642206314F534B29AD297D82440A5F9F210E30CA5CED805A587CA402']), 122([])]), {_ h'': {_ h'': 7000000 } }, 123([_ 121([])]), 122([])]), 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432']), 122([])]), {_ h'': {_ h''
: 1200000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 } }, 123([_ 121([_ h'000643B00136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', h'000DE140015F47F8F1A3D0FB2B1B3E98
58E3AB77725785678981507D8920A19D', 121([_ h'', h'']), 1100000000, 1000000, 5185000000, 2592000000, 100000000, 2, 500000, 121([_ h'', h'']), 1000000, 2000000])]), 122([])])]
    | Reference Inputs:
    | [_ 121([_ 121([_ 121([_ h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE']), 1]), 121([_ 121([_ 122([_ h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE']), 122([])]), {_ h'': {_ h'': 4000000 }, h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16
F48BABE': {_ h'000643B00136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4': 1 } }, 123([_ 121([_ 121([_ h'', h'']), 100000000, 121([_ h'', h'']), 1000000, 2592000000, 1, 2000000, 122([])])]), 122([])])])]
    | Step 5: Execution Result
    | ------------------------------------
    | Subscription Successfully Extended!
    | ------------------------------------
    | Test Completed!
    | token_name
    | h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A'
    | token_quantity
    | 1
  └──────────────────────────────────────────────────────────────

  ══════════════════════════════════════════════════ 1 tests | 1 passed | 0 failed

    Summary 1 check, 0 errors, 0 warnings
```

Figure 4: Succeed Extend Subscription Test

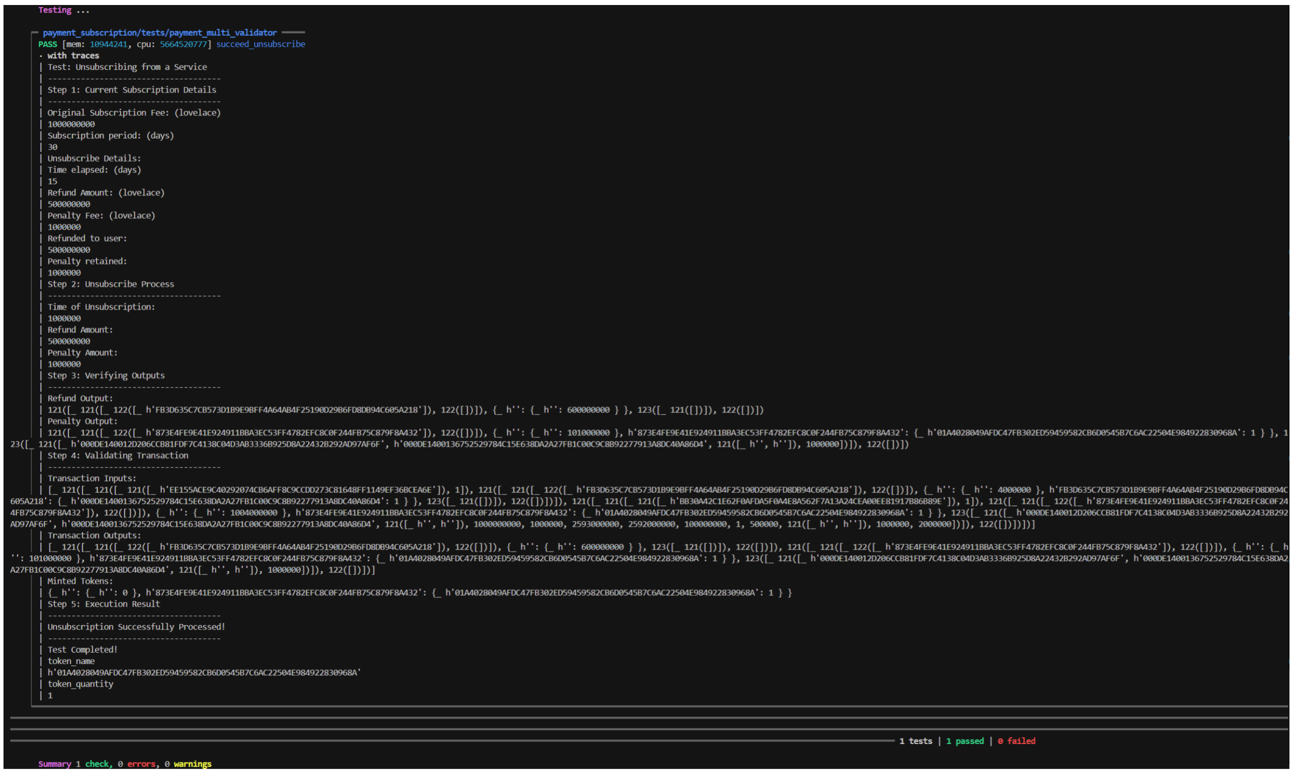
This test demonstrates the contract's ability to extend an existing subscription, showcasing the flexibility offered to subscribers. It shows:

- 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

4.4. Unsubscribe (succeed_unsubscribe)

Test Case Code: succeed_unsubscribe

This test covers the scenario where a user unsubscribes before the end of the subscription period, including handling any remaining funds, refunds, or penalties. It verifies that the contract correctly processes the unsubscription and updates the state to reflect the change.



```
    Testing ...

  ┌ payment_subscription/tests/payment_multi_validator ────
    PASS [mem: 10944241, cpu: 5664520777] succeed_unsubscribe
    · with traces
    | Test: Unsubscribing from a Service
    | ------------------------------------
    | Step 1: Current Subscription Details
    | ------------------------------------
    | Original Subscription Fee: (lovelace)
    | 1000000000
    | Subscription period: (days)
    | 30
    | Unsubscribe Details:
    | Time elapsed: (days)
    | 15
    | Refund Amount: (lovelace)
    | 500000000
    | Penalty Fee: (lovelace)
    | 1000000
    | Refunded to user:
    | 500000000
    | Penalty retained:
    | 1000000
    | Step 2: Unsubscribe Process
    | ------------------------------------
    | Time of Unsubscription:
    | 1000000
    | Refund Amount:
    | 500000000
    | Penalty Amount:
    | 1000000
    | Step 3: Verifying Outputs
    | ------------------------------------
    | Refund Output:
    | 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 600000000 } }, 123([_ 121([])]), 122([])])
    | Penalty Output:
    | 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 101000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 } }, 1
23([_ 121([_ h'000DE140012D206CCB81FDF7C4138C04D3AB3336B925D8A22432B292AD97AF6F', h'000DE1400136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', 121([_ h'', h'']), 1000000])]), 122([])])
    | Step 4: Validating Transaction
    | ------------------------------------
    | Transaction Inputs:
    | [_ 121([_ 121([_ 121([_ h'EE155ACE9C40292074CB6AFF8C9CCDD273C81648FF1149EF36BCEA6E']), 1]), 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 4000000 }, h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C
605A218': {_ h'000DE1400136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4': 1 } }, 123([_ 121([])]), 122([])])]), 121([_ 121([_ 121([_ h'BB30A42C1E62F0AFDA5F0A4E8A562F7A13A24CEA00EE81917B86B89E']), 1]), 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F24
4FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 1004000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 } }, 123([_ 121([_ h'000DE140012D206CCB81FDF7C4138C04D3AB3336B925D8A22432B292
AD97AF6F', h'000DE1400136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', 121([_ h'', h'']), 1000000000, 1000000, 2593000000, 2592000000, 100000000, 1, 500000, 121([_ h'', h'']), 1000000, 2000000])]), 122([])])])]
    | Transaction Outputs:
    | [_ 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 600000000 } }, 123([_ 121([])]), 122([])]), 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432']), 122([])]), {_ h'': {_ h
'': 101000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 } }, 123([_ 121([_ h'000DE140012D206CCB81FDF7C4138C04D3AB3336B925D8A22432B292AD97AF6F', h'000DE1400136752529784C15E638DA2
A27FB1C00C9C8B92277913A8DC40A86D4', 121([_ h'', h'']), 1000000])]), 122([])])]
    | Minted Tokens:
    | {_ h'': {_ h'': 0 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 } }
    | Step 5: Execution Result
    | ------------------------------------
    | Unsubscription Successfully Processed!
    | ------------------------------------
    | Test Completed!
    | token_name
    | h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A'
    | token_quantity
    | 1

  ──────────────────────────────────────────────────────── 1 tests | 1 passed | 0 failed

    Summary 1 check, 0 errors, 0 warnings
```

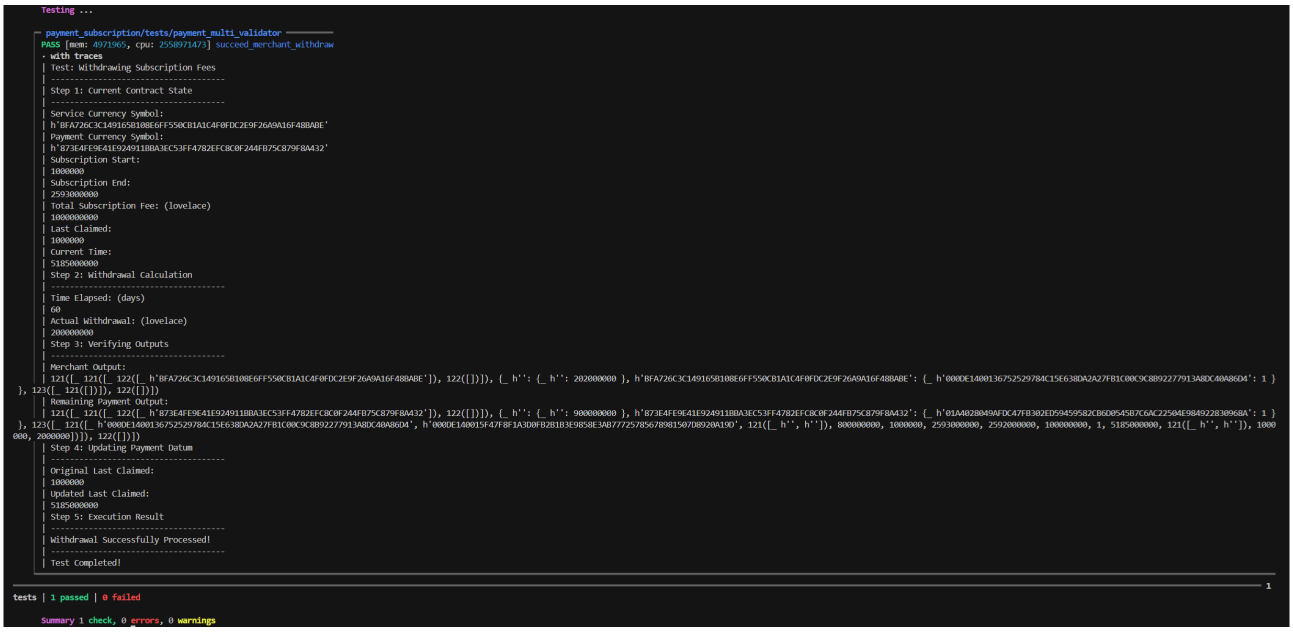
Figure 5: Succeed Unsubscribe Test

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

4.5. Merchant Withdrawing Fees (succeed_merchant_withdraw)

Test Case Code: succeed_merchant_withdraw



```
    Testing ...

  ┌ payment_subscription/tests/payment_multi_validator ──────
    PASS [mem: 4971965, cpu: 2558971473] succeed_merchant_withdraw
    · with traces
    | Test: Withdrawing Subscription Fees
    | ------------------------------------
    | Step 1: Current Contract State
    | ------------------------------------
    | Service Currency Symbol:
    | h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE'
    | Payment Currency Symbol:
    | h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432'
    | Subscription Start:
    | 1000000
    | Subscription End:
    | 2593000000
    | Total Subscription Fee: (lovelace)
    | 1000000000
    | Last Claimed:
    | 1000000
    | Current Time:
    | 5185000000
    | Step 2: Withdrawal Calculation
    | ------------------------------------
    | Time Elapsed: (days)
    | 60
    | Actual Withdrawal: (lovelace)
    | 200000000
    | Step 3: Verifying Outputs
    | ------------------------------------
    | Merchant Output:
    | 121([_ 121([_ 122([_ h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE']), 122([])]), {_ h'': {_ h'': 202000000 }, h'BFA726C3C149165B108E6FF550CB1A1C4F0FDC2E9F26A9A16F48BABE': {_ h'000DE1400136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4': 1 }
 }, 123([_ 121([])]), 122([])])
    | Remaining Payment Output:
    | 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 900000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 }
 }, 123([_ 121([_ h'000DE1400136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D', 121([_ h'', h'']), 800000000, 1000000, 2593000000, 2592000000, 100000000, 1, 5185000000, 121([_ h'', h'']), 1000
000, 2000000])]), 122([])])
    | Step 4: Updating Payment Datum
    | ------------------------------------
    | Original Last Claimed:
    | 1000000
    | Updated Last Claimed:
    | 5185000000
    | Step 5: Execution Result
    | ------------------------------------
    | Withdrawal Successfully Processed!
    | ------------------------------------
    | Test Completed!
  └──────────────────────────────────────────────────────── 1

  tests | 1 passed | 0 failed

    Summary 1 check, 0 errors, 0 warnings
```

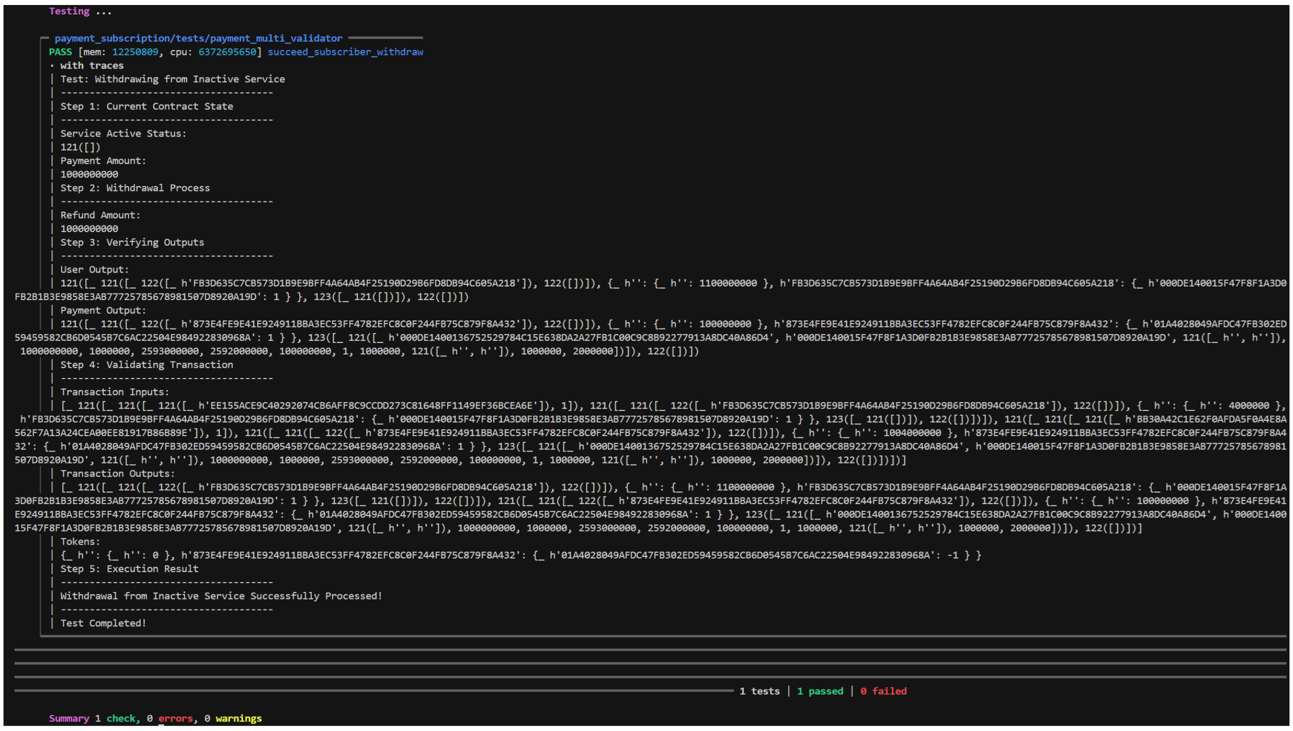
Figure 6: Succeed Merchant Withdraw Test

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

4.6. Subscriber Withdrawing Fees (succeed_subscriber_withdraw)

Test Case Code: succeed_subscriber_withdraw



```
    Testing ...

  ┌ payment_subscription/tests/payment_multi_validator ──────
  │ PASS [mem: 12250809, cpu: 6372695650] succeed_subscriber_withdraw
  │ · with traces
  │ | Test: Withdrawing from Inactive Service
  │ | ------------------------------------
  │ | Step 1: Current Contract State
  │ | ------------------------------------
  │ | Service Active Status:
  │ | 121([])
  │ | Payment Amount:
  │ | 1000000000
  │ | Step 2: Withdrawal Process
  │ | ------------------------------------
  │ | Refund Amount:
  │ | 1000000000
  │ | Step 3: Verifying Outputs
  │ | ------------------------------------
  │ | User Output:
  │ | 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 1100000000 }, h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218': {_ h'000DE140015F47F8F1A3D0
FB2B1B3E9858E3AB77725785678981507D8920A19D': 1 } }, 123([_ 121([])]), 122([])])
  │ | Payment Output:
  │ | 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 100000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED
59459582CB6D0545B7C6AC22504E984922830968A': 1 } }, 123([_ 121([_ h'000DE1400136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D', 121([_ h'', h'']),
 1000000000, 1000000, 2593000000, 2592000000, 100000000, 1, 1000000, 121([_ h'', h'']), 1000000, 2000000])]), 122([])])
  │ | Step 4: Validating Transaction
  │ | ------------------------------------
  │ | Transaction Inputs:
  │ | [_ 121([_ 121([_ 121([_ h'EE155ACE9C40292074CB6AFF8C9CCDD273C81648FF1149EF36BCEA6E']), 1]), 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 4000000 },
 h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218': {_ h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D': 1 } }, 123([_ 121([])]), 122([])])]), 121([_ 121([_ 121([_ h'BB30A42C1E62F0AFDA5F0A4E8A
562F7A13A24CEA00EE81917B86B89E']), 1]), 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 1004000000 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A4
32': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 } }, 123([_ 121([_ h'000DE1400136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', h'000DE140015F47F8F1A3D0FB2B1B3E9858E3AB77725785678981
507D8920A19D', 121([_ h'', h'']), 1000000000, 1000000, 2593000000, 2592000000, 100000000, 1, 1000000, 121([_ h'', h'']), 1000000, 2000000])]), 122([])])])]
  │ | Transaction Outputs:
  │ | [_ 121([_ 121([_ 122([_ h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218']), 122([])]), {_ h'': {_ h'': 1100000000 }, h'FB3D635C7CB573D1B9E9BFF4A64AB4F25190D29B6FD8DB94C605A218': {_ h'000DE140015F47F8F1A
3D0FB2B1B3E9858E3AB77725785678981507D8920A19D': 1 } }, 123([_ 121([])]), 122([])]), 121([_ 121([_ 122([_ h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432']), 122([])]), {_ h'': {_ h'': 100000000 }, h'873E4FE9E41
E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': 1 } }, 123([_ 121([_ h'000DE1400136752529784C15E638DA2A27FB1C00C9C8B92277913A8DC40A86D4', h'000DE1400
15F47F8F1A3D0FB2B1B3E9858E3AB77725785678981507D8920A19D', 121([_ h'', h'']), 1000000000, 1000000, 2593000000, 2592000000, 100000000, 1, 1000000, 121([_ h'', h'']), 1000000, 2000000])]), 122([])])]
  │ | Tokens:
  │ | {_ h'': {_ h'': 0 }, h'873E4FE9E41E924911BBA3EC53FF4782EFC8C0F244FB75C879F8A432': {_ h'01A4028049AFDC47FB302ED59459582CB6D0545B7C6AC22504E984922830968A': -1 } }
  │ | Step 5: Execution Result
  │ | ------------------------------------
  │ | Withdrawal from Inactive Service Successfully Processed!
  │ | ------------------------------------
  │ | Test Completed!
  └──────────────────────────────────────────────────────────

────────────────────────────────────────────────────────────
────────────────────────────────────────────────────────────
────────────────────────────────────────────────────────────
──────────────────────────────── 1 tests | 1 passed | 0 failed

    Summary 1 check, 0 errors, 0 warnings
```

Figure 7: Succeed Subscriber Withdraw 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

5. 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.

6. 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 efficiently.

Key achievements include:

- Successful implementation of subscription initiation, extension, and termination processes
- Accurate handling of fee calculations, including prorated refunds and penalties
- Secure management of funds through the payment contract
- Flexible service and account management through dedicated contracts

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.

All documentation, including detailed test cases and explanations, is available in our Payment Subscription Github Repo