



ANASTASIA LABS

Payment Subscription Smart Contract

Design Specification

Contents

1. Introduction	1
2. Test Suite Details	2
2.1. Test Execution Results	2
3. Managing Recurring Payments Tests	3
3.1. Test Case: Initiating a Subscription (succeed_initiate_subscription)	3
3.2. Test Case: Terminate Subscription (succeed_terminate_subscription)	4
3.3. Test Case: Extend Subscription (succeed_extend_subscription)	5
3.4. Test Case: Unsubscribe (succeed_unsubscribe)	6
3.5. Test Case: Withdrawing Subscription Fees by Merchant (succeed_merchant_withdraw)	7
3.6. Test Case: Withdrawing Subscription Fees by Subscriber (succeed_subscriber_withdraw)	8
4. User Workflow for Managing Recurring Payments	8
5. Conclusion	9

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)

[illegible]

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)

```

Testing ...
Payment_subscription_tests/payment_multi_validator
PASS [mom: 10165035, cpu: 5301512491] succeed_terminate_subscription
  with traces
    Test: Terminating a Subscription
      Step 1: Subscription Details
        Service Currency Symbol:
          h'BF726C3C149165810866FF55ACB1A1CA4F8DC2E9F26A9A16F48BAE'
        Account Currency Symbol:
          h'F830635C7C857301B998F4A6A04F251902986FDB09AC685A218'
        Payment Currency Symbol:
          h'873E4F19E41924911BB3ECS3FF4782EFC80F244F875C879FBA32'
        Original Subscription Start:
          100000
        Original Subscription End:
          259300000
        Duration Time (mid-subscription):
          129700000
        Step 2: Calculating Refund and Penalty
          Total Subscription Time:
            259300000
          Time Elapsed:
            129700000
          Original Payment Amount:
            100000000
          Refund Amount:
            -50000000
          Penalty Applied:
            1000000
          Step 3: Processing Termination
            Payment left to be burned:
              h'01A028049FDC47F830E19549582C8D0545B7C62594E98492283096A'
            Refund Output:
              121([ _ 121([ _ h'F830635C7C857301B998F4A6A04F251902986FDB09AC685A218' ]), 122([ ]), [ _ h'': 500000000 ] ), 123([ _ 121([ ]), 122([ ]))
            Penalty Output:
              121([ _ 121([ _ 121([ _ h'BF726C3C149165810866FF55ACB1A1CA4F8DC2E9F26A9A16F48BAE' ]), 122([ ]), [ _ h'': 1000000 ] ), 123([ _ 121([ _ h'0000E140015F47F87BA30F02B183E985E3AB7727578567898150708920A190' ]), 122([ ]))
            Step 4: Verifying Transaction
              Transaction Inputs:
                [ _ 121([ _ 121([ _ h'EE155AC9C4029207ACB6AFFBC9CCD27381648FF1149F368CEA6' ]), 11), 121([ _ 121([ _ 121([ _ h'873E4F19E41924911BB3ECS3FF4782EFC80F244F875C879FBA32' ]), 122([ ]), [ _ h'': 4000000 ] ), h'F830635C7C857301B998F4A6A04F251902986FDB09AC685A218' ]), 122([ ]), [ _ h'': 4000000 ] ), h'873E4F19E41924911BB3ECS3FF4782EFC80F244F875C879FBA32' ] ), 123([ _ 121([ _ h'01A028049FDC47F830E19549582C8D0545B7C62594E98492283096A' ]), 11 ]), 121([ _ 121([ _ h'873E4F19E41924911BB3ECS3FF4782EFC80F244F875C879FBA32' ]), 122([ ]), [ _ h'': 1000000 ] ), 123([ _ 121([ _ h'0000E140015F47F87BA30F02B183E985E3AB7727578567898150708920A190' ]), 122([ ]))
              Transaction Outputs:
                [ _ 121([ _ 121([ _ h'F830635C7C857301B998F4A6A04F251902986FDB09AC685A218' ]), 122([ ]), [ _ h'': 500000000 ] ), 123([ _ 121([ ]), 122([ ]), 121([ _ 121([ _ h'BF726C3C149165810866FF55ACB1A1CA4F8DC2E9F26A9A16F48BAE' ]), 122([ ]), [ _ h'': [ _ h'': 1000000 ] ], 123([ _ 121([ _ h'0000E140015F47F87BA30F02B183E985E3AB7727578567898150708920A190' ]), 122([ ]), [ _ h'': 1000000 ] ), 123([ _ 121([ _ h'0000E140015F47F87BA30F02B183E985E3AB7727578567898150708920A190' ]), 122([ ]))
            Burned Tokens:
              [ _ h'': [ _ h'': 0 ] ), h'873E4F19E41924911BB3ECS3FF4782EFC80F244F875C879FBA32' ] _ h'01A028049FDC47F830E19549582C8D0545B7C62594E98492283096A' : -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.

3.3. Test Case: Extend Subscription (succeed_extend_subscription)

```

Testing ...
- payment_subscription/tests/payment_multi_validator
PASS [mem: 12924678, cpu: 6708324546] succeed_extend_subscription
- with traces
| Test: Extending an Existing Subscription
| .....
| Step 1: Current Subscription Details
| .....
| Service Currency Symbol:
| h'8fA726c3C14916581B866fF550K31A1C4F8fDC2E9f26A9A16f4BBABE'
| Account Currency Symbol:
| h'f1B30c3CDB7d0B8998f4A6A46A2F25190D2986fD8094C605A218'
| Current Subscription Start:
| 1000000
| Current Subscription End:
| 259200000
| Current Subscription Fee (lowvalue)
| 1000000000
| Step 2: Extension Details
| .....
| Extension Period: (days)
| 30
| New Subscription End:
| 518500000
| Additional Fee for Extension: (lowvalue)
| 100000000
| Step 3: Updating Payment Datum
| .....
| Original Payment Datum:
| 121([_ h'000643800136752529784C15E638DA2A2f7B1C00C9CB92277913A8C40A804', h'0000E140015f47f8f1A300f82B183E9S8E348f772578567898150708920A19D', 121([_ h'', h'']), 1000000000, 1000000, 2592000000, 2592000000, 1000000000, 1, 5000000, 121([_ h'', h'']), 1000000, 200000
|
| Updated Payment Datum:
| 121([_ h'000643800136752529784C15E638DA2A2f7B1C00C9CB92277913A8C40A804', h'0000E140015f47f8f1A300f82B183E9S8E348f772578567898150708920A19D', 121([_ h'', h'']), 1100000000, 1000000, 5185000000, 2592000000, 1000000000, 2, 5000000, 121([_ h'', h'']), 1000000, 200000
|
| Step 4: Verifying Transaction
| .....
| Transaction Inputs:
| 121([_ 121([_ 121([_ h'E6155FCAC8029267AC6A6fF8C9C02D2737B1648f11A96f3C8FAEE', 1]), 121([_ 121([_ 122([_ h'F1B03C57C7857d0B98f98f4A6A46A2F25190D2986fD8094C605A218', 122([[]]), (_ h'': _ h'': 4000000 ), h'F1B03C57C7857d0B98f98f4A6A46A2F25190D2986fD8094C605A218' ], 122([[]]), (_ h'': _ h'': 1000000000 ), h'873E4fE9E41E924911B8A1C53F4782EFC80F2A4f875C879F8A332' ], 122([[]]), (_ h'': _ h'': 1000000000 ), h'873E4fE9E41E924911B8A1C53F4782EFC80F2A4f875C879F8A332' ], 121([_ 121([_ 121([_ h'8838A42C1E62f9A4F0A5F0A8E8A562F7A13A24EFA08E1917868B89E', 1]), 121([_ 121([_ 122([_ h'873E4fE9E41E924911B8A1C53F4782EFC80F2A4f875C879F8A332' ], 122([[]]), (_ h'': _ h'': 1000000000 ), h'873E4fE9E41E924911B8A1C53F4782EFC80F2A4f875C879F8A332' ], 121([_ 121([_ 121([_ h'01A402B8A94DC47fB302E0545958C38D00545B7C6AC22504E984922830968A', 1 ] ), 123([_ 121([_ h'000643800136752529784C15E638DA2A2f7B1C00C9CB92277913A8C40A804', h'0000E140015f47f8f1A300f82B183E9S8E348f772578567898150708920A19D', 121([_ h'', h'']), 1000000000, 1000000, 2592000000, 2592000000, 1000000000, 1, 5000000, 121([_ h'', h''), 1000000, 200000000]), 122([[]]))])
| Transaction Outputs:
| 121([_ 121([_ 121([_ h'642286314f534B29A02970824A0A5f921083C5C2EDB05A387CMA02', 122([[]]), (_ h'': _ h'': 700000000 ) ], 123([_ 121([[]]), 122([[]]), 121([_ 121([_ 122([_ h'873E4fE9E41E924911B8A1C53F4782EFC80F2A4f875C879F8A332' ], 122([[]]), (_ h'': _ h'': 1200000000 ), h'873E4fE9E41E924911B8A1C53F4782EFC80F2A4f875C879F8A332' ], (_ h'8838A42C1E62f9A4F0A5F0A8E8A562F7A13A24EFA08E1917868B89E', 1 ] ), 123([_ 121([_ 121([_ h'000643800136752529784C15E638DA2A2f7B1C00C9CB92277913A8C40A804', h'0000E140015f47f8f1A300f82B183E9S8E348f772578567898150708920A19D', 121([_ h'', h''), 1100000000, 1000000, 5185000000, 2592000000, 1000000000, 2, 5000000, 121([_ h'', h''), 1000000, 200000000]), 122([[]]))])
| Reference Inputs:
| 121([_ 121([_ 121([_ h'8fA726c3C14916581B866fF550K31A1C4F8fDC2E9f26A9A16f4BBABE', 1]), 121([_ 121([_ 122([_ h'8fA726c3C14916581B866fF550K31A1C4F8fDC2E9f26A9A16f4BBABE', 122([[]]), (_ h'': _ h'': 4000000 ), h'8fA726c3C14916581B866fF550K31A1C4F8fDC2E9f26A9A16f4BBABE' ], 122([[]]), (_ h'': _ h'': 1000000000 ), h'873E4fE9E41E924911B8A1C53F4782EFC80F2A4f875C879F8A332' ], 121([_ 121([_ 121([_ h'01A402B8A94DC47fB302E0545958C38D00545B7C6AC22504E984922830968A', 1 ] ), 123([_ 121([_ 121([_ h'000643800136752529784C15E638DA2A2f7B1C00C9CB92277913A8C40A804', h'0000E140015f47f8f1A300f82B183E9S8E348f772578567898150708920A19D', 121([_ h'', h''), 1100000000, 1000000, 5185000000, 2592000000, 1000000000, 2, 5000000, 121([_ h'', h''), 1000000, 200000000]), 122([[]]))])
| Step 5: Execution Result
| .....
| Subscription Successfully Extended
| .....
| Test Completed
| .....
| token_name
| h'01A402B8A94DC47fB302E0545958C38D00545B7C6AC22504E984922830968A'
| token_quantity
| 1
|
| 1 tests | 1 passed | 0 failed

```

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)

```

Testing ...
payment_subscription/tests/payment_multi_validator ---
PASS [mem: 10944248, 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([_ 121([_ h'FB3D635C7B573D1B9E9BF4A6A04F25190D29B6FDBD9AC605A218' ]), 122([ ])), (_ h'': 600000000 ) ], 123([_ 121([ ])), 122([ ])))
| Penalty Output:
| 121([_ 121([_ 121([_ h'873E4FE9E41E924911BBA3C53FF4782EFC80F244F875C879F8A32' ]), 122([ ])), (_ h'': 101000000 ) ], 123([_ 121([_ 121([_ h'01A020B0A9AFC47FB302ED59459582CB0D054587C6AC22504E984922830968A' : 1 ] ), 1
23([_ 121([_ h'0000E1400120206CC8B1DF7C4138C04D3AB3336892508A22432B292A097AF6F', h'0000E140013675252978AC15638DA2A27FB1C80C9CB92277913ABDC40A86D4', 121([_ h'', h'']), 1000000)])), 122([ ])))
| Step 4: Validating Transaction
| -----
| Transaction Inputs:
| 121([_ 121([_ 121([_ h'EE155ACE9C40292074C86AFFB8C9CCD0273C81648FF1149EF368CEA6E' ]), 11), 121([_ 121([_ 121([_ h'FB3D635C7B573D1B9E9BF4A6A04F25190D29B6FDBD9AC605A218' ]), 122([ ])), (_ h'': 4000000 ) ], h'FB3D635C7B573D1B9E9BF4A6A04F25190D29B6FDBD9AC
605A218' : 1 ], h'0000E140013675252978AC15638DA2A27FB1C80C9CB92277913ABDC40A86D4' : 1 ] ), 123([_ 121([ ])), 122([ ])))], 121([_ 121([_ 121([_ h'873E4FE9E41E924911BBA3C53FF4782EFC80F244F875C879F8A32' ]), 11), 121([_ 121([_ 121([_ h'01A020B0A9AFC47FB302ED59459582CB0D054587C6AC22504E984922830968A' : 1 ] ), 1
23([_ 121([_ h'0000E1400120206CC8B1DF7C4138C04D3AB3336892508A22432B292A097AF6F', h'0000E140013675252978AC15638DA2A27FB1C80C9CB92277913ABDC40A86D4', 121([_ h'', h'']), 1000000)])), 122([ ])))
| Transaction Outputs:
| 121([_ 121([_ 121([_ h'FB3D635C7B573D1B9E9BF4A6A04F25190D29B6FDBD9AC605A218' ]), 122([ ])), (_ h'': 600000000 ) ], 123([_ 121([ ])), 122([ ])))], 121([_ 121([_ 121([_ h'873E4FE9E41E924911BBA3C53FF4782EFC80F244F875C879F8A32' ]), 11), 121([_ 121([_ 121([_ h'01A020B0A9AFC47FB302ED59459582CB0D054587C6AC22504E984922830968A' : 1 ] ), 1
23([_ 121([_ h'0000E1400120206CC8B1DF7C4138C04D3AB3336892508A22432B292A097AF6F', h'0000E140013675252978AC15638DA2A27FB1C80C9CB92277913ABDC40A86D4', 121([_ h'', h'']), 1000000)])), 122([ ])))
| Printed Tokens:
| 121([_ 121([_ 121([_ h'873E4FE9E41E924911BBA3C53FF4782EFC80F244F875C879F8A32' ]), 11), 121([_ 121([_ 121([_ h'01A020B0A9AFC47FB302ED59459582CB0D054587C6AC22504E984922830968A' : 1 ] ), 123([_ 121([_ h'0000E1400120206CC8B1DF7C4138C04D3AB3336892508A22432B292A097AF6F', h'0000E140013675252978AC15638DA2
A27FB1C80C9CB92277913ABDC40A86D4', 121([_ h'', h'']), 1000000)])), 122([ ])))
| Step 5: Execution Result
| -----
| Unsubscription Successfully Processed!
| -----
| Test Completed!
| Token name
| h'01A020B0A9AFC47FB302ED59459582CB0D054587C6AC22504E984922830968A'
| token quantity
| 1

```

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_t_withdraw)

```

Testing ...
payment_subscription/tests/payment_multi_validator
PASS [mem: 49/1365, cpu: 25589/1473] succeed_merchant_withdraw
- with traces
  test: Withdrawing Subscription Fees
  -----
  Step 1: Current Contract State
  -----
  Service Currency Symbol:
  h'873E4FE9E41E9249118BA3EC53FF4782EFC80F244F875C879F8A432'
  Payment Currency Symbol:
  h'873E4FE9E41E9249118BA3EC53FF4782EFC80F244F875C879F8A432'
  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([ 121([ h'873E4FE9E41E9249118BA3EC53FF4782EFC80F244F875C879F8A432' ], 122([ ])), [ _ h'': 200000000 ], h'873E4FE9E41E9249118BA3EC53FF4782EFC80F244F875C879F8A432': [ _ h'000DE140013675252978AC15E638DA2A27F81C00C9C8B92277913ABDC40A86D4': 1 ]
), 122([ 121([ ])), 122([ ]))
  Remaining Payment Output:
  121([ 121([ 121([ h'873E4FE9E41E9249118BA3EC53FF4782EFC80F244F875C879F8A432' ], 122([ ])), [ _ h'': 900000000 ], h'873E4FE9E41E9249118BA3EC53FF4782EFC80F244F875C879F8A432': [ _ h'01A8028049AFDC47FB302E59459582CB600545B7C6AC22504E984922830968A': 1 ]
), 122([ 121([ h'000DE140013675252978AC15E638DA2A27F81C00C9C8B92277913ABDC40A86D4', h'000DE14001547F8F1A300F82B1B3E9858E3AB7772578567898150708920A19D', 121([ _ h'', h'']), 800000000, 1000000, 2593000000, 2593000000, 1000000000, 1, 5185000000, 121([ _ h'', h'']), 1000
000, 200000000 ])), 122([ ]))
  Step 4: Updating Payment Datum
  -----
  Original Last Claimed:
  1000000
  Updated Last Claimed:
  5185000000
  Step 5: Execution Result
  -----
  Withdrawal Successfully Processed!
  -----
  Test Completed!
  -----
tests | 1 passed | 0 failed
Summary 1 check, 0 errors, 0 warnings

```

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

subscriber_withdraw)[illegible]

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.