# Subscription Payments Smart Contract Documentation

## 1. Overview

The Subscription Payments Smart Contract provides a decentralized solution for recurring payments between subscribers and service providers. It enables users to lock funds for a subscription period, ensures providers can claim payments only at scheduled intervals, and allows subscribers to cancel subscriptions at any time with appropriate refunds.  
  
This contract is useful for SaaS models, newsletters, online content platforms, or any recurring payment service.

## 2. Actors (Roles)

1. Subscriber  
- Pays for a service by locking funds into the contract.  
- Can cancel the subscription before all intervals are completed.  
- Eligible to receive refunds for unused time.  
  
2. Provider  
- Offers services in exchange for recurring payments.  
- Can claim subscription payments only after the interval has passed.  
- Can withdraw accumulated earnings from the contract.  
  
3. (Optional) Admin  
- Not required for a minimal design.  
- Could be included if multiple providers are managed under one system.

## 3. Core Features

### 3.1 Subscription Creation

A subscriber creates a subscription by depositing funds for one or more intervals. Each subscription records:  
- Subscriber address  
- Provider address  
- Payment amount per interval  
- Number of intervals or total balance  
- Interval duration (e.g., 30 days)  
- Next payment due timestamp  
- Active/inactive status

### 3.2 Payment Processing

- At each interval, the provider can request payment.  
- The contract checks if the current time is past the `nextPaymentTime`.  
- If yes, the payment is marked as due, and the provider can later withdraw it.  
- The `nextPaymentTime` is updated for the next cycle.

### 3.3 Withdrawal by Provider

- Providers can withdraw only what has been accrued (payments that have passed their due interval).  
- Prevents withdrawing future payments in advance.  
- Must handle cases where multiple intervals have passed without withdrawal.

### 3.4 Cancellation of Subscription

- Subscriber can cancel anytime.  
- The remaining unused balance is refunded.  
- Provider only keeps the amount for completed intervals.  
- Contract marks the subscription as inactive.

### 3.5 Refund Handling

- If a subscription is canceled midway, refund logic ensures:  
- Provider keeps funds for elapsed intervals.  
- Subscriber receives unused funds.

### 3.6 Events

The contract must emit events for transparency and easy off-chain tracking:  
- SubscriptionCreated(subscriber, provider, amount, interval, totalDeposit, startTime)  
- PaymentProcessed(subId, provider, amount, timestamp)  
- SubscriptionCancelled(subId, subscriber, refundAmount, timestamp)  
- FundsWithdrawn(provider, amount, timestamp)

## 4. Workflow Example

1. Alice (Subscriber) creates a subscription to Bob (Provider) for:  
- $10 per 30 days  
- 3 months prepaid → $30 deposited  
- Next payment due = Today + 30 days  
  
2. After 30 days → Bob calls processPayment.  
- $10 is unlocked for Bob.  
- Next payment due = Today + 30 more days  
  
3. After 60 days → Bob forgets to claim.  
- At day 61, he calls again.  
- Contract unlocks $20 (for 2 months).  
  
4. At day 70 → Alice cancels subscription.  
- Bob has already earned $20.  
- Remaining $10 is refunded to Alice.  
- Subscription is set inactive.

## 5. Edge Cases to Handle

- Missed withdrawals: If provider doesn’t withdraw on time, the system should accumulate past due payments.  
- Insufficient funds: If subscriber deposits less than expected, subscription ends early.  
- Double withdrawal attempts: Prevent provider from withdrawing the same interval twice.  
- Time manipulation: Always use block.timestamp for time checks.  
- Refund race conditions: Ensure cancellation refunds are calculated before provider withdrawals.

## 6. Security Considerations

- Reentrancy protection on withdrawals and refunds.  
- Access control to ensure only subscriber can cancel, only provider can withdraw.  
- Balance safety to prevent over-withdrawal.  
- Event logging for external tracking and transparency.

## 7. Possible Extensions

- Auto-renewal → subscriber recharges balance before expiry.  
- Discounts for longer commitments (e.g., 6 months upfront = 10% off).  
- Penalty for early cancellation (optional feature).  
- Multi-subscriber pooling → one provider with many subscribers managed in one contract.  
- NFT-based subscription → access rights tied to NFTs instead of addresses.