BLS Staking Smart Contract Documentation

Introduction

The BLSStaking smart contract is designed to facilitate the staking and unstaking of BLS tokens in exchange for staked BLS tokens (stBLS). This contract allows users to stake their BLS tokens and receive stBLS tokens, representing their stake, which can be used for various decentralized finance (DeFi) applications. Users can also unstake their BLS tokens by burning their stBLS tokens.

Contract Overview

Inheritance

The BLSStaking contract inherits from the Ownable contract, which provides ownership management functionalities.

State Variables

- **blsToken**: An instance of the IERC20 interface representing the BLS token.
- **stBlsToken**: An instance of the IERC20 interface representing the staked BLS token.
- **stakedAmountOf**: A mapping that tracks the amount of BLS tokens staked by each address.

Constructor

```solidity
constructor(address initialOwner) Ownable(initialOwner) {}
```

The constructor initializes the contract and sets the initial owner.

Functions

setAddresses

```solidity
function setAddresses(address _blsToken, address _stBlsToken) external
```

This function sets the addresses of the BLS token and the staked BLS token contracts.
- **_blsToken**: The address of the BLS token contract.
- **_stBlsToken**: The address of the staked BLS token contract.

stake
```solidity
function stake(uint256 amount) external
```

This function allows users to stake a specified amount of BLS tokens.
- **amount**: The amount of BLS tokens to stake.

The function performs the following actions:
1. Checks that the staking amount is greater than 0.
2. Transfers BLS tokens from the sender to the contract.
3. Mints an equivalent amount of stBLS tokens to the sender.
4. Updates the `stakedAmountOf` mapping to reflect the new staked amount for the sender.

unstake
```solidity
function unstake(uint256 amount) external
```

This function allows users to unstake a specified amount of BLS tokens.
- **amount**: The amount of BLS tokens to unstake.

The function performs the following actions:
1. Checks that the unstaking amount is greater than 0.
2. Ensures that the sender has enough staked BLS tokens to unstake the specified amount.
3. Burns the specified amount of stBLS tokens from the sender.
4. Transfers an equivalent amount of BLS tokens from the contract to the sender.
5. Updates the `stakedAmountOf` mapping to reflect the new staked amount for the sender.

Example Usage

Staking BLS Tokens
```solidity
BLSStaking stakingContract = new BLSStaking(ownerAddress);
stakingContract.setAddresses(blsTokenAddress, stBlsTokenAddress);
stakingContract.stake(100);
```

In this example, the user stakes 100 BLS tokens.

Unstaking BLS Tokens
```solidity
stakingContract.unstake(50);
```

In this example, the user unstakes 50 BLS tokens.

Conclusion

The BLSStaking smart contract provides a straightforward mechanism for staking and unstaking BLS tokens, allowing users to earn stBLS tokens while participating in the staking process. This contract is essential for DeFi applications that require token staking functionalities.