# BSToken Contract Documentation

The **BSToken** contract is the foundational ERC-20 token contract for the **Bullish Social ecosystem**. It serves as the native token for governance, rewards, and other ecosystem functionalities. The contract is built on the OpenZeppelin library and includes features such as **token minting**, **burning**, **vesting**, and **trusted address management**.

The **BSToken** contract is an ERC-20 token with a fixed maximum supply of **256,000,000 tokens**. It implements the following key features:

1. **Initial Token Distribution:** Tokens are distributed according to predefined percentages for rewards, liquidity, team, marketing, treasury, partners, and affiliates.
2. **Vesting Mechanism:** Tokens allocated to the team, marketing, and partners are locked in vesting wallets and released over time.
3. **Trusted Address Management:** Certain addresses (e.g., governance and reward contracts) are marked as trusted to enable secure interactions.
4. **Burnable Tokens:** Users can burn their tokens, reducing the total supply.
5. **Reentrancy Protection:** The contract is secured against reentrancy attacks using OpenZeppelin's `ReentrancyGuard`.

## Key Features

### 1. Tokenomics

The tokenomics of **BSToken** are designed to support the ecosystem's growth and sustainability. The initial token distribution is as follows:

- **Reward & Incentives:** 50% (128,000,000 tokens)
- **Liquidity & Fair Launch:** 15% (38,400,000 tokens)
- **Team Members:** 15% (38,400,000 tokens)
- **Marketing:** 10% (25,600,000 tokens)
- **Treasury:** 5% (12,800,000 tokens)
- **Partners:** 3% (7,680,000 tokens)
- **Affiliate:** 2% (5,120,000 tokens)

### 2. Initialization

The contract must be initialized by the **signer** (deployer) to distribute tokens to predefined addresses. During initialization:

- Tokens are minted and allocated to the **rewarder**, **liquidity provider**, **governor**, and **vesting wallets**.
- The **governor** and **rewarder** addresses are set as trusted.

### 3. Vesting Mechanism

Tokens allocated to the **team**, **marketing**, and **partners** are locked in vesting wallets. The vesting periods are as follows:

- **Team:** 545 days, starting after a 180-day cliff.
- **Marketing:** 365 days, starting immediately.
- **Partners:** 545 days, starting immediately.

### 4. Trusted Address Management

The contract maintains a mapping of trusted addresses. Only trusted addresses can interact with certain functionalities in the ecosystem (e.g., governance and reward contracts). The **governor** can update the list of trusted addresses.

### 5. Burnable Tokens

Users can burn their tokens, reducing the total supply. The total amount of burned tokens is tracked in the `totalBurned` variable.

## Key Functions

1. **`initialize`**
    a. Initializes the contract by distributing tokens to specified addresses.
    b. Can only be called once by the **signer**.
    c. Parameters:
        - `governorAddress`: Address of the governance contract.
        - `rewarderAddress`: Address of the reward distribution contract.
        - `teamMembersAddress`: Address for team member vesting.
        - `marketingAddress`: Address for marketing vesting.
        - `partnersAddress`: Address for partners vesting.
2. **`setTrustedAddress`**
    a. Sets or unsets a trusted address.
    b. Can only be called by the **governor**.
    c. Parameters:
        - `addr`: The address to set as trusted.
        - `isTrusted`: Whether the address should be trusted or not.
3. **`rewardSupply`**
    a. Returns the current balance of the rewarder address.
4. **`treasuryBalance`**
    a. Returns the current balance of the governor address (treasury).
5. **`isTrustedAddress`**
    a. Checks if an address is trusted.
    b. Parameters:
        - `addr`: The address to check.
    c. Returns: `true` if the address is trusted, otherwise `false`.

### Private Functions

1. **`mint`**
    a. Mints tokens to a specified address.
    b. Can only be called during initialization.
    c. Parameters:
        - `to`: The address to mint tokens to.
        - `amount`: The amount of tokens to mint.
2. **`_setTrustedAddress`**
    a. Internal function to set or unset a trusted address.
    b. Emits the `TrustedAddressSet` event.
    c. Parameters:
        - `addr`: The address to set as trusted.
        - `isTrusted`: Whether the address should be trusted or not.
3. **`_setVesting`**
    a. Creates a vesting wallet and mints tokens to it.
    b. Emits the `VestingWalletSet` event.
    c. Parameters:
        - `addr`: The beneficiary address of the vesting wallet.
        - `amount`: The amount of tokens to vest.
        - `startTimestamp`: The start timestamp of the vesting period.
        - `durationSeconds`: The duration of the vesting period in seconds.
4. **`_calculateSupply`**
    a. Calculates the token amount based on a percentage of the maximum supply.
    b. Parameters:
        - `percent`: The percentage of the maximum supply to calculate.
    c. Returns: The calculated token amount.

## Usage Scenarios

1. **Initialization:**
    a. The **signer** deploys the contract and calls `initialize` to distribute tokens to the **rewarder**, **governor**, and vesting wallets.
2. **Vesting:**
    a. Tokens allocated to the **team**, **marketing**, and **partners** are released over time according to the vesting schedule.
3. **Governance:**
    a. The **governor** contract uses trusted addresses to manage critical functions and proposals.
4. **Reward Distribution:**
    a. The **rewarder** contract weekly distributes tokens to users based on their performance.
5. **Token Burning:**
    a. In the future there will be contracts in the ecosystem that use this function for deflation.