SSN TOKEN TECHNICAL DOCUMENTATION

Version: 1.0.0
Last Updated: February 25, 2025
Smart Contract Address: 0x615FB7Debb3d7b229D884e771a9fe5e3AcCe1D1A

TABLE OF CONTENTS

1. INTRODUCTION

SSN Token is an ERC-20 compliant token on the Ethereum blockchain designed to honor Dr. Sam Shafiishuna Nujoma's legacy through blockchain innovation. The token features a fixed supply of 100,000 tokens, with each token backed by approximately 0.4 carats of certified Namibian diamonds.

This technical documentation provides developers, exchanges, and technical users with comprehensive information about SSN Token's implementation, functionality, and integration methods.

2. CONTRACT SPECIFICATIONS

2.1 Basic Information

Parameter	Value
Token Name	SSN Token
Token Symbol	SSN

Parameter	Value
Decimals	6
Total Supply	100,000
Contract Address	0x615FB7Debb3d7b229D884e771a9fe5e3AcCe1D1A
Blockchain	Ethereum
Standard	ERC-20
Compiler Version	Solidity 0.8.17
Optimization	Enabled (200 runs)

2.2 Contract Architecture

SSN Token implements the standard ERC-20 interface with additional security features:

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.17;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/security/ReentrancyGuard.sol";
import "@openzeppelin/contracts/access/Ownable.sol";

contract SSNToken is ERC20, ReentrancyGuard, Ownable {
    // Contract implementation
    // See section 3 for detailed function descriptions
}
```

2.3 Code Verification

The contract source code has been verified on Etherscan and can be accessed at: https://etherscan.io/address/0x615FB7Debb3d7b229D884e771a9fe5e3AcCe1D1A#code

3. TOKEN FUNCTIONS

3.1 Standard ERC-20 Functions

SSN Token implements all standard ERC-20 functions:

Function	Description
totalSupply()	Returns the total token supply (100,000)
balanceOf(address account)	Returns the token balance of the specified address
transfer(address to, uint256 amount)	Transfers tokens to the specified address
allowance(address owner, address spender)	Returns the remaining allowance for a spender
approve(address spender, uint256 amount)	Approves a spender to withdraw from your account
transferFrom(address from, address to, uint256 amount)	Transfers tokens from one address to another

3.2 Additional Functions

Function	Description
diamondBackingDetails()	Returns information about the diamond backing
getTokenomicsData()	Returns detailed tokenomics information

3.3 Events

Event	Description
Transfer(address indexed from, address indexed to, uint256 value)	Emitted when tokens are transferred

Event	Description
Approval(address indexed owner, address indexed spender, uint256 value)	Emitted when approval is set

4. SECURITY FEATURES

4.1 Contract Security

SSN Token implements multiple security features to ensure the integrity and safety of the token:

- **Reentrancy Protection**: Utilizes OpenZeppelin's ReentrancyGuard to prevent reentrancy attacks
- **Integer Overflow Protection**: Solidity 0.8.x's built-in overflow checking
- **Access Control**: Carefully implemented permission controls using OpenZeppelin's Ownable
- **Fixed Supply**: No mint function, ensuring the total supply can never be increased
- **Independent Audit**: Contract audited by blockchain security specialists

4.2 Security Audit Results

The smart contract has undergone a comprehensive security audit with the following results:

- **Critical Vulnerabilities**: None found
- **Major Vulnerabilities**: None found
- **Minor Findings**: 2 (resolved prior to deployment)
- **Informational Findings**: 3 (related to gas optimization)

Full audit report available upon request for exchanges and partners.

5. INTEGRATION GUIDE

5.1 Exchange Integration

For exchanges looking to list SSN Token, follow these steps:

1. **Contract Verification**: Verify the contract at address 0x615FB7Debb3d7b229D884e771a9fe5e3AcCe1D1A
2. **Token Configuration**:

- Name: SSN Token
- Symbol: SSN
- Decimals: 6
- Standard: ERC-20

3. **RPC Configuration**: Use standard Ethereum RPC endpoints
4. **Test Transactions**: Perform test transfers with small amounts
5. **Implement Standard Endpoints**: Use standard ERC-20 endpoints for balance checking and transfers

5.2 Wallet Integration

SSN Token is compatible with all Ethereum wallets that support the ERC-20 standard. No special configuration is required.

5.3 API Interaction Examples

Web3.js Example

javascript

Copy

```
const Web3 = require('web3');

const web3 = new Web3('https://mainnet.infura.io/v3/YOUR_INFURA_KEY');


const ssnTokenABI = [...]; // Full ABI available on Etherscan

const ssnTokenAddress = '0x615FB7Debb3d7b229D884e771a9fe5e3AcCe1D1A';

const ssnToken = new web3.eth.Contract(ssnTokenABI, ssnTokenAddress);


// Get token balance

async function getBalance(address) {
```

```
    const balance = await ssnToken.methods.balanceOf(address).call();
    return web3.utils.fromWei(balance, 'ether');
}

// Transfer tokens
async function transferTokens(fromAddress, toAddress, amount, privateKey) {
    const nonce = await web3.eth.getTransactionCount(fromAddress);
    const data = ssnToken.methods.transfer(toAddress, web3.utils.toWei(amount, 'ether')).encodeABI();

    const tx = {
        from: fromAddress,
        to: ssnTokenAddress,
        nonce: nonce,
        gas: 200000,
        data: data
    };

    const signedTx = await web3.eth.accounts.signTransaction(tx, privateKey);
    return web3.eth.sendSignedTransaction(signedTx.rawTransaction);
}
```

Ethers.js Example

javascript

Copy

```
const { ethers } = require('ethers');

const provider = new
ethers.providers.JsonRpcProvider('https://mainnet.infura.io/v3/YOUR_INFURA_KEY');


const ssnTokenABI = [...]; // Full ABI available on Etherscan

const ssnTokenAddress = '0x615FB7Debb3d7b229D884e771a9fe5e3AcCe1D1A';

const ssnToken = new ethers.Contract(ssnTokenAddress, ssnTokenABI, provider);


// Get token balance

async function getBalance(address) {

   const balance = await ssnToken.balanceOf(address);

   return ethers.utils.formatUnits(balance, 6);

}


// Transfer tokens

async function transferTokens(toAddress, amount, signer) {

   const ssnTokenWithSigner = ssnToken.connect(signer);

   const tx = await ssnTokenWithSigner.transfer(toAddress, ethers.utils.parseUnits(amount, 6));

   return tx.wait();

}
```

6. DIAMOND BACKING IMPLEMENTATION

6.1 Technical Implementation

The diamond backing for SSN Token operates through a dual-layer implementation:

1. **Blockchain Layer**: Immutable smart contract with fixed supply

2. **Physical Layer**: 40,000 carats of certified Namibian diamonds stored in secure facilities

The two layers are connected through:

- Regular audit processes that verify the diamond backing
- Published verification reports that confirm the existence and security of the diamonds
- Digital certification using cryptographic signatures of audit reports

6.2 Verification Methodology

The verification process incorporates several technical components:

- Digital fingerprinting of diamond certification documents
- Cryptographic signing of verification reports by authorized auditors
- Secure, timestamped publication of verification results
- Hash verification of audit documentation

6.3 Technical Specifications of Diamond Reserve

Parameter	**Value**
Total Carats	40,000
Diamonds Per Token	~0.4 carats
Quality Assessment	VS1-VS2 clarity, D-F color
Verification Frequency	Quarterly
Verification Methodology	Physical inspection + documentation review
Security Protocol	Multi-layer physical and digital security

7. TECHNICAL FAQS

7.1 How does SSN Token handle gas fees?

SSN Token operates as a standard ERC-20 token on the Ethereum network. Users must have ETH in their wallet to cover gas fees for transactions.

7.2 Is SSN Token compatible with hardware wallets?

Yes, SSN Token can be stored on any hardware wallet that supports Ethereum ERC-20 tokens, including Ledger, Trezor, and others.

7.3 Can the total supply be increased?

No. The smart contract does not include a mint function. The total supply is fixed at 100,000 tokens and cannot be increased.

7.4 How can I verify the token's diamond backing?

The diamond backing is verified through quarterly audit reports conducted by independent gemological experts. These reports are published on the SSN Token website.

7.5 Is the contract upgradeable?

No, the SSN Token contract is not upgradeable. This design choice ensures that the core functionality, including the fixed supply, cannot be altered.

7.6 What is the SSN Token's transaction speed?

As an ERC-20 token on Ethereum, SSN Token transactions are processed at the same speed as the Ethereum network, typically within 15 seconds to 5 minutes depending on network congestion and gas price.

8. SUPPORT & CONTACT

8.1 Technical Support

For technical inquiries, integration assistance, or smart contract questions:

- Email: technical@nujoma.biz
- Telegram: https://t.me/+Hy1ILj-DMmE5ZmI8

8.2 Documentation Updates

This technical documentation is versioned and will be updated as needed. Check the website for the latest version.

Disclaimer: This technical documentation is intended for informational purposes only. Developers, exchanges, and users should perform their own due diligence before integrating or interacting with the SSN Token smart contract.

Document Version: 1.0.0 (February 25, 2025