# CFG NINJA AUDITS

## Security Assessment

# Tweety Inu Token

April 19, 2023

Audit Status: Pass

Audit Edition: Advance


POWERED BY BLADE POOL

# Table of Contents





# Assessment Summary

This report has been prepared for Tweety Inu Token on the Binance Smart Chain network. CFGNINJA provides both client-centered and user-centered examination of the smart contracts and their current status when applicable. This report represents the security assessment made to find issues and vulnerabilities on the source code along with the current liquidity and token holder statistics of the protocol.

A comprehensive examination has been performed, utilizing Cross Referencing, Static Analysis, In-House Security Tools, and line-by-line Manual Review.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Inspecting liquidity and holders statistics to inform the current status to both users and client when applicable.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Verifying contract functions that allow trusted and/or untrusted actors to mint, lock, pause, and transfer assets.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders
- Thorough line-by-line manual review of the entire codebase by industry experts.

# Project Overview

## Token Summary

| Parameter | Result |
|---|---|
| Address | 0xEc1869C673E85698DA3Ef3B48B791Bfa8c697d51 |
| Name | Tweety Inu |
| Token Tracker | Tweety Inu (TWEETY) |
| Decimals | 9 |
| Supply | 420,000,000,000,000,000 |
| Platform | Binance Smart Chain |
| compiler | v0.8.18+commit.87f61d96 |
| Contract Name | TweetyInu |
| Optimization | No |
| LicenseType | None |
| Language | Solidity |
| Codebase | https://bscscan.com/address/0xEc1869C673E85698DA3Ef3B48B791Bfa8c697d51#code |
| Payment Tx | 0x7c01448176f2e9c940183803a4b648038cc99df24f0b796575b3ddf549ebe89f |

# Project Overview

## Risk Analysis Summary

| Parameter | Result |
|---|---|
| Buy Tax | 9% |
| Sale Tax | 9% |
| Is honeypot? | Clean |
| Is CoolDown? | undefined |
| Can edit tax? | Yes |
| Is anti whale? | Yes |
| Is blacklisted? | No |
| Is whitelisted? | No |
| Holders | 1 |
| Confidence Level | Low |

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.

# Project Overview

## Simulation Summary

| Parameter | Result |
|---|---|
| Transfer From Owner | Pass |
| Transfer From Holder | Pass |
| Add Liquidity | Pass |
| Buy from Owner | Pass |
| Buy from Holder | Pass |
| Remove Liquidity | Pass |
| SwapAndLiquify | Pass |
| RemoveLiquidity | Pass |
| LaunchPad | PinkSale |

The following quick summary it's added to the project overview; however, there are more details about the audit and its results. Please read every detail.

## Main Contract Assessed Contract Name

| Name | Contract | Live |
|---|---|---|
| Tweety Inu | 0xEc1869C673E85698DA3Ef3B48B791Bfa8c697d51 | No |

## TestNet Contract Assessed Contract Name

| Name | Contract | Live |
|---|---|---|
| Tweety Inu | 0x76Eb6a3c0Ab3479571554616eAa3c2190f1E77E5 | No |

## Solidity Code Provided

| SolID | File Sha-1 | FileName |
|---|---|---|
| TweetyInu | 2fdd158f34d42ac0e7e945e189aa1f071ded05f0 | TweetyInu.sol |
| TweetyInu | | |
| TweetyInu | | |
| TweetyInu | | |
| TweetyInu | | |
| TweetyInu | | |

# Mint Check

**The project owners of Tweety Inu do not have a mint function in the contract, owner cannot mint tokens after initial deploy.**

**The Project has a Total Supply of 420,000,000,000,000,000 and cannot mint any more than the Max Supply.**

**Mint Notes:**

**Auditor Notes:**

**Project Owner Notes:**

# Fees Check

## The project owners of Tweety Inu do not have the ability to set fees higher than 25% .

## The team May have fees defined; however, they can't set those fees higher than 25% or may not be able to configure the same.

**Tax Fee Notes:**

**Auditor Notes: The contract currently has 9% buy and 9% sale taxes, and cannot be set higher than 20%.**

**Project Owner Notes:**

**Fees Can Be Changed up to a maximum of 25%**

# Blacklist Check

**The project owners of Tweety Inu do not have a blacklist function their contract.**

**The Project allow owners to transfer their tokens without any restrictions.**

**Token owner cannot blacklist the contract: Malicious or compromised owners can trap contracts relying on tokens with a blacklist.**

**Blacklist Notes:**

**Auditor Notes:**

**Project Owner Notes:**

# MaxTx Check

## The Project Owners of Tweety Inu can set max tx amount.

## The ability to set MaxTx can be used as bad actor, this can limit the ability of investors to sale their tokens at any given time if is set too low..

## We recommend the project to set MaxTx to Total Supply or simiar to avoid swap or transfer from failures

**MaxTX Notes:**

**Auditor Notes:** uint256 public _maxWalletToken = _totalSupply.mul(1).div(100); uint256 public _maxTxAmount = _totalSupply.mul(100).div(100);

**Project Owner Notes:**

# Pause Trade Check

## The Project Owners of Tweety Inu don't have the ability to stop or pause trading.

## The Team has done a great job to avoid stop trading, and investors has the ability to trade at any given time without any problems

**Pause Trade Notes:**

**Auditor Notes: no Pause, however there is a start trade.**

**Project Owner Notes: .**

**Owner can’t pause trading**

# Contract Ownership

**The contract ownership of Tweety Inu is not currently renounced. The ownership of the contract grants special powers to the protocol creators, making them the sole addresses that can call sensible ownable functions that may alter the state of the protocol.**

**The current owner is the address 0x3319ebfaacf12490edad5e8aa0370dc11a827f24 which can be viewed:**
**HERE**

**The owner wallet has the power to call the functions displayed on the privileged functions chart below, if the owner's wallet is compromised, they could exploit these privileges.**

**We recommend the team renounce ownership at the right time, if possible, or gradually migrate to a timelock with governing functionalities regarding transparency and safety considerations.**

**We recommend the team use a Multisignature Wallet if the contract is not going to be renounced; this will give the team more control over the contract.**

# Liquidity Ownership

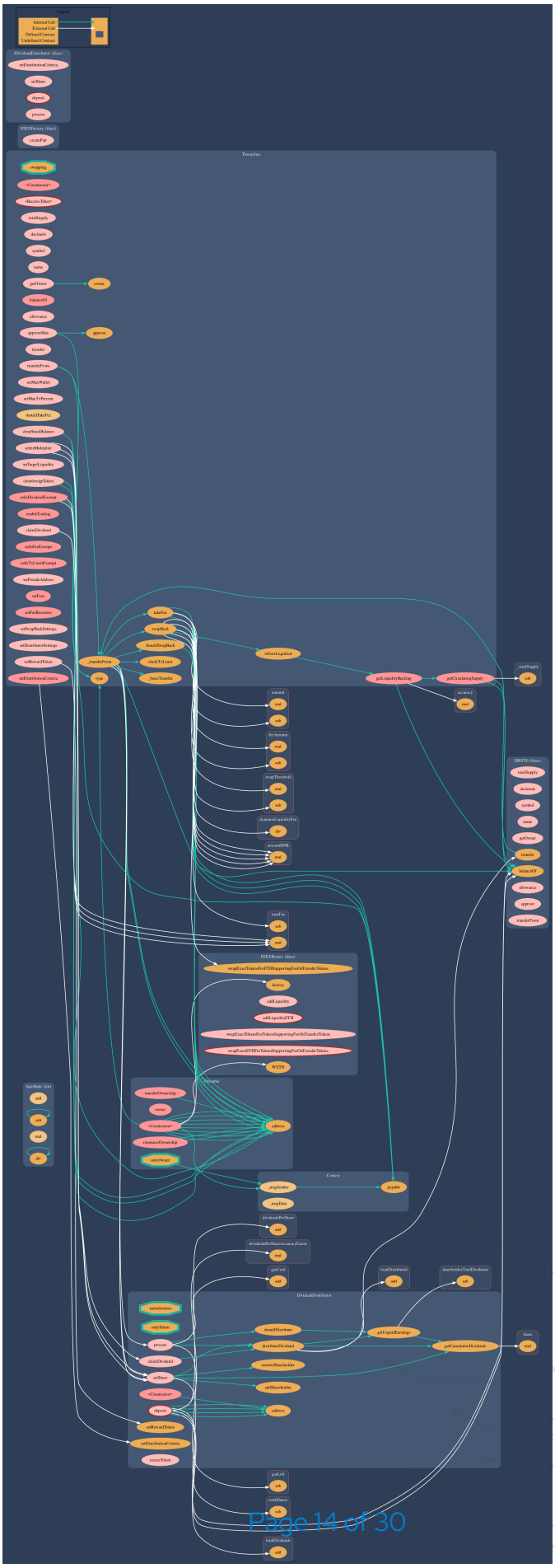
The token does not have liquidity at the moment of the audit, block  27495119

If liquidity is unlocked, then the token developers can do what is infamously known as 'rugpull'. Once investors start buying token from the exchange, the liquidity pool will accumulate more and more coins of established value (e.g., ETH or BNB or Tether). This is because investors are basically sending these tokens of value to the exchange, to get the new token. Developers can withdraw this liquidity from the exchange, cash in all the value and run off with it. Liquidity is locked by renouncing the ownership of liquidity pool (LP) tokens for a fixed time period, by sending them to a time-lock smart contract. Without ownership of LP tokens, developers cannot get liquidity pool funds back. This provides confidence to the investors that the token developers will not run away with the liquidity money. It is now a standard practice that all token developers follow, and this is what really differentiates a scam coin from a real one.

Read More

# Call Graph

The contract for Tweety Inu has the following call graph structure.



# KYC Information

## The Project Owners of Tweety Inu is not KYC.

**KYC Information Notes:**

**Auditor Notes: KYC to be completed by PinkSale, project will be a SAFU Project.**

**Project Owner Notes:**

# Smart Contract Vulnerability Checks

**The Smart Contract Weakness Classification Registry (SWC Registry) is an implementation of the weakness classification scheme proposed in EIP-1470. It is loosely aligned to the terminologies and structure used in the Common Weakness Enumeration (CWE) while overlaying a wide range of weakness variants that are specific to smart contracts.**

| ID | Severity | Name | File | location |
|---|---|---|---|---|
| SWC-100 | Pass | Function Default Visibility | Tweetylnu.sol | L: 0 C: 0 |
| SWC-101 | Pass | Integer Overflow and Underflow. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-102 | Pass | Outdated Compiler Version file. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-103 | Pass | A floating pragma is set. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-104 | Pass | Unchecked Call Return Value. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-105 | Pass | Unprotected Ether Withdrawal. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-106 | Pass | Unprotected SELFDESTRUCT Instruction | Tweetylnu.sol | L: 0 C: 0 |
| SWC-107 | Pass | Read of persistent state following external call. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-108 | Pass | State variable visibility is not set.. | Tweetylnu.sol | L: 175 C: 12 |
| SWC-109 | Pass | Uninitialized Storage Pointer. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-110 | Pass | Assert Violation. | Tweetylnu.sol | L: 0 C: 0 |



| ID | Severity | Name | File | location |
|---|---|---|---|---|
| SWC-111 | Pass | Use of Deprecated Solidity Functions. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-112 | Pass | Delegate Call to Untrusted Callee. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-113 | Pass | Multiple calls are executed in the same transaction. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-114 | Pass | Transaction Order Dependence. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-115 | Pass | Authorization through tx.origin. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-116 | Pass | A control flow decision is made based on The block.timestamp environment variable. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-117 | Pass | Signature Malleability. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-118 | Pass | Incorrect Constructor Name. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-119 | Pass | Shadowing State Variables. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-120 | Pass | Potential use of block.number as source of randonmness. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-121 | Pass | Missing Protection against Signature Replay Attacks. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-122 | Pass | Lack of Proper Signature Verification. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-123 | Pass | Requirement Violation. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-124 | Pass | Write to Arbitrary Storage Location. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-125 | Pass | Incorrect Inheritance Order. | Tweetylnu.sol | L: 0 C: 0 |

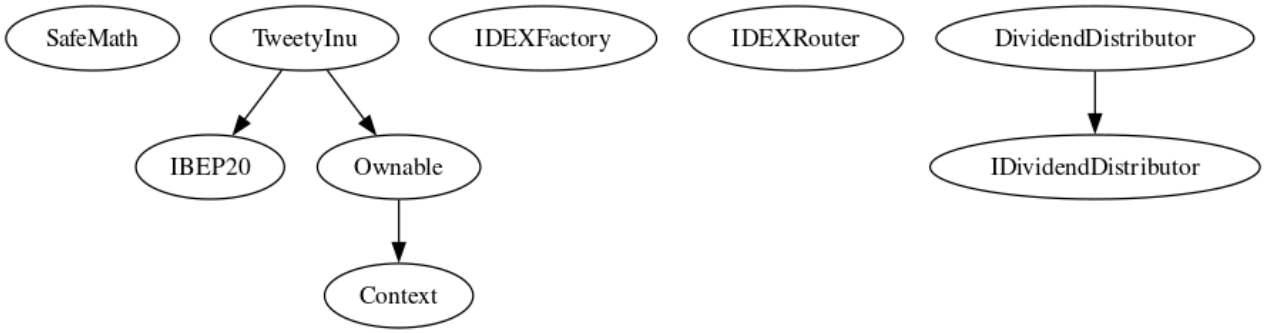


| ID | Severity | Name | File | location |
|---|---|---|---|---|
| SWC-126 | Pass | Insufficient Gas Griefing. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-127 | Pass | Arbitrary Jump with Function Type Variable. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-128 | Pass | DoS With Block Gas Limit. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-129 | Pass | Typographical Error. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-130 | Pass | Right-To-Left-Override control character (U +202E). | Tweetylnu.sol | L: 0 C: 0 |
| SWC-131 | Pass | Presence of unused variables. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-132 | Pass | Unexpected Ether balance. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-133 | Pass | Hash Collisions with Multiple Variable Length Arguments. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-134 | Pass | Message call with hardcoded gas amount. | Tweetylnu.sol | L: 0 C: 0 |
| SWC-135 | Pass | Code With No Effects (Irrelevant/Dead Code). | Tweetylnu.sol | L: 0 C: 0 |
| SWC-136 | Pass | Unencrypted Private Data On-Chain. | Tweetylnu.sol | L: 0 C: 0 |

We scan the contract for additional security issues using MYTHX and industry-standard security scanning tools.

# Inheritance

## The contract for Tweety Inu has the following inheritance structure.



# Privileged Functions (onlyOwner)

Please Note if the contract is Renounced none of this functions can be executed.

| Function Name | Parameters | Visibility |
|---|---|---|
| setMaxWallet | | External |
| setMaxTxPercent | | External |
| clearforeignToken | | External |
| setRewardToken | | External |
| setTargetLiquidity | | External |
| setfeeMultiplier | | External |
| enableTrading | | External |
| setIsDividendExempt | | External |
| setIsFeeExempt | | External |
| setIsTxLimitExempt | | External |
| setPresaleAddress | | External |
| setFees | | External |
| setFeeReceivers | | External |

| Function Name | Parameters | Visibility |
| --- | --- | --- |
| setSwapBackSettings | | External |
| setDistributorSettings | | External |
| setDistributionCriteria | | External |
| setDistributionCriteria | | External |

# Smart Contract Advance Checks

| ID | Severity | Name | Result | Status |
|---|---|---|---|---|
| TWEETY-01 | Minor | Potential Sandwich Attacks. | Pass | Remediated |
| TWEETY-02 | Minor | Function Visibility Optimization | Pass | Remediated |
| TWEETY-03 | Minor | Lack of Input Validation. | Pass | Remediated |
| TWEETY-04 | Major | Centralized Risk In addLiquidity. | Pass | Remediated |
| TWEETY-05 | Major | Missing Event Emission. | Pass | Renediated |
| TWEETY-06 | Minor | Conformance with Solidity Naming Conventions. | Pass | Remediated |
| TWEETY-07 | Minor | State Variables could be Declared Constant. | Pass | Remediated |
| TWEETY-08 | Major | Dead Code Elimination. | Pass | Not-Found |
| TWEETY-09 | Major | Third Party Dependencies. | Pass | Not-Found |
| TWEETY-10 | Major | Initial Token Distribution. | Pass | Not-Found |
| TWEETY-11 | Major | dAPP Approval is set to all NFTs on the wallet an not limited to an specific contract. | Pass | Not-Found |
| TWEETY-12 | Major | Centralization Risks In The X Role | Pass | Not-Found |
| TWEETY-13 | Informational | Extra Gas Cost For User.. | Pass | Remediated |
| TWEETY-14 | Medium | Unnecessary Use Of SafeMath | Fail | Pending |



| ID | Severity | Name | Result | Status |
|---|---|---|---|---|
| TWEETY-15 | Medium | Symbol Length Limitation due to Solidity Naming Standards. | Pass | Not-Found |
| TWEETY-16 | Medium | Invalid collection of Taxes during Transfer. | Pass | Not-Found |

# TWEETY-14 | Unnecessary Use Of SafeMath

| Category | Severity | Location | Status |
| --- | --- | --- | --- |
| Logical Issue | Medium | TweetyInu.sol: 13,9 | Pending |

## Description

The SafeMath library is used unnecessarily. With Solidity compiler versions 0.8.0 or newer, arithmetic operations
will automatically revert in case of integer overflow or underflow.
library SafeMath {
An implementation of SafeMath library is found.
using SafeMath for uint256;
SafeMath library is used for uint256 type in  contract.
_balances[recipient] = _balances[recipient].add(amount);
magnifiedDividendPerShare = magnifiedDividendPerShare.add(
(amount).mul(magnitude) / totalSupply()
);
Note: Only a sample of 2 SafeMath library usage in this contract (out of 14) are shown above.

## Remediation

We advise removing the usage of SafeMath library and using the built-in arithmetic operations provided by the
Solidity programming language

## Project Action

# Technical Findings Summary

## Classification of Risk

| Severity | Description |
|---|---|
| Critical | Risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks. |
| Major | Risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project. |
| Medium | Risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform |
| Minor | Risks can be any of the above but on a smaller scale. They generally do not compromise the overall integrity of the Project, but they may be less efficient than other solutions. |
| Informational | Errors are often recommended to improve the code's style or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code. |

## Findings

| Severity | Found | Pending | Resolved |
|---|---|---|---|
| Critical | 0 | 0 | 0 |
| Major | 1 | 0 | 0 |
| Medium | 0 | 0 | 0 |
| Minor | 0 | 0 | 0 |
| Informational | 0 | 0 | 0 |
| Total | 1 | 0 | 0 |

# Social Media Checks

| Social Media | URL | Result |
|---|---|---|
| Twitter | https://twitter.com/Tweety_Inu | Pass |
| Other | https://medium.com/@TweetyInu | Pass |
| Website | http://tweetyinu.com | Pass |
| Telegram | https://t.me/TweetyInuOfficial | Pass |

We recommend to have 3 or more social media sources including a completed working websites.

**Social Media Information Notes:**

**Auditor Notes: undefined**

**Project Owner Notes:**

# Assessment Results

## Score Results

| Review | Score |
|---|---|
| Overall Score | 80/100 |
| Auditor Score | 81/100 |
| **Review by Section** | **Score** |
| Manual Scan Score | 32/51 |
| SWC Scan Score | 37 /37 |
| Advance Check Score | 11 /18 |

The Following Score System Has been Added to this page to help understand the value of the audit, the maximun score is 100, however to attain that value the project most pass and provide all the data needed for the assessment. Our Passing Score has been changed to 80 Points, if a project does not attain 80% is an automatic failure. Read our notes and final assessment below.

## Audit Passed

## Assessment Results

## Important Notes:

- No issues or vulnerabilities were found.
- The contract is using SafeMath, this is no longer needed after 0.8.x was introduced.
- The contract is missing some required functions.
- During the simulation there was a few delays in the code execution, this maybe due to double validation.
- please review and update code accordingly.

**Auditor Score =81**
**Audit Passed**

# Appendix

## Finding Categories

### Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that actagainst the nature of decentralization, such as explicit ownership or specialized access roles incombination with a mechanism to relocate funds.

### Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimalEVM opcodes resulting in a reduction on the total gas cost of a transaction.

### Logical Issue

Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on howblock.timestamp works.

### Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functionsbeing invoke-able by anyone under certain circumstances.

### Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that mayresult in a vulnerability.

### Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to makethe codebase more legible and, as a result, easily maintainable.

### Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code,such as a constructor assignment imposing different require statements on the input variables than a setterfunction.

### Coding Best Practices

ERC 20 Conding Standards are a set of rules that each developer should follow to ensure the code meet a set of creterias and is readable by all the developers.

# Disclaimer

CFGNINJA has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocation for the Project, and users relying on this report should not consider this as having any merit for financial advice in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or deprecation of technologies.

All information provided in this report does not constitute financial or investment advice, nor should it be used to signal that any persons reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided 'as is, and CFGNINJA is under no covenant to audited completeness, accuracy, or solidity of the contracts. In no event will CFGNINJA or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment services provided by CFGNINJA are subject to dependencies and are under continuing development. You agree that your access or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The services may access, and depend upon, multiple layers of third parties.