auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Bowser Inu

DATED : 14 April 23'

AUDIT SUMMARY

Project name – Bowser Inu

Date: 14 April, 2023

Scope of Audit- Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

Audit Status: Passed With Critical Risk

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	1	1	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1- Manual Review:
a line by line code review has been performed by audit ace team.

2- BSC Test Network:
all tests were done on BSC Test network, each test has its transaction has attached to it.

3- Slither :
The code has undergone static analysis using Slither.

Testnet Link:
https://testnet.bscscan.com/token/0x86a1c324D842379d4D577096768eaBff6EfC7D74

Token Information

Token Name : Bowser Inu

Token Symbol: BOWSER

Decimals: 18

Token Supply: 100,000,000,000,000

Token Address:
0xd69E9D9510da587Ea5Fa7038d963774538bca701

Checksum:
2e11115598ed9120a0119940ad98a1b45f5d6a9c

Owner:
0x0EE13b44c1995B1f3f369baaD33464128765FE0A
(at time of audit)

TOKEN OVERVIEW

Fees:

Buy Fees: 15%

Sell Fees: 15%

Transfer Fees: 0%

Fees Privilege: Owner

Ownership: Owned

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: including and excluding form fee - changing swap threshold - enabling trades - modifying fees - changing max wallet/buy/sell/transfers

AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.

- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.

- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.

- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.

- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.

- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST

- ✓ Return values of low-level calls
- ✓ Private modifier
- ✓ Multiple Sends
- ✓ Using Suicide
- ✓ Gas Limitand Loops
- ✓ Address hardcoded
- ✓ Exception Disorder
- ✓ Using inline assembly
- ✓ Divide before multiply
- ✓ Missing Zero Address Validation
- ✓ Compiler version not fixed
- ✓ **Gasless Send**
- ✓ Using block.timestamp
- ✓ Re-entrancy
- ✓ Tautology or contradiction
- ✓ Timestamp Dependence
- ✓ Revert/require functions
- ✓ Use of tx.origin
- ✓ Integer overflow/underflow
- ✓ Dangerous strict equalities
- ✓ Using SHA3
- ✓ Using throw

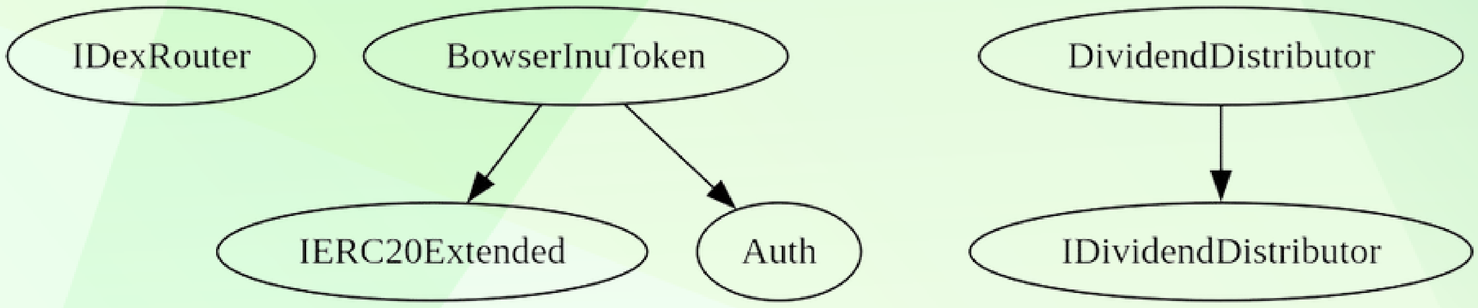
CLASSIFICATION OF RISK

Severity	Description
🟥 Critical	These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
🟫 High-Risk	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
🟧 Medium-Risk	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
🟨 Low-Risk	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
🟦 Gas Optimization /Suggestion	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
🟥 Critical	1
🟫 High-Risk	1
🟧 Medium-Risk	0
🟨 Low-Risk	0
🟦 Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to set buy/sell fees over 15%
- Owner is not able to set transfer fees (0% always)
- Owner is not able to set max buy/sell/transfer/hold amount
- Owner is not able to blacklist an arbitrary wallet
- **Owner is able to disable trades**
- Owner is not able to mint new tokens
- **Owner must enable trading for investors**

CONTRACT ASSESMENT

```
| Contract | Type | Bases | | |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
| └ | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
||||||
| **SafeMath** | Library | |||
| └ | tryAdd | Internal 🔒 | | ||
| └ | trySub | Internal 🔒 | | ||
| └ | tryMul | Internal 🔒 | | ||
| └ | tryDiv | Internal 🔒 | | ||
| └ | tryMod | Internal 🔒 | | ||
| └ | add | Internal 🔒 | | ||
| └ | sub | Internal 🔒 | | ||
| └ | mul | Internal 🔒 | | ||
| └ | div | Internal 🔒 | | ||
| └ | mod | Internal 🔒 | | ||
| └ | sub | Internal 🔒 | | ||
| └ | div | Internal 🔒 | | ||
| └ | mod | Internal 🔒 | | ||
||||||
| **IDexFactory** | Interface | |||
| └ | createPair | External ❗ | 🛑 |NO ❗ |
||||||
| **IDexRouter** | Interface | |||
| └ | factory | External ❗ | |NO ❗ |
| └ | WETH | External ❗ | |NO ❗ |
| └ | addLiquidityETH | External ❗ | 💵 |NO ❗ |
| └ | swapExactETHForTokensSupportingFeeOnTransferTokens | External ❗ | 💵 |NO ❗ |
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑 |NO ❗ |
||||||
| **IERC20Extended** | Interface | |||
| └ | totalSupply | External ❗ | |NO ❗ |
| └ | decimals | External ❗ | |NO ❗ |
| └ | symbol | External ❗ | |NO ❗ |
| └ | name | External ❗ | |NO ❗ |
| └ | balanceOf | External ❗ | |NO ❗ |
| └ | transfer | External ❗ | 🛑 |NO ❗ |
| └ | allowance | External ❗ | |NO ❗ |
| └ | approve | External ❗ | 🛑 |NO ❗ |
| └ | transferFrom | External ❗ | 🛑 |NO ❗ |
||||||
| **Auth** | Implementation | |||
| └ | <Constructor> | Public ❗ | 🛑 |NO ❗ |
| └ | authorize | Public ❗ | 🛑 | onlyOwner |
```

CONTRACT ASSESMENT

```
| └ | unauthorize | Public ❗ | 🛑 | onlyOwner |
| └ | isOwner | Public ❗ |  |NO ❗ |
| └ | isAuthorized | Public ❗ |  |NO ❗ |
| └ | transferOwnership | Public ❗ | 🛑 | onlyOwner |
||||||
| **IDividendDistributor** | Interface |  |||
| └ | setDistributionCriteria | External ❗ | 🛑 |NO ❗ |
| └ | setShare | External ❗ | 🛑 |NO ❗ |
| └ | deposit | External ❗ | 💵 |NO ❗ |
| └ | process | External ❗ | 🛑 |NO ❗ |
| └ | claimDividend | External ❗ | 🛑 |NO ❗ |
| └ | getPaidEarnings | External ❗ |  |NO ❗ |
| └ | getUnpaidEarnings | External ❗ |  |NO ❗ |
| └ | totalDistributed | External ❗ |  |NO ❗ |
||||||
| **DividendDistributor** | Implementation | IDividendDistributor |||
| └ | <Constructor> | Public ❗ | 🛑 |NO ❗ |
| └ | setDistributionCriteria | External ❗ | 🛑 | onlyToken |
| └ | setShare | External ❗ | 🛑 | onlyToken |
| └ | deposit | External ❗ | 💵 | onlyToken |
| └ | process | External ❗ | 🛑 | onlyToken |
| └ | shouldDistribute | Internal 🔒 |  | |
| └ | distributeDividend | Internal 🔒 | 🛑 | |
| └ | claimDividend | External ❗ | 🛑 |NO ❗ |
| └ | getPaidEarnings | Public ❗ |  |NO ❗ |
| └ | getUnpaidEarnings | Public ❗ |  |NO ❗ |
| └ | getCumulativeDividends | Internal 🔒 |  | |
| └ | addShareholder | Internal 🔒 | 🛑 | |
| └ | removeShareholder | Internal 🔒 | 🛑 | |
||||||
| **BowserInuToken** | Implementation | IERC20Extended, Auth |||
| └ | <Constructor> | Public ❗ | 🛑 | Auth |
| └ | <Receive Ether> | External ❗ | 💵 |NO ❗ |
| └ | totalSupply | External ❗ |  |NO ❗ |
| └ | decimals | External ❗ |  |NO ❗ |
| └ | symbol | External ❗ |  |NO ❗ |
| └ | name | External ❗ |  |NO ❗ |
| └ | balanceOf | Public ❗ |  |NO ❗ |
| └ | allowance | External ❗ |  |NO ❗ |
| └ | approve | Public ❗ | 🛑 |NO ❗ |
| └ | approveMax | External ❗ | 🛑 |NO ❗ |
| └ | transfer | External ❗ | 🛑 |NO ❗ |
```

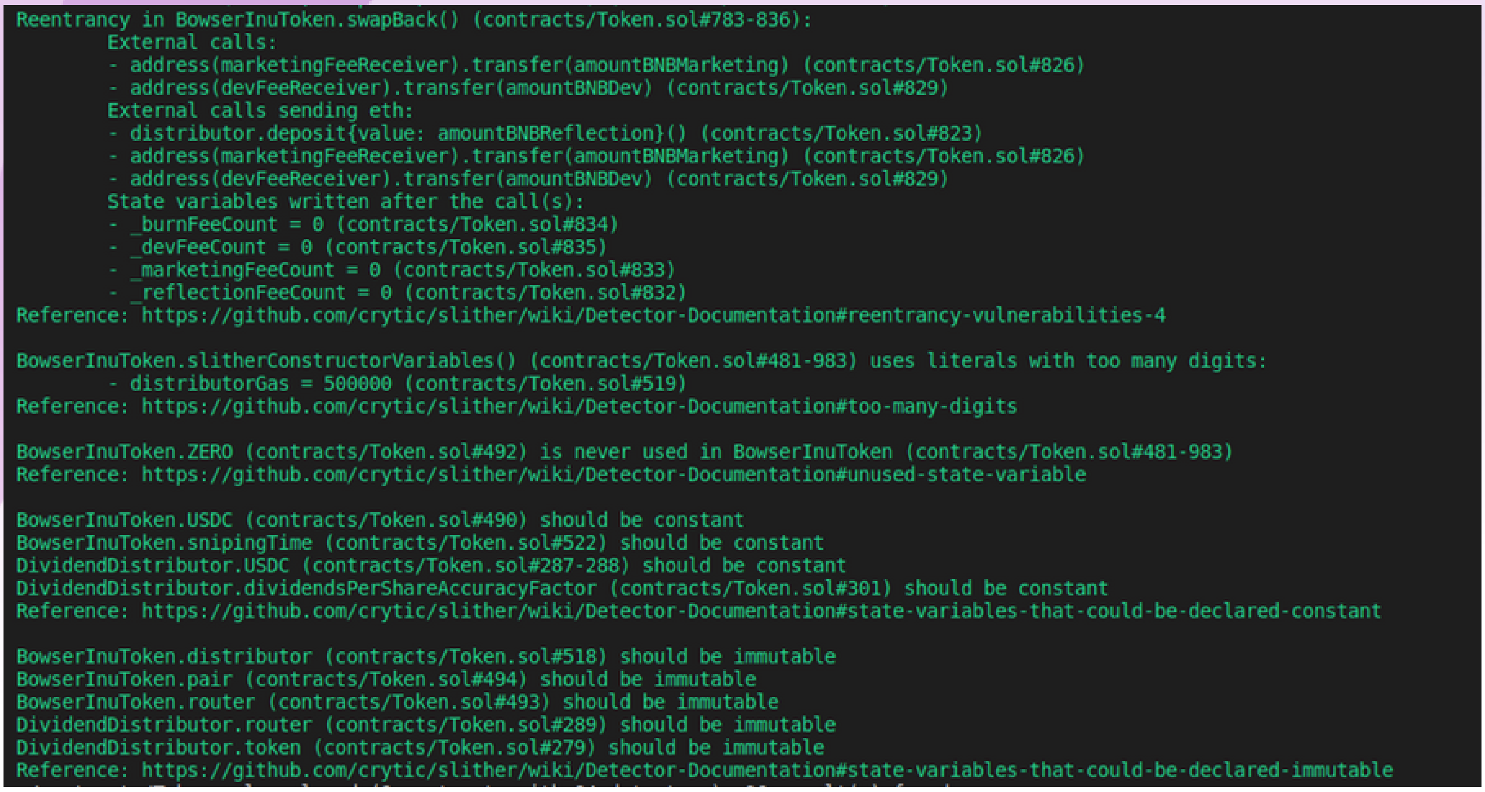
CONTRACT ASSESMENT

```
| └ | transferFrom | External ❗ | 🛑 |NO ❗ |
| └ | _transferFrom | Internal 🔒 | 🛑 | |
| └ | _basicTransfer | Internal 🔒 | 🛑 | |
| └ | takeFee | Internal 🔒 | 🛑 | |
| └ | setBuyAccFee | Internal 🔒 | 🛑 | |
| └ | setSellAccFee | Internal 🔒 | 🛑 | |
| └ | shouldSwapBack | Internal 🔒 | | |
| └ | swapBack | Internal 🔒 | 🛑 | swapping |
| └ | enableTrading | External ❗ | 🛑 | authorized |
| └ | claimDividend | External ❗ | 🛑 |NO ❗ |
| └ | getPaidDividend | Public ❗ | |NO ❗ |
| └ | getUnpaidDividend | External ❗ | |NO ❗ |
| └ | getTotalDistributedDividend | External ❗ | |NO ❗ |
| └ | removeStuckBnb | External ❗ | 🛑 | authorized |
| └ | setIsDividendExempt | External ❗ | 🛑 | authorized |
| └ | setIsFeeExempt | External ❗ | 🛑 | authorized |
| └ | setIsLimitExempt | External ❗ | 🛑 | authorized |
| └ | removeBots | External ❗ | 🛑 | onlyOwner |
| └ | setIsWalletExempt | External ❗ | 🛑 | authorized |
| └ | setBuyFees | Public ❗ | 🛑 | authorized |
| └ | setSellFees | Public ❗ | 🛑 | authorized |
| └ | setFeeReceivers | External ❗ | 🛑 | authorized |
| └ | setMaxWalletlimit | External ❗ | 🛑 | authorized |
| └ | setMaxTxnLimit | External ❗ | 🛑 | authorized |
| └ | setSwapBackSettings | External ❗ | 🛑 | authorized |
| └ | setDistributionCriteria | External ❗ | 🛑 | authorized |
| └ | setDistributorSettings | External ❗ | 🛑 | authorized |
```

Legend

```
| Symbol | Meaning |
|:--------:|-----------|
| 🛑 | Function can modify state |
| 💵 | Function is payable |
```

STATIC ANALYSIS



```
Reentrancy in BowserInuToken.swapBack() (contracts/Token.sol#783-836):
        External calls:
        - address(marketingFeeReceiver).transfer(amountBNBMarketing) (contracts/Token.sol#826)
        - address(devFeeReceiver).transfer(amountBNBDev) (contracts/Token.sol#829)
        External calls sending eth:
        - distributor.deposit{value: amountBNBReflection}() (contracts/Token.sol#823)
        - address(marketingFeeReceiver).transfer(amountBNBMarketing) (contracts/Token.sol#826)
        - address(devFeeReceiver).transfer(amountBNBDev) (contracts/Token.sol#829)
        State variables written after the call(s):
        - _burnFeeCount = 0 (contracts/Token.sol#834)
        - _devFeeCount = 0 (contracts/Token.sol#835)
        - _marketingFeeCount = 0 (contracts/Token.sol#833)
        - _reflectionFeeCount = 0 (contracts/Token.sol#832)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

BowserInuToken.slitherConstructorVariables() (contracts/Token.sol#481-983) uses literals with too many digits:
        - distributorGas = 500000 (contracts/Token.sol#519)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

BowserInuToken.ZERO (contracts/Token.sol#492) is never used in BowserInuToken (contracts/Token.sol#481-983)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

BowserInuToken.USDC (contracts/Token.sol#490) should be constant
BowserInuToken.snipingTime (contracts/Token.sol#522) should be constant
DividendDistributor.USDC (contracts/Token.sol#287-288) should be constant
DividendDistributor.dividendsPerShareAccuracyFactor (contracts/Token.sol#301) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

BowserInuToken.distributor (contracts/Token.sol#518) should be immutable
BowserInuToken.pair (contracts/Token.sol#494) should be immutable
BowserInuToken.router (contracts/Token.sol#493) should be immutable
DividendDistributor.router (contracts/Token.sol#289) should be immutable
DividendDistributor.token (contracts/Token.sol#279) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

Result => A static analysis of contract's source code has been performed using slither,

No major issues were found in the output

FUNCTIONAL TESTING

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x8fb90e8a5e02c134423eced5b19efd63bc7bb7255e68ab32234ff8fd9b679e5c

2- Buying when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x883b0a2927d5adeb5144145d1bb6b02e05faf3b59ac50dc492327520acbfeb3e

3- Selling when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x4f04d6441cea4d69908aff0a374832acf3860dc0b07b95ad557774e332a028ee

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xa0038c66f910ccaedd08cf370473972be27b73907a2c131fabd8444b4672dc70

5- Buying when not excluded from fees (up to 15% tax) (passed):
https://testnet.bscscan.com/tx/0x64855af4f7b663c3937b0d5e0c0b2ec58a786ccc590d54b1befc34064ebd7ee3

6- Selling when not excluded from fees (up to 15% tax) (passed):
https://testnet.bscscan.com/tx/0xe9648f56b9b4e26e3741b124f20f65e4e2af8f4928bd78f6753561b306e254ed

7- Transferring when not excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x2a337442d4cf404fa34b445012a9e05c81a1090fe2341a0d8a96ee69796b217e

FUNCTIONAL TESTING

8- Internal swap (passed):
fee wallets received BNB
https://testnet.bscscan.com/address/0xD7973B7baf14699646AebeF631875c65DAcc493F#internaltx

9- Distribution of rewards (passed):
reward tokens are distributed between holders, this can be seen in this transaction
https://testnet.bscscan.com/tx/0xe9648f56b9b4e26e3741b124f20f65e4e2af8f4928bd78f6753561b306e254ed

MANUAL TESTING

Centralization – Unbounded max wallet and max sell/buys/transfers

Severity: Critical

Function: setMaxWalletlimit - setMaxTxnLimit

Lines: 959 - 963

Status: Not Resolved

Overview:

The current implementation does not have minimum safeguards for max wallet and max/buy/sell/transfers amounts. Setting max wallet to 0 means buys/transfers would be disabled, and by setting maxTxnAmount to zero, all actions would be disabled.

```
function setMaxWalletlimit(uint256 _maxWalletAmount) external authorized {
    maxWalletAmount = _maxWalletAmount;
}
function setMaxTxnLimit(uint256 _maxTxnAmount) external authorized {
    maxTxnAmount = _maxTxnAmount;
}
```

Recommendation:

Ensure that the maxWalletAmount and maxTxnAmount are greater than a reasonable value (0.1% of supply based on PinkSale SAFU criteria). Additionally, include appropriate error messages to provide feedback to the user.

Example:

```
function setMaxWalletlimit(uint256 _maxWalletAmount) external authorized {
        require(_maxWalletAmount > totalSupply() / 1000);
    maxWalletAmount = _maxWalletAmount;
}
function setMaxTxnLimit(uint256 _maxTxnAmount) external authorized {
        require(_maxWalletAmount > totalSupply() / 1000);
    maxTxnAmount = _maxTxnAmount;
}
```

MANUAL TESTING

Centralization – Owner must enable trading

Severity: High

Function: enableTrading

Lines: 813

Status: Not Resolved

Overview:

The owner must activate trading for investors to buy, sell, or transfer tokens. If trading remains disabled, token holders will be unable to trade their tokens.

```
function enableTrading() external authorized {
    require(!trading, "LYKOICare: already enabled");
    trading = true;
    swapEnabled = true;
    launchedAt = block.timestamp;
}
```

Recommendation:

To address this issue, consider the following options:

- transfer ownership of contract to a trusted 3rd wallet (pinksale safu developer) to guarante enabling of trades
- Incorporate a safety mechanism that allows investors to activate trading if a specified duration has elapsed since the conclusion of the presale or consider alternative ways such as allowing trades ater investors claimed their presale tokens.

MANUAL TESTING

Centralization - Setting swap threshold to 0 can disable sell/transfers

Severity: Critical

Function: swapBack

Lines: 783

Status: Not Resolved

Overview:

setting swapThreshold to 0 can disable sell/transfers, this is because contract tries to swap 0 tokens for BNB which will revert the whole transaction (INSUFFICIENT_INPUT_AMOUNT Error)

```
function swapBack() internal swapping {
    uint256 totalFee = _reflectionFeeCount
        .add(_marketingFeeCount)
        .add(_burnFeeCount)
        .add(_devFeeCount);
    if (totalFee == 0) return;

    uint256 amountBurn = swapThreshold.mul(_burnFeeCount).div(totalFee);
    uint256 amountToSwap = swapThreshold.sub(amountBurn);
    .
        .
        .

function setSwapBackSettings(
    bool _enabled,
    uint256 _amount
) external authorized {
    swapEnabled = _enabled;
    swapThreshold = _amount;
}
```

MANUAL TESTING

Recommendation:

make sure that swapThreshold is always more than 0 and less than a reasonable max limit

```
function setSwapBackSettings(
    bool _enabled,
    uint256 _amount
) external authorized {
           require(_amount > 0, “cant set swap threshold to 0””);
    swapEnabled = _enabled;
    swapThreshold = _amount;
}
```

MANUAL TESTING

Informational – No way to withdraw stuck tokens

Severity: Informational
Function: ---
Lines: ---
Status: Not Resolved
Overview:
Currently threre are no functions to withdraw ERC20 tokens from the contract. If tokens are sent to the contract by mistake there will not be anyway to withdraw them.

Recommendation:
to address this issue implement a function to be able to withdraw ERC20 tokens from the contract

DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token. The Auditace team disclaims any liability for the resulting losses.

ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.