AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Batman Inu

DATED : 16 Apr 23'

AUDIT SUMMARY

Project name – Batman Inu

Date: 16 April, 2023

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

Audit Status: **Passed with High Risk**

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	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 : Static Analysis

Testnet Link: all tests were done using this contract, tests are done on BSC Testnet

https://testnet.bscscan.com/token/0x39e094b67e23de5eebf3353a2bbe3e536438a476#code

Token Information

Token Name : Batman Inu

Token Symbol: BATMANINU

Decimals: 9

Token Supply: 10,000,000,000

Token Address:
0xf2C0142a804434b8b1c812F3e801819DdF730619

Checksum:
2bd4a4a8f6957b3559bd90d9cf4e82ff23d2506e

Owner:
0xb7708A7ddFD9C69a14AaE4D5746C45E355591Afe
(at time of audit)

Deployer:
0xb7708A7ddFD9C69a14AaE4D5746C45E355591Afe

TOKEN OVERVIEW

Fees:

Buy Fees: up to 36%

Sell Fees: up to 36%

Transfer Fees: 0%

Fees Privilige: 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 - modifying fees

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

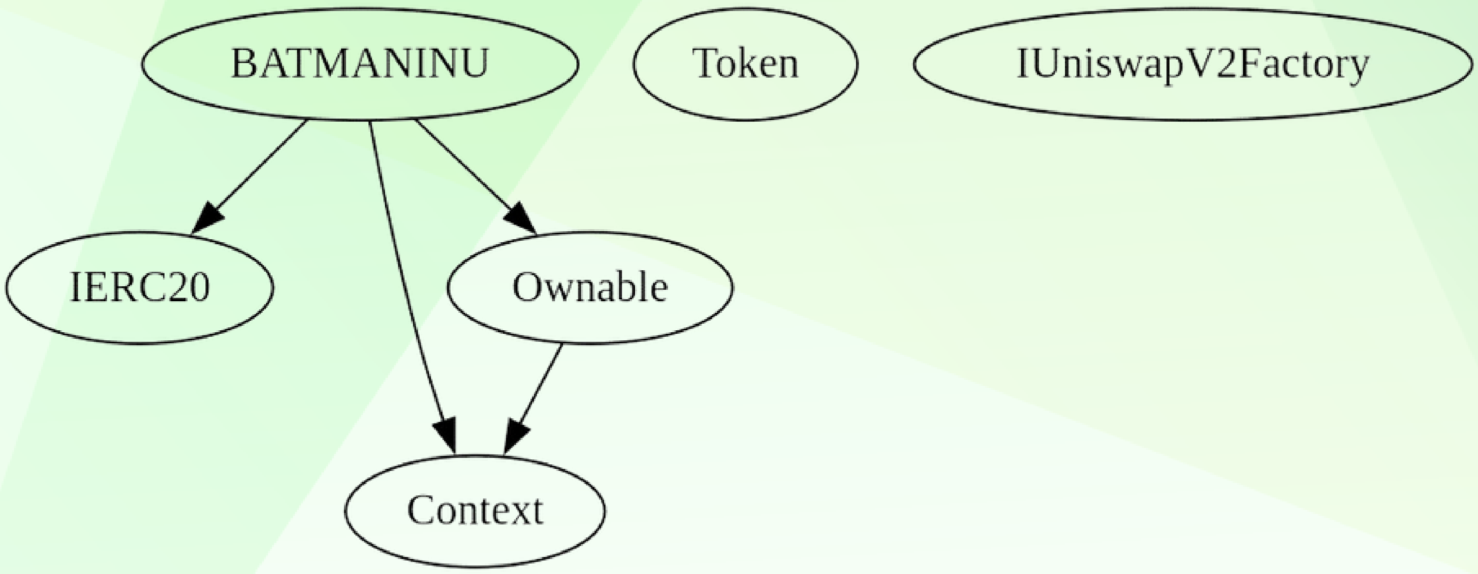
CLASSIFICATION OF RISK

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	0
🟫 High-Risk	1
🟧 Medium-Risk	0
🟨 Low-Risk	0
🟦 Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is able to set buy/sell fees up to 36% each (72% max tax)
- Owner is not ablet o set transfer taxes (0% forever)
- Owner is not able to set max buy/sell/transfer/hold amount
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to disable trades
- Owner is not able to mint new tokens

CONTRACT ASSESMENT

```
| Contract |      Type      |     Bases     |              |             |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
|    └    | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
||||||
| **IERC20** | Interface | |||
| └ | totalSupply | External ❗️ |   |NO❗️ |
| └ | balanceOf | External ❗️ |   |NO❗️ |
| └ | transfer | External ❗️ | 🛑  |NO❗️ |
| └ | allowance | External ❗️ |   |NO❗️ |
| └ | approve | External ❗️ | 🛑  |NO❗️ |
| └ | transferFrom | External ❗️ | 🛑  |NO❗️ |
||||||
| **Token** | Interface | |||
| └ | transferFrom | External ❗️ | 🛑  |NO❗️ |
| └ | transfer | External ❗️ | 🛑  |NO❗️ |
||||||
| **IUniswapV2Factory** | Interface | |||
| └ | createPair | External ❗️ | 🛑  |NO❗️ |
||||||
| **IUniswapV2Router02** | Interface | |||
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗️ | 🛑  |NO❗️ |
| └ | factory | External ❗️ |   |NO❗️ |
| └ | WETH | External ❗️ |   |NO❗️ |
| └ | addLiquidityETH | External ❗️ |  💵 |NO❗️ |
||||||
| **Context** | Implementation | |||
| └ | _msgSender | Internal 🔒 |   | |
||||||
| **SafeMath** | Library | |||
| └ | add | Internal 🔒 |   | |
| └ | sub | Internal 🔒 |   | |
| └ | sub | Internal 🔒 |   | |
| └ | mul | Internal 🔒 |   | |
| └ | div | Internal 🔒 |   | |
| └ | div | Internal 🔒 |   | |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗️ | 🛑  |NO❗️ |
| └ | owner | Public ❗️ |   |NO❗️ |
| └ | renounceOwnership | Public ❗️ | 🛑  | onlyOwner |
| └ | transferOwnership | Public ❗️ | 🛑  | onlyOwner |
||||||
```

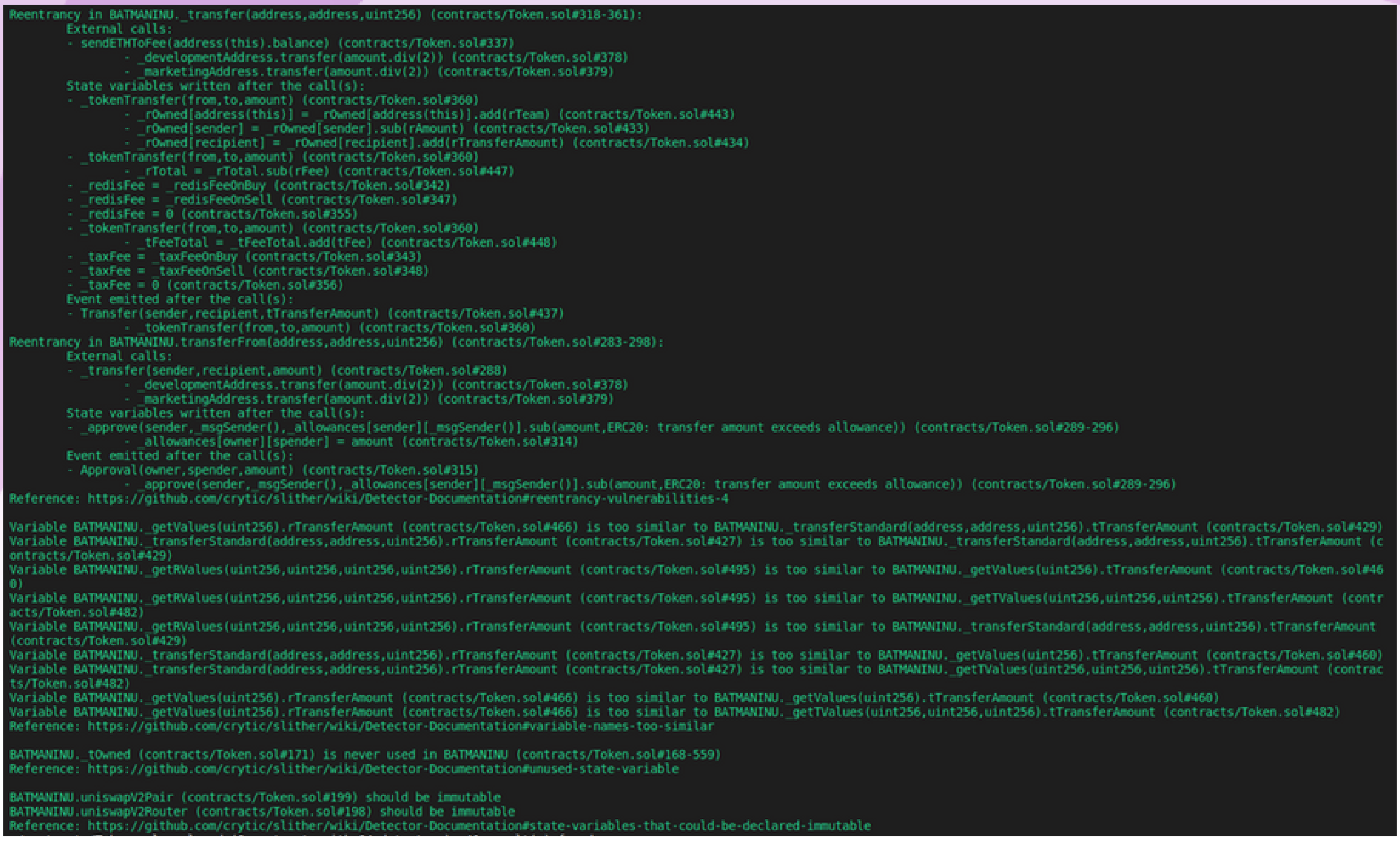
CONTRACT ASSESMENT

```
| **BATMANINU** | Implementation | Context, IERC20, Ownable |||
| └ | <Constructor> | Public ❗️ | 🛑  |NO❗️ |
| └ | name | Public ❗️ |   |NO❗️ |
| └ | symbol | Public ❗️ |   |NO❗️ |
| └ | decimals | Public ❗️ |   |NO❗️ |
| └ | totalSupply | Public ❗️ |   |NO❗️ |
| └ | balanceOf | Public ❗️ |   |NO❗️ |
| └ | transfer | Public ❗️ | 🛑  |NO❗️ |
| └ | allowance | Public ❗️ |   |NO❗️ |
| └ | approve | Public ❗️ | 🛑  |NO❗️ |
| └ | transferFrom | Public ❗️ | 🛑  |NO❗️ |
| └ | tokenFromReflection | Private 🔐 |   | |
| └ | _approve | Private 🔐 | 🛑  | |
| └ | _transfer | Private 🔐 | 🛑  | |
| └ | swapTokensForEth | Private 🔐 | 🛑  | lockTheSwap |
| └ | sendETHToFee | Private 🔐 | 🛑  | |
| └ | _tokenTransfer | Private 🔐 | 🛑  | |
| └ | rescueForeignTokens | Public ❗️ | 🛑  | onlyDev |
| └ | setNewDevAddress | Public ❗️ | 🛑  | onlyDev |
| └ | setNewMarketingAddress | Public ❗️ | 🛑  | onlyDev |
| └ | _transferStandard | Private 🔐 | 🛑  | |
| └ | _takeTeam | Private 🔐 | 🛑  | |
| └ | _reflectFee | Private 🔐 | 🛑  | |
| └ | <Receive Ether> | External ❗️ | 💵 |NO❗️ |
| └ | _getValues | Private 🔐 |   | |
| └ | _getTValues | Private 🔐 |   | |
| └ | _getRValues | Private 🔐 |   | |
| └ | _getRate | Private 🔐 |   | |
| └ | _getCurrentSupply | Private 🔐 |   | |
| └ | manualswap | External ❗️ | 🛑  |NO❗️ |
| └ | manualsend | External ❗️ | 🛑  |NO❗️ |
| └ | setFee | Public ❗️ | 🛑  | onlyDev |
| └ | toggleSwap | Public ❗️ | 🛑  | onlyDev |
| └ | excludeMultipleAccountsFromFees | Public ❗️ | 🛑  | onlyOwner |
```

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS



```
Reentrancy in BATMANINU._transfer(address,address,uint256) (contracts/Token.sol#318-361):
        External calls:
        - sendETHToFee(address(this).balance) (contracts/Token.sol#337)
                - _developmentAddress.transfer(amount.div(2)) (contracts/Token.sol#378)
                - _marketingAddress.transfer(amount.div(2)) (contracts/Token.sol#379)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount) (contracts/Token.sol#360)
                - _rOwned[address(this)] = _rOwned[address(this)].add(rTeam) (contracts/Token.sol#443)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (contracts/Token.sol#433)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (contracts/Token.sol#434)
        - _tokenTransfer(from,to,amount) (contracts/Token.sol#360)
                - _rTotal = _rTotal.sub(rFee) (contracts/Token.sol#447)
        - _redisFee = _redisFeeOnBuy (contracts/Token.sol#342)
        - _redisFee = _redisFeeOnSell (contracts/Token.sol#347)
        - _redisFee = 0 (contracts/Token.sol#355)
        - _tokenTransfer(from,to,amount) (contracts/Token.sol#360)
                - _tFeeTotal = _tFeeTotal.add(tFee) (contracts/Token.sol#448)
        - _taxFee = _taxFeeOnBuy (contracts/Token.sol#343)
        - _taxFee = _taxFeeOnSell (contracts/Token.sol#348)
        - _taxFee = 0 (contracts/Token.sol#356)
        Event emitted after the call(s):
        - Transfer(sender,recipient,tTransferAmount) (contracts/Token.sol#437)
                - _tokenTransfer(from,to,amount) (contracts/Token.sol#360)
Reentrancy in BATMANINU.transferFrom(address,address,uint256) (contracts/Token.sol#283-298):
        External calls:
        - _transfer(sender,recipient,amount) (contracts/Token.sol#288)
                - _developmentAddress.transfer(amount.div(2)) (contracts/Token.sol#378)
                - _marketingAddress.transfer(amount.div(2)) (contracts/Token.sol#379)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (contracts/Token.sol#289-296)
                - _allowances[owner][spender] = amount (contracts/Token.sol#314)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (contracts/Token.sol#315)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (contracts/Token.sol#289-296)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4

Variable BATMANINU._getValues(uint256).rTransferAmount (contracts/Token.sol#466) is too similar to BATMANINU._transferStandard(address,address,uint256).tTransferAmount (contracts/Token.sol#429)
Variable BATMANINU._transferStandard(address,address,uint256).rTransferAmount (contracts/Token.sol#427) is too similar to BATMANINU._transferStandard(address,address,uint256).tTransferAmount (c
ontracts/Token.sol#429)
Variable BATMANINU._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (contracts/Token.sol#495) is too similar to BATMANINU._getValues(uint256).tTransferAmount (contracts/Token.sol#46
0)
Variable BATMANINU._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (contracts/Token.sol#495) is too similar to BATMANINU._getTValues(uint256,uint256,uint256).tTransferAmount (contr
acts/Token.sol#482)
Variable BATMANINU._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (contracts/Token.sol#495) is too similar to BATMANINU._transferStandard(address,address,uint256).tTransferAmount
(contracts/Token.sol#429)
Variable BATMANINU._transferStandard(address,address,uint256).rTransferAmount (contracts/Token.sol#427) is too similar to BATMANINU._getValues(uint256).tTransferAmount (contracts/Token.sol#460)
Variable BATMANINU._transferStandard(address,address,uint256).rTransferAmount (contracts/Token.sol#427) is too similar to BATMANINU._getTValues(uint256,uint256,uint256).tTransferAmount (contrac
ts/Token.sol#482)
Variable BATMANINU._getValues(uint256).rTransferAmount (contracts/Token.sol#466) is too similar to BATMANINU._getValues(uint256).tTransferAmount (contracts/Token.sol#460)
Variable BATMANINU._getValues(uint256).rTransferAmount (contracts/Token.sol#466) is too similar to BATMANINU._getTValues(uint256,uint256,uint256).tTransferAmount (contracts/Token.sol#482)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

BATMANINU._tOwned (contracts/Token.sol#171) is never used in BATMANINU (contracts/Token.sol#168-559)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

BATMANINU.uniswapV2Pair (contracts/Token.sol#199) should be immutable
BATMANINU.uniswapV2Router (contracts/Token.sol#198) 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

Router (PCS V2):
0xD99D1c33F9fC3444f8101754aBC46c52416550D1

All the functionalities have been tested, no issues were found

1- Adding liquidity **(passed):**
https://testnet.bscscan.com/tx/0x691e3e17ef1cac47c560c1bc479b8865d2d1958594352a81c1dad288a69948c6

2- Buying when excluded (0% tax) **(passed):**
https://testnet.bscscan.com/tx/0x3601329ebef4f4a2374cc7246f66e7a40d8d5a122bbb3eab42f2153d22ff0fc5

3- Selling when excluded (0% tax) **(passed):**
https://testnet.bscscan.com/tx/0x45e82025092dad24a4b0b414fb5625f3d4ad5de48e00c38065d25b697ce7a3a3

4- Transferring when excluded (0% tax) **(passed):**
https://testnet.bscscan.com/tx/0x1b24273efd938b856585bd051686049a79a3fe7d943d2c3797a805ef7d707da9

5- Buying when not excluded (upto 36% tax) **(passed):**
https://testnet.bscscan.com/tx/0x729ed6d5d2ab346c93dce1d4e31fba26acd17726f2913fc7a7d3342d94d4793d

6- Selling when not excluded (upto 36% tax) **(passed):**
https://testnet.bscscan.com/tx/0x7acad8342331331704dad758b3f2ae1a075c1657dbf26e5450936681e9fd0a4e

FUNCTIONAL TESTING

7- Transferring when not excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x4a5bbd7d95a11925c17a3b119042277a4dd7d36d2f54bfb8f594cd39d2cacc04

8- Internal swap (passed):
Marketing and development wallets received BNB
https://testnet.bscscan.com/address/0xB55Dd75C228eD63d41417E083140157904c78DDF#internaltx
https://testnet.bscscan.com/address/0xD1Ed4833ED38d8A584475d4C3c10C90CefdB743D#internaltx

MANUAL TESTING

Centralization - Excessive buy/sell fees

Severity: High
Function: setFee
Lines: 393
Status: Not Resolved
Overview:
Current implementation of the code allows owner to set up to 36% fee for buy and sells seperatly.

```
function setFee(uint256 redisFeeOnBuy, uint256 redisFeeOnSell, uint256        taxFeeOnBuy, uint256 taxFeeOnSell)
public onlyDev {
                require(redisFeeOnBuy < 11, "Redis cannot be more than 10.");
                require(redisFeeOnSell < 11, "Redis cannot be more than 10.");
                require(taxFeeOnBuy < 26, "Tax cannot be more than 25.");
                require(taxFeeOnSell < 26, "Tax cannot be more than 25.");
        _redisFeeOnBuy = redisFeeOnBuy;
        _redisFeeOnSell = redisFeeOnSell;
        _taxFeeOnBuy = taxFeeOnBuy;
        _taxFeeOnSell = taxFeeOnSell;
}
```

Recommendation:
According to pinksale safu critieria, sum of buy + sell tax should not exceed 25% in total in order to be considered a safu token

```
function setFee(uint256 redisFeeOnBuy, uint256 redisFeeOnSell, uint256        taxFeeOnBuy, uint256 taxFeeOnSell)
public onlyDev {
                require(redisFeeOnBuy < 26, "Redis cannot be more than 10.");
                require(redisFeeOnSell < 26, "Redis cannot be more than 10.");
                require(taxFeeOnBuy < 26, "Tax cannot be more than 25.");
                require(taxFeeOnSell < 26, "Tax cannot be more than 25.");
                require(_redisFeeOnBuy + _redisFeeOnSell + _taxFeeBuy + _taxFeeOnSell            < 26, "Total Fee
should not exceed 25%.");
        _redisFeeOnBuy = redisFeeOnBuy;
        _redisFeeOnSell = redisFeeOnSell;
        _taxFeeOnBuy = taxFeeOnBuy;
        _taxFeeOnSell = taxFeeOnSell;
}
```

DISCLAIMER

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.

https://auditace.tech/

https://t.me/Audit_Ace

https://twitter.com/auditace_

https://github.com/Audit-Ace