auditace.tech

AuditAce
FROM INCEPTION TO SUCCESS

# Smart Contract Audit

FOR

# Cat Grok Inu

DATED : 17 Dec 23'

# AUDIT SUMMARY

**Project name** – Cat Grok Inu

**Date**: 17 Dec, 2023

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

**Audit Status: Passed**

## Issues Found

| Status | Critical | High | Medium | Low | Suggestion |
|---|---|---|---|---|---|
| Open | 0 | 0 | 0 | 1 | 1 |
| 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 conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

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

**Testnet version:**
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:
https://testnet.bscscan.com/address/0x5ff46498a4a9f916fab5866ea0415b98a5988a22#code

# Token Information

**Token Address:**
0xE82902d6F6c2AD5083dF8Ada46E223c3549aD47e

**Name:** Cat Grok Inu

**Symbol:** Cat Grok

**Decimals:** 18

**Network:** Binance Smart Chain

**Token Type:** BEP-20

**Owner:** 0x0Ba6EdAA629A0977e683eDa2d26F3c874dfB5cc7

**Deployer:**
0x0Ba6EdAA629A0977e683eDa2d26F3c874dfB5cc7

**Token Supply:** 300000000000000000000000000

**Checksum:** aEde641126e217b2b455d49e77fc4197

**Testnet:**
https://testnet.bscscan.com/address/0x5ff46498a4a9f916fab5866ea0415b98a5988a22#code

# TOKEN OVERVIEW

**Buy Fee:** 5-25%

**Sell Fee:** 5-25%

**Transfer Fee:** 0-0%

**Fee Privilege:** Owner

**Ownership:** Owned

**Minting:** None

**Max Tx:** Yes

**Blacklist:** No

# 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

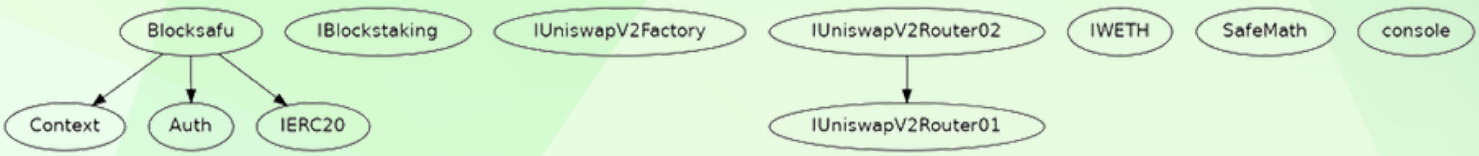
# CLASSIFICATION OF RISK

| 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 | 0 |
| Medium-Risk | 0 |
| Low-Risk | 1 |
| Gas Optimization / Suggestions | 1 |

# INHERITANCE TREE



# POINTS TO NOTE

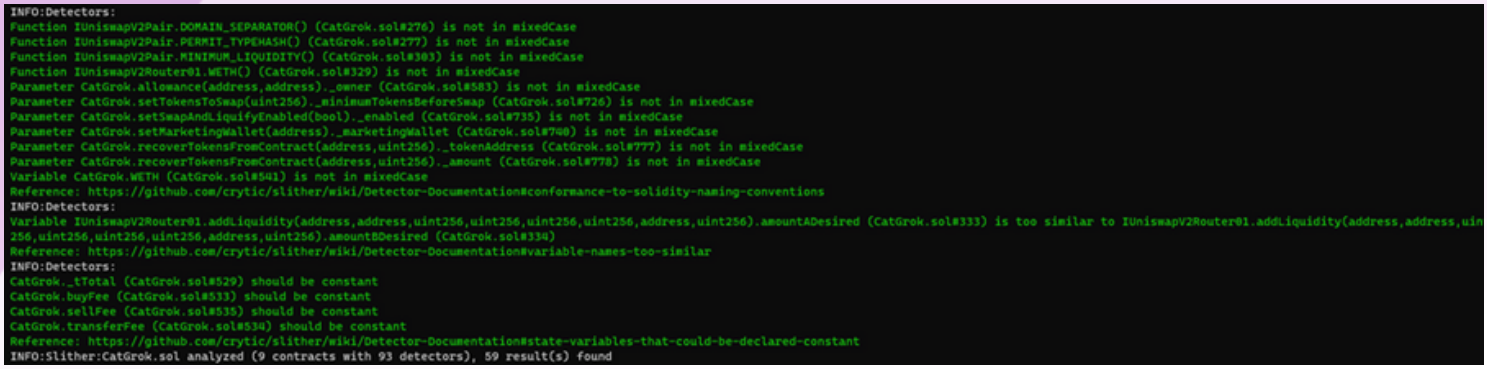
- The owner can transfer ownership.
- The owner can renounce ownership.
- The owner can exclude/include wallets from fees.
- The owner can enable/disable swapping.

# STATIC ANALYSIS

```
INFO:Detectors:
CatGrok.transferToAddressETH(address,uint256) (CatGrok.sol#745-752) uses a dangerous strict equality:
        - amount == 0 (CatGrok.sol#749)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dangerous-strict-equalities
INFO:Detectors:
CatGrok.constructor().currentRouter (CatGrok.sol#551) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
CatGrok.allowance(address,address)._owner (CatGrok.sol#583) shadows:
        - Ownable._owner (CatGrok.sol#197) (state variable)
CatGrok._approve(address,address,uint256)._owner (CatGrok.sol#631) shadows:
        - Ownable._owner (CatGrok.sol#197) (state variable)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
Reentrancy in CatGrok._transfer(address,address,uint256) (CatGrok.sol#637-675):
        External calls:
        - swapAndLiquify() (CatGrok.sol#657)
                - (succ) = recipient.call{value: amount}() (CatGrok.sol#750)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (CatGrok.sol#688-694)
        External calls sending eth:
        - swapAndLiquify() (CatGrok.sol#657)
                - (succ) = recipient.call{value: amount}() (CatGrok.sol#750)
        State variables written after the call(s):
        - marketingTokensCollected += fee (CatGrok.sol#671)
        - totalMarketingTokensCollected += fee (CatGrok.sol#672)
Reentrancy in CatGrok.swapAndLiquify() (CatGrok.sol#676-682):
        External calls:
        - swapTokensForEth(totalTokens) (CatGrok.sol#678)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (CatGrok.sol#688-694)
        - transferToAddressETH(marketingWallet,ethBalance) (CatGrok.sol#680)
                - (succ) = recipient.call{value: amount}() (CatGrok.sol#750)
        External calls sending eth:
        - transferToAddressETH(marketingWallet,ethBalance) (CatGrok.sol#680)
                - (succ) = recipient.call{value: amount}() (CatGrok.sol#750)
        State variables written after the call(s):
        - marketingTokensCollected = 0 (CatGrok.sol#681)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in CatGrok._transfer(address,address,uint256) (CatGrok.sol#637-675):
        External calls:
        - swapAndLiquify() (CatGrok.sol#657)
                - (succ) = recipient.call{value: amount}() (CatGrok.sol#750)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (CatGrok.sol#688-694)
        External calls sending eth:
        - swapAndLiquify() (CatGrok.sol#657)
                - (succ) = recipient.call{value: amount}() (CatGrok.sol#750)
        Event emitted after the call(s):
        - Transfer(sender,recipient,amount) (CatGrok.sol#704)
                - _tokenTransfer(from,address(this),fee) (CatGrok.sol#670)
        - Transfer(sender,recipient,amount) (CatGrok.sol#704)
```

```
INFO:Detectors:
Address._revert(bytes,string) (CatGrok.sol#182-194) uses assembly
        - INLINE ASM (CatGrok.sol#187-190)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
CatGrok.excludeFromFee(address) (CatGrok.sol#709-716) compares to a boolean constant:
        -require(bool,string)(_isExcludedFromFee[account] != true,The wallet is already excluded!) (CatGrok.sol#710-713)
CatGrok.includeInFee(address) (CatGrok.sol#717-724) compares to a boolean constant:
        -require(bool,string)(_isExcludedFromFee[account] != false,The wallet is already included!) (CatGrok.sol#718-721)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality
INFO:Detectors:
Address._revert(bytes,string) (CatGrok.sol#182-194) is never used and should be removed
Address.functionCall(address,bytes) (CatGrok.sol#53-64) is never used and should be removed
Address.functionCall(address,bytes,string) (CatGrok.sol#65-71) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (CatGrok.sol#72-84) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (CatGrok.sol#85-105) is never used and should be removed
Address.functionDelegateCall(address,bytes) (CatGrok.sol#131-141) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (CatGrok.sol#142-155) is never used and should be removed
Address.functionStaticCall(address,bytes) (CatGrok.sol#106-116) is never used and should be removed
Address.functionStaticCall(address,bytes,string) (CatGrok.sol#117-130) is never used and should be removed
Address.isContract(address) (CatGrok.sol#39-41) is never used and should be removed
Address.sendValue(address,uint256) (CatGrok.sol#42-52) is never used and should be removed
Address.verifyCallResult(bool,bytes,string) (CatGrok.sol#171-181) is never used and should be removed
Address.verifyCallResultFromTarget(address,bool,bytes,string) (CatGrok.sol#156-170) is never used and should be removed
CatGrok.swapETHForTokens(uint256) (CatGrok.sol#755-768) is never used and should be removed
Context._msgData() (CatGrok.sol#10-12) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version0.8.19 (CatGrok.sol#5) necessitates a version too recent to be trusted. Consider deploying with 0.8.18.
solc-0.8.19 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (CatGrok.sol#42-52):
        - (success) = recipient.call{value: amount}() (CatGrok.sol#47)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (CatGrok.sol#85-105):
        - (success,returndata) = target.call{value: value}(data) (CatGrok.sol#95-97)
Low level call in Address.functionStaticCall(address,bytes,string) (CatGrok.sol#117-130):
        - (success,returndata) = target.staticcall(data) (CatGrok.sol#122)
Low level call in Address.functionDelegateCall(address,bytes,string) (CatGrok.sol#142-155):
        - (success,returndata) = target.delegatecall(data) (CatGrok.sol#147)
Low level call in CatGrok.transferToAddressETH(address,uint256) (CatGrok.sol#745-752):
        - (succ) = recipient.call{value: amount}() (CatGrok.sol#750)
Low level call in CatGrok.recoverETHfromContract() (CatGrok.sol#769-775):
        - (succ) = address(marketingWallet).call{value: ethBalance}() (CatGrok.sol#771)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (CatGrok.sol#276) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (CatGrok.sol#277) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (CatGrok.sol#303) is not in mixedCase
Function IUniswapV2Router01.WETH() (CatGrok.sol#329) is not in mixedCase
```

# STATIC ANALYSIS



```
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (CatGrok.sol#276) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (CatGrok.sol#277) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (CatGrok.sol#303) is not in mixedCase
Function IUniswapV2Router01.WETH() (CatGrok.sol#329) is not in mixedCase
Parameter CatGrok.allowance(address,address)._owner (CatGrok.sol#583) is not in mixedCase
Parameter CatGrok.setTokensToSwap(uint256)._minimumTokensBeforeSwap (CatGrok.sol#726) is not in mixedCase
Parameter CatGrok.setSwapAndLiquifyEnabled(bool)._enabled (CatGrok.sol#735) is not in mixedCase
Parameter CatGrok.setMarketingWallet(address)._marketingWallet (CatGrok.sol#740) is not in mixedCase
Parameter CatGrok.recoverTokensFromContract(address,uint256)._tokenAddress (CatGrok.sol#777) is not in mixedCase
Parameter CatGrok.recoverTokensFromContract(address,uint256)._amount (CatGrok.sol#778) is not in mixedCase
Variable CatGrok.WETH (CatGrok.sol#541) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (CatGrok.sol#333) is too similar to IUniswapV2Router01.addLiquidity(address,address,uin
256,uint256,uint256,uint256,address,uint256).amountBDesired (CatGrok.sol#334)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
CatGrok._tTotal (CatGrok.sol#529) should be constant
CatGrok.buyFee (CatGrok.sol#533) should be constant
CatGrok.sellFee (CatGrok.sol#535) should be constant
CatGrok.transferFee (CatGrok.sol#534) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Slither:CatGrok.sol analyzed (9 contracts with 93 detectors), 59 result(s) found
```

**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- Approve (passed):**

https://testnet.bscscan.com/tx/0x793f60db331394332800c426be2c21d8174e0b463aa476ea2fe8a82f15a5477b

**2- Increase Allowance (passed):**

https://testnet.bscscan.com/tx/0xa9e2ae73da120cd7baa5afa604aa280bc2e9c43caea0821a7b06b69a928a4d20

**3- Decrease Allowance (passed):**

https://testnet.bscscan.com/tx/0x20c8b7949174e96e10c971637ab0c4ee9850d378ea2586059fdf7af9f77a5cea

**4- Recover Eth from the contract (passed):**

https://testnet.bscscan.com/tx/0x1bfd8cb94b992975a813da93ceceab7b4f527923bef5239fcc4190ef197ce459

**5- Set swap And liquify Enabled (passed):**

https://testnet.bscscan.com/tx/0xc64eb31a46437bef9b7562d6d5fa3523c281adc36a072e9df1c1ea7f512cf757

# MANUAL TESTING

## Centralization – Missing Zero Address

**Severity: Low**

**Subject: Zero Check**

**Status: Open**

**Overview:**

functions can take a zero address as a parameter (0x00000...). If a function parameter of address type is not properly validated by checking for zero addresses, there could be serious consequences for the contract's functionality.
function excludeFromFee(address account) external

```
onlyOwner {
  require(
    _isExcludedFromFee[account] != true,
    "The wallet is already excluded!"
  );
  _isExcludedFromFee[account] = true;
  emit ExcludeStatus(account, true);
}
```

# MANUAL TESTING

## Optimization

**Severity: Optimization**

**Subject: Remove unused code.**

**Status: Open**

**Overview:**
Unused variables are allowed in Solidity, and they do. not pose a direct security issue. It is the best practice. though to avoid them.

```
event AuditLog(string, address);
event UpdateStakingWallet(address);
event UpdateBuyFee(uint256);
event UpdateSellFee(uint256);
event UpdateTransferFee(uint256);
event TransferStatus(bool);
event UpdateDistribution(uint256, uint256);
event TradingStarted(bool);
```

# 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.


**https://auditace.tech/**

**https://t.me/Audit_Ace**

**https://twitter.com/auditace_**

**https://github.com/Audit-Ace**