expelee

Building the Futuristic Blockchain Ecosystem

SECURITY AUDIT REPORT

Doge King

TOKEN OVERVIEW

Risk Findings

Severity	Found
High	1
Medium	1
Low	0
Informational	1

Centralization Risks

Owner Privileges	Description
Can Owner Set Taxes >25% ?	Not Detected
Owner needs to enable trading ?	Detected
Can Owner Disable Trades ?	Not Detected
Can Owner Mint ?	Not Detected
Can Owner Blacklist ?	Not Detected
Can Owner set Max Wallet amount ?	Not Detected
Can Owner Set Max TX amount ?	Not Detected

TABLE OF CONTENTS

OVERVIEW

The Expelee team has performed a line-by-line manual analysis and automated review of the smart contract. The smart contract was analysed mainly for common smart contract vulnerabilities, exploits, and manipulation hacks. According to the smart contract audit:

Audit Result	**Passed with high risk**
Audit Date	**25 May 2024**

CONTRACT DETAILS

Token Address: 0x56346C19F15CC2a27e6b2991c176Bba7E9554fbB

Name: Doge King

Symbol: DogeKing

Decimals: 18

Network: BscScan

Token Type: BEP-20

Owner: 0x91BA890c9D5242ea576a9B60D3dc4A7e03275335

Deployer: 0x91BA890c9D5242ea576a9B60D3dc4A7e03275335

Token Supply: 1000000000

Checksum: A2032c616934aeb47e6039f76b20d311

Testnet:
https://testnet.bscscan.com/address/0x6c1c66780b157c64f9665b27020882a6676d6eef#code

AUDIT METHODOLOGY

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch , that lead to scams and rugpulls.

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

Tools

- Manual Review: The code has undergone a line-by-line review by the Ace team.
- 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.
- Slither: The code has undergone static analysis using Slither.

VULNERABILITY CHECKS

Design Logic	Passed
Compiler warnings	Passed
Private user data leaks	Passed
Timestamps dependence	Passed
Integer overflow and underflow	Passed
Race conditions & reentrancy. Cross-function race conditions	Passed
Possible delays in data delivery	Passed
Oracle calls	Passed
Front Running	Passed
DoS with Revert	Passed
DoS with block gas limit	Passed
Methods execution permissions	Passed
Economy model	Passed
Impact of the exchange rate on the logic	Passed
Malicious event log	Passed
Scoping and declarations	Passed
Uninitialized storage pointers	Passed
Arithmetic accuracy	Passed
Cross-function race conditions	Passed
Safe Zepplin module	Passed

RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and acces control issues within the code. The exploitation of these issues by malicious actors may cause serious financial damage to projects that failed to get an audit in time. We categorize these vulnerabilities by the following levels:

High Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

Medium Risk

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

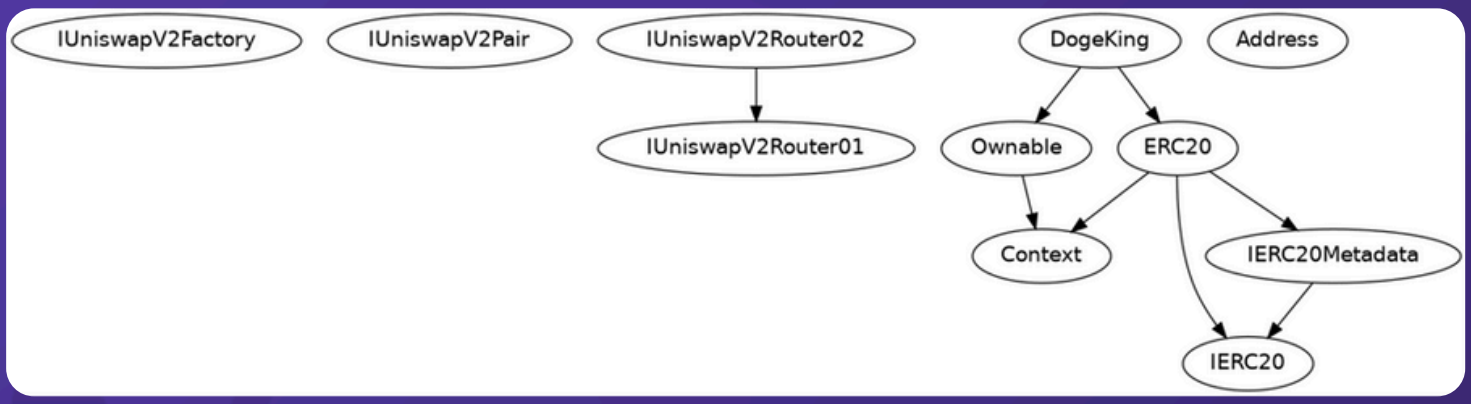
Low Risk

Issues on this level are minor details and warning that can remain unfixed.

Informational

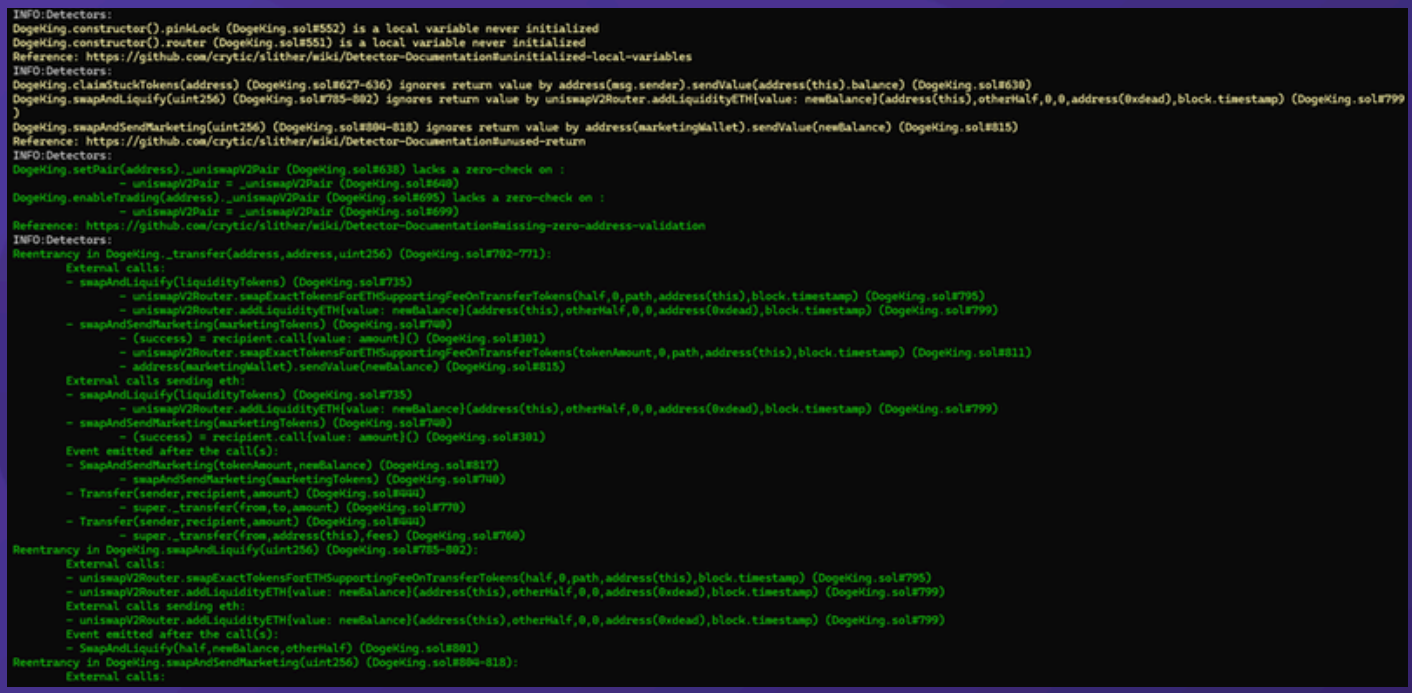
Issues on this level are minor details and warning that can remain unfixed.

INHERITANCE TREE

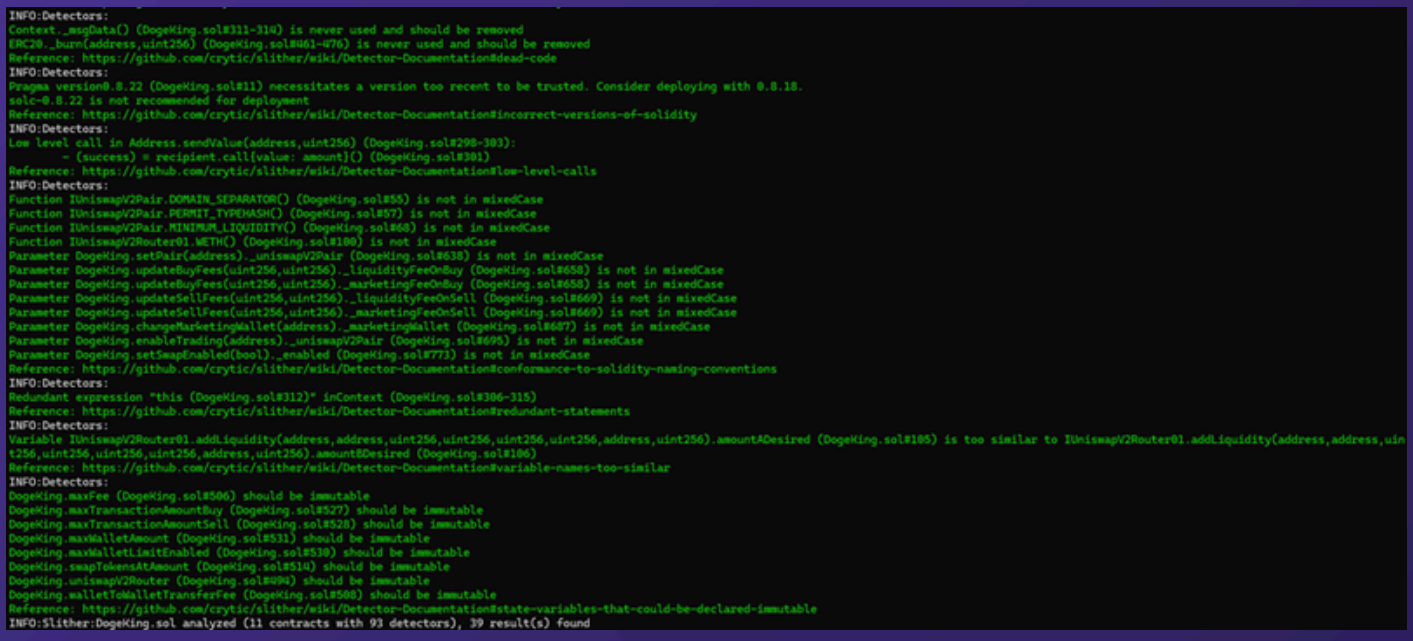


expelee

STATIC ANALYSIS



```
INFO:Detectors:
DogeKing.constructor().pinkLock (DogeKing.sol#552) is a local variable never initialized
DogeKing.constructor().router (DogeKing.sol#551) is a local variable never initialized
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables
INFO:Detectors:
DogeKing.claimStuckTokens(address) (DogeKing.sol#627-636) ignores return value by address(msg.sender).sendValue(address(this).balance) (DogeKing.sol#630)
DogeKing.swapAndLiquify(uint256) (DogeKing.sol#785-802) ignores return value by uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (DogeKing.sol#799
)
DogeKing.swapAndSendMarketing(uint256) (DogeKing.sol#804-818) ignores return value by address(marketingWallet).sendValue(newBalance) (DogeKing.sol#815)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
DogeKing.setPair(address)._uniswapV2Pair (DogeKing.sol#638) lacks a zero-check on :
        - uniswapV2Pair = _uniswapV2Pair (DogeKing.sol#640)
DogeKing.enableTrading(address)._uniswapV2Pair (DogeKing.sol#695) lacks a zero-check on :
        - uniswapV2Pair = _uniswapV2Pair (DogeKing.sol#699)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in DogeKing._transfer(address,address,uint256) (DogeKing.sol#702-771):
        External calls:
        - swapAndLiquify(liquidityTokens) (DogeKing.sol#735)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestamp) (DogeKing.sol#795)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (DogeKing.sol#799)
        - swapAndSendMarketing(marketingTokens) (DogeKing.sol#740)
                - (success) = recipient.call{value: amount}() (DogeKing.sol#381)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (DogeKing.sol#811)
                - address(marketingWallet).sendValue(newBalance) (DogeKing.sol#815)
        External calls sending eth:
        - swapAndLiquify(liquidityTokens) (DogeKing.sol#735)
                - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (DogeKing.sol#799)
        - swapAndSendMarketing(marketingTokens) (DogeKing.sol#740)
                - (success) = recipient.call{value: amount}() (DogeKing.sol#381)
        Event emitted after the call(s):
        - SwapAndSendMarketing(tokenAmount,newBalance) (DogeKing.sol#817)
                - swapAndSendMarketing(marketingTokens) (DogeKing.sol#740)
        - Transfer(sender,recipient,amount) (DogeKing.sol#444)
                - super._transfer(from,to,amount) (DogeKing.sol#770)
        - Transfer(sender,recipient,amount) (DogeKing.sol#444)
                - super._transfer(from,address(this),fees) (DogeKing.sol#760)
Reentrancy in DogeKing.swapAndLiquify(uint256) (DogeKing.sol#785-802):
        External calls:
        - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(half,0,path,address(this),block.timestamp) (DogeKing.sol#795)
        - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (DogeKing.sol#799)
        External calls sending eth:
        - uniswapV2Router.addLiquidityETH{value: newBalance}(address(this),otherHalf,0,0,address(0xdead),block.timestamp) (DogeKing.sol#799)
        Event emitted after the call(s):
        - SwapAndLiquify(half,newBalance,otherHalf) (DogeKing.sol#801)
Reentrancy in DogeKing.swapAndSendMarketing(uint256) (DogeKing.sol#804-818):
        External calls:
```



```
INFO:Detectors:
Context._msgData() (DogeKing.sol#311-314) is never used and should be removed
ERC20._burn(address,uint256) (DogeKing.sol#461-476) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version0.8.22 (DogeKing.sol#11) necessitates a version too recent to be trusted. Consider deploying with 0.8.18.
solc-0.8.22 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) (DogeKing.sol#298-303):
        - (success) = recipient.call{value: amount}() (DogeKing.sol#381)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (DogeKing.sol#55) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (DogeKing.sol#57) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (DogeKing.sol#68) is not in mixedCase
Function IUniswapV2Router01.WETH() (DogeKing.sol#100) is not in mixedCase
Parameter DogeKing.setPair(address)._uniswapV2Pair (DogeKing.sol#638) is not in mixedCase
Parameter DogeKing.updateBuyFees(uint256,uint256)._liquidityFeeOnBuy (DogeKing.sol#658) is not in mixedCase
Parameter DogeKing.updateBuyFees(uint256,uint256)._marketingFeeOnBuy (DogeKing.sol#658) is not in mixedCase
Parameter DogeKing.updateSellFees(uint256,uint256)._liquidityFeeOnSell (DogeKing.sol#669) is not in mixedCase
Parameter DogeKing.updateSellFees(uint256,uint256)._marketingFeeOnSell (DogeKing.sol#669) is not in mixedCase
Parameter DogeKing.changeMarketingWallet(address)._marketingWallet (DogeKing.sol#687) is not in mixedCase
Parameter DogeKing.enableTrading(address)._uniswapV2Pair (DogeKing.sol#695) is not in mixedCase
Parameter DogeKing.setSwapEnabled(bool)._enabled (DogeKing.sol#773) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (DogeKing.sol#312)" inContext (DogeKing.sol#306-315)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (DogeKing.sol#105) is too similar to IUniswapV2Router01.addLiquidity(address,address,uin
t256,uint256,uint256,uint256,address,uint256).amountBDesired (DogeKing.sol#106)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar
INFO:Detectors:
DogeKing.maxFee (DogeKing.sol#506) should be immutable
DogeKing.maxTransactionAmountBuy (DogeKing.sol#527) should be immutable
DogeKing.maxTransactionAmountSell (DogeKing.sol#528) should be immutable
DogeKing.maxWalletAmount (DogeKing.sol#531) should be immutable
DogeKing.maxWalletLimitEnabled (DogeKing.sol#538) should be immutable
DogeKing.swapTokensAtAmount (DogeKing.sol#514) should be immutable
DogeKing.uniswapV2Router (DogeKing.sol#494) should be immutable
DogeKing.walletToWalletTransferFee (DogeKing.sol#508) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:DogeKing.sol analyzed (11 contracts with 93 detectors), 39 result(s) found
```

TESTNET VERSION

1- **Approve (passed):**
https://testnet.bscscan.com/tx/0x4183cc2c40d76d0a606113c91ebf79a8e9b75272f4d0f866836b6c2fbf9d2626

2- Increase Allowance (passed):
https://testnet.bscscan.com/tx/0x5c13dca833648c8e17f9c2d1b7ec7c3047739a0773e1af4d4566d9d5d19d58a4

3- Decrease Allowance (passed):
https://testnet.bscscan.com/tx/0xa265ebcb239e6cf9f96c5bb2d32ba04d2589badc2831088b94cd3b38bd58bf91

4- Change Marketing Wallet (passed):
https://testnet.bscscan.com/tx/0x8a01f8b927f3f4f766e1bf2fdb970b62a20d154405b7c97cd1eb2791c74f0506

5- Update Buy Fees (passed):
https://testnet.bscscan.com/tx/0xecf36e62c4c79fe8f7bf2fd05b91c0e97149cf9ba6698edf7c95af86889c9ef2

6- Update Sell Fees (passed):
https://testnet.bscscan.com/tx/0xff15a929f2bccc83e270ac4be1f92bc2600b3dee2700dcad817068bf418abffe

MANUAL REVIEW

Severity Criteria

Expelee assesses the severity of disclosed vulnerabilities according to methodology based on OWASP standarts.

Vulnerabilities are dividend into three primary risk categroies:
High
Medium
Low

High-level considerations for vulnerabilities span the following key areas when conducting assessments:

- Malicious input handling
- Escalation of privileges
- Arithmetic
- Gas use

Overall Risk Severity				
Impact	HIGH	Medium	High	Critical
	MEDIUM	Low	Medium	High
	LOW	Note	Low	Medium
		LOW	MEDIUM	HIGH
	Likelihood			

HIGH RISK FINDING

Centralization – Enabling Trades

Severity: High

Function: EnableTrading

Status: Open

Overview:

The EnableTrading function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

```
function enableTrading() external onlyOwner{
    require(!tradingEnabled, "CSLT: Trading already enabled.");
    tradingEnabled = true;
    swapEnabled = true;

    emit TradingEnabled(tradingEnabled);
  }
```

Suggestion:

To reduce centralization and potential manipulation, consider one of the following approaches:

1.Automatically enable trading after a specified condition, such as the completion of a presale, is met.

2.If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can give investors more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad-faith actions by the original owner.

MEDIUM RISK FINDING

Centralization – Missing Require Check.
Severity: Medium
Function: change Marketing Wallet
Status: Open

Overview:
The owner can set any arbitrary address excluding zero address as this is not recommended because if the owner will set the address to the contract address, then the Eth will not be sent to that address and the transaction will fail and this will lead to a potential honeypot in the contract.

```
function changeMarketingWallet(address _marketingWallet) external
onlyOwner {
    require(_marketingWallet != marketingWallet, "Marketing wallet is already
that address");
    require(_marketingWallet != address(0), "Marketing wallet cannot be the
zero address");
    marketingWallet = _marketingWallet;

    emit MarketingWalletChanged(marketingWallet);
  }
```

Suggestion: It is recommended that the address should not be able to set as a
contract address.

INFORMATIONAL & OPTIMIZATIONS

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.

```
interface IUniswapV2Factory {
  event PairCreated(address indexed token0, address indexed token1, address pair, uint);

  function feeTo() external view returns (address);

  function feeToSetter() external view returns (address);

  function getPair(address tokenA, address tokenB) external view returns (address pair);

  function allPairs(uint) external view returns (address pair);

  function allPairsLength() external view returns (uint);

  function createPair(address tokenA, address tokenB) external returns (address pair);

  function setFeeTo(address) external;

  function setFeeToSetter(address) external;
}
```

INFORMATIONAL & OPTIMIZATIONS

```
function _msgData() internal view virtual returns (bytes calldata) {
 this;
// silence state mutability warning without generating bytecode - see
https://github.com/ethereum/solidity/issues/2691
 return msg.data;
 }
```

ABOUT EXPELEE

Expelee is a product-based aspirational Web3 start-up. Coping up with numerous solutions for blockchain security and constructing a Web3 ecosystem from deal making platform to developer hosting open platform, while also developing our own commercial and sustainable blockchain.

www.expelee.com

expeleeofficial

expelee

Expelee

expelee

expelee_official

expelee-co

expelee

Building the Futuristic Blockchain Ecosystem

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 guarantess against the sale of team tokens or the removal of liquidity by the project audited in this document.

Always do your own research and project yourselves from being scammed. The Expelee 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 Expelee receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Alway 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 Expelee team disclaims any liability for the resulting losses.