

# expelee

Building the Futuristic **Blockchain Ecosystem**




## Security Audit Report FOR



## Space Doge

# 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:

 <b>Audit Result</b>	<b>Passed with High Risk</b>
 <b>KYC Verification</b>	-
 <b>Audit Date</b>	<b>4 Feb 2023</b>

**Audit Passed With High Risk**

**-Team Expelee**

# PROJECT DESCRIPTION

## Space Doge

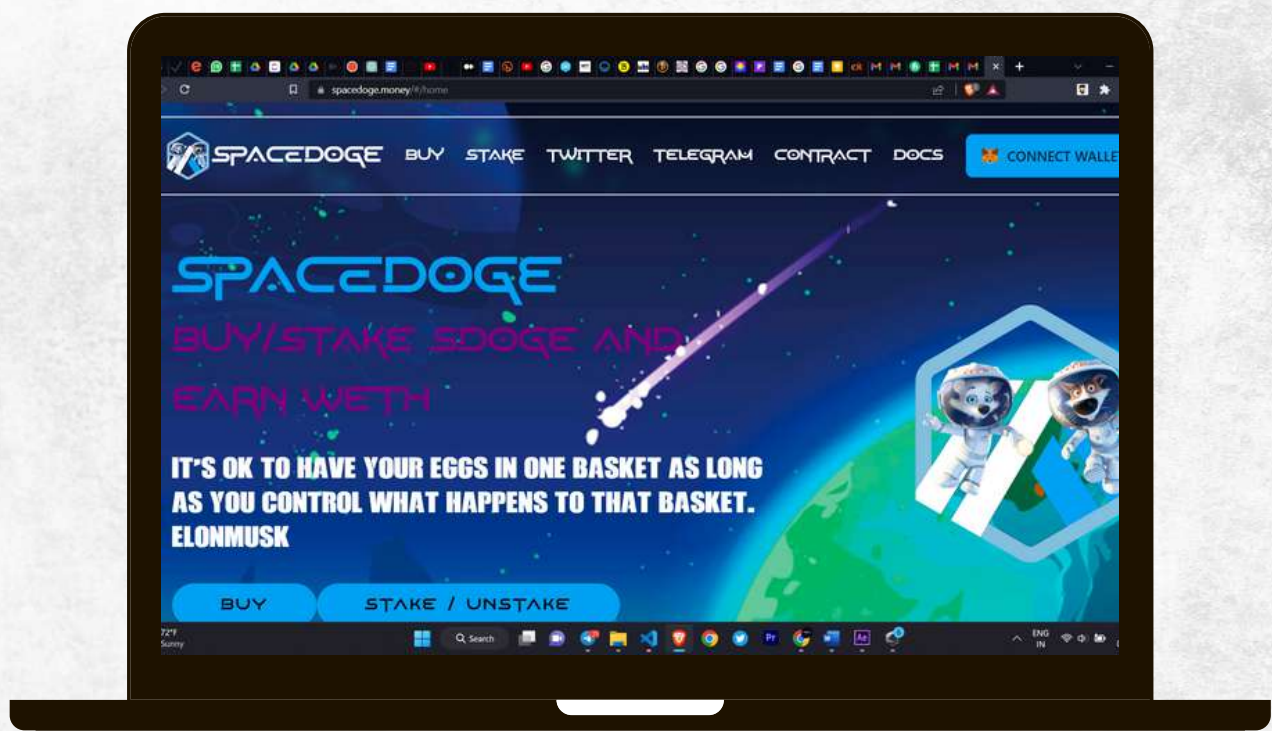
Buy/Stake sdoge and Earn WETH





# Social Media Profiles

## Space Doge



 <https://spacedoge.money/#/home>

 [https://t.me/spacedoge\\_money](https://t.me/spacedoge_money)

 [https://twitter.com/spacedoge\\_money](https://twitter.com/spacedoge_money)

**It's always good to check the social profiles of the project,  
before making your investment.**

**-Team Expelee**

# CONTRACT DETAILS

Token Name

**SpaceDoge**

---

Symbol

**sDOGE**

---

Network

**Arbitrum**

---

Language

**Solidity**

---

Contract Address (Verified)

**0x9c52a1BABd62973BaBa7C81B412994A31d4aD31a**

---

Token Type

**ERC20**

---

Total Supply

**10,000,000**

---

Compiler

**v0.6.12+commit.27d51765**

---

Optimization Enabled

**Yes with 200 runs**

---

Contract SHA-256 Checksum:

**d3a2080b82afbec6ed19b46a7bfb6f809f88ef88daf39a4aa86952c456848abd**

---

Owner's Wallet

**0x943B3a966Ed16a361551ADBA9C444f7176a0fFd0**

---

Deployer's Wallet

**0x943B3a966Ed16a361551ADBA9C444f7176a0fFd0**



# 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

- DE
- Open Zeppelin
- Code Analyzer
- Solidity Code
- Compiler
- Hardhat

# FUNCTION OVERVIEW

Can Take Back Ownership

Not Detected

Owner Change Balance

Not Detected

Blacklist

Not Detected

Modify Fees

Not Detected

Proxy

Not Detected

Whitelisted

Not Detected

Anti Whale

Not Detected

Trading Cooldown

Not Detected

Transfer Pausable

Not Detected

Cannot Sell All

Not Detected

Hidden Owner

Not Detected

Mint

Not Detected



# VULNERABILITY CHECKLIST

Design Logic	Passed
Compiler warnings.	Passed
Private user data leaks	Passed
Timestamp 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 Zeppelin module	Passed
Fallback function security	Passed



# RISK CLASSIFICATION

When performing smart contract audits, our specialists look for known vulnerabilities as well as logical and access 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

---

Information level is to offer suggestions for improvement of efficacy or security for features with a risk free factor.

# AUDIT SUMMARY

## Tools

### 1- Manual review:

A line by line code review has been performed by Expelee team.

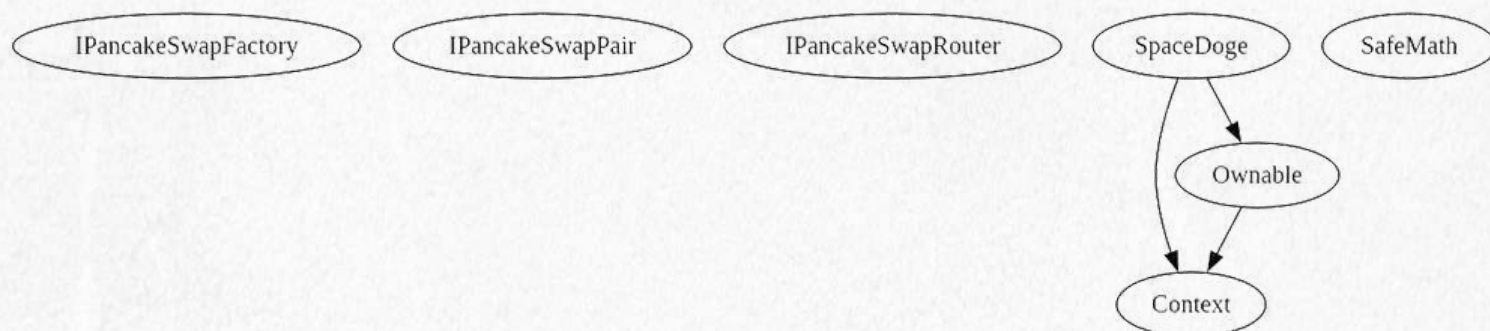
### 2- BSC Test network:

All tests were done on BSC Test network, each test has its transaction has attached to it. You can check this tests in “functionality tests” section of the report.

### 3- Slither : Static analysis

---

## Inheritance Trees:





# CONTRACT ASSESSMENT

Contract	Type	Bases			
├──────────┤├──────────┤├──────────┤├──────────┤├──────────┤					
├┬┤	**Function Name**	**Visibility**	**Mutability**	**Modifiers**	
**IPancakeSwapFactory**	Interface				
└┤ feeTo	External	!		NO	!
└┤ feeToSetter	External	!		NO	!
└┤ getPair	External	!		NO	!
└┤ allPairs	External	!		NO	!
└┤ allPairsLength	External	!		NO	!
└┤ createPair	External	!	●	NO	!
└┤ setFeeTo	External	!	●	NO	!
└┤ setFeeToSetter	External	!	●	NO	!
**IPancakeSwapPair**	Interface				
└┤ name	External	!		NO	!
└┤ symbol	External	!		NO	!
└┤ decimals	External	!		NO	!
└┤ totalSupply	External	!		NO	!
└┤ balanceOf	External	!		NO	!
└┤ allowance	External	!		NO	!
└┤ approve	External	!	●	NO	!
└┤ transfer	External	!	●	NO	!
└┤ transferFrom	External	!	●	NO	!
└┤ DOMAIN_SEPARATOR	External	!		NO	!
└┤ PERMIT_TYPEHASH	External	!		NO	!
└┤ nonces	External	!		NO	!
└┤ permit	External	!	●	NO	!

|  $\perp$  | MINIMUM\_LIQUIDITY | External ! | | NO ! |

|  $\perp$  | factory | External ! | | NO ! |

|  $\perp$  | token0 | External ! | | NO ! |

|  $\perp$  | token1 | External ! | | NO ! |

|  $\perp$  | getReserves | External ! | | NO ! |

|  $\perp$  | price0CumulativeLast | External ! | | NO ! |

|  $\perp$  | price1CumulativeLast | External ! | | NO ! |

|  $\perp$  | kLast | External ! | | NO ! |

|  $\perp$  | mint | External ! | ● | NO ! |

|  $\perp$  | burn | External ! | ● | NO ! |

|  $\perp$  | swap | External ! | ● | NO ! |

|  $\perp$  | skim | External ! | ● | NO ! |

|  $\perp$  | sync | External ! | ● | NO ! |

|  $\perp$  | initialize | External ! | ● | NO ! |

|||||

| \*\*IPancakeSwapRouter\*\* | Interface | |||

|  $\perp$  | factory | External ! | | NO ! |

|  $\perp$  | WETH | External ! | | NO ! |

|  $\perp$  | addLiquidity | External ! | ● | NO ! |

|  $\perp$  | addLiquidityETH | External ! | 🇺🇸 | NO ! |

|  $\perp$  | removeLiquidity | External ! | ● | NO ! |

|  $\perp$  | removeLiquidityETH | External ! | ● | NO ! |

|  $\perp$  | removeLiquidityWithPermit | External ! | ● | NO ! |

|  $\perp$  | removeLiquidityETHWithPermit | External ! | ● | NO ! |

|  $\perp$  | swapExactTokensForTokens | External ! | ● | NO ! |

|  $\perp$  | swapTokensForExactTokens | External ! | ● | NO ! |

|  $\perp$  | swapExactETHForTokens | External ! | 💵 | NO ! |

|  $\perp$  | swapTokensForExactETH | External ! | ● | NO ! |

|  $\perp$  | swapExactTokensForETH | External ! | ● | NO ! |

|  $\perp$  | swapETHForExactTokens | External ! | 💵 | NO ! |



```

|  | quote | External ! | | NO ! |
|  | getAmountOut | External ! | | NO ! |
|  | getAmountIn | External ! | | NO ! |
|  | getAmountsOut | External ! | | NO ! |
|  | getAmountsIn | External ! | | NO ! |
|  | removeLiquidityETHSupportingFeeOnTransferTokens | External ! | ● | NO ! |
|  | removeLiquidityETHWithPermitSupportingFeeOnTransferTokens | External ! | ● | NO ! |
|
|  | swapExactTokensForTokensSupportingFeeOnTransferTokens | External ! | ● | NO ! |
|  | swapExactETHForTokensSupportingFeeOnTransferTokens | External ! | 💵 | NO ! |
|  | swapExactTokensForETHSupportingFeeOnTransferTokens | External ! | ● | NO ! |
|||||
| **Context** | Implementation | |||
|  | <Constructor> | Internal 🔒 | ● | |
|  | _msgSender | Internal 🔒 | | |
|  | _msgData | Internal 🔒 | | |
|||||
| **Ownable** | Implementation | Context |||
|  | <Constructor> | Internal 🔒 | ● | |
|  | owner | Public ! | | NO ! |
|  | renounceOwnership | Public ! | ● | onlyOwner |
|  | transferOwnership | Public ! | ● | onlyOwner |
|  | _transferOwnership | Internal 🔒 | ● | |
|||||
| **SafeMath** | Library | |||
|  | add | Internal 🔒 | | |
|  | sub | Internal 🔒 | | |
|  | sub | Internal 🔒 | | |
|  | mul | Internal 🔒 | | |
|  | div | Internal 🔒 | | |
|  | div | Internal 🔒 | | |

```



```



|  | mod | Internal  |  |  |
|  | mod | Internal  |  |  |
|||||
| **SpaceDoge** | Implementation | Context, Ownable |||
|  | getOwner | External  |  | NO  |
|  | decimals | External  |  | NO  |
|  | symbol | External  |  | NO  |
|  | name | External  |  | NO  |
|  | totalSupply | External  |  | NO  |
|  | balanceOf | Public  |  | NO  |
|  | transfer | External  |  | NO  |
|  | allowance | External  |  | NO  |
|  | approve | External  |  | NO  |
|  | transferFrom | External  |  | NO  |
|  | increaseAllowance | Public  |  | NO  |
|  | decreaseAllowance | Public  |  | NO  |
|  | burn | External  |  | NO  |
|  | _mint | Internal  |  |  |
|  | _burn | Internal  |  |  |
|  | _approve | Internal  |  |  |
|  | _burnFrom | Internal  |  |  |
|  | <Constructor> | Public  |  | NO  |
|  | setInitialAddresses | External  |  | onlyOwner |
|  | setFeeAddresses | External  |  | onlyOwner |
|  | setMaxTxAmount | External  |  | onlyOwner |
|  | setbuyFee | External  |  | onlyOwner |
|  | setsellFee | External  |  | onlyOwner |
|  | getTotalSellFee | Public  |  | NO  |
|  | getTotalBuyFee | Public  |  | NO  |
|  | excludeAddressFromFee | External  |  | onlyOwner |



```





| <sup>L</sup> | \_transfer | Internal  |  ||

| <sup>L</sup> | swapAndLiquify | Private  |  | lockTheSwap |

| <sup>L</sup> | swapTokensForEth | Private  |  ||

| <sup>L</sup> | addLiquidity | Private  |  ||

| <sup>L</sup> | <Receive Ether> | External  |  | NO  |

## Legend

| Symbol | Meaning |

|:-----:|-----|

|  | Function can modify state |

|  | Function is payable |

# Static Analysis

A static analysis of contract's source code has been performed using slither. No major issues were found in the output

```
Context: msgData() (contracts/TestToken.sol#351-354) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/TestToken.sol#547-549) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/TestToken.sol#562-569) is never used and should be removed
SpaceDoge._burnFrom(address,uint256) (contracts/TestToken.sol#717-727) is never used and should be removed
SpaceDoge._mint(address,uint256) (contracts/TestToken.sol#690-696) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.17 (contracts/TestToken.sol#12) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
Pragma version^0.8.17 (contracts/TestToken.sol#339) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
SolidC-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Function IPancakeSwapPair.DOMAIN_SEPARATOR() (contracts/TestToken.sol#74) is not in mixedCase
Function IPancakeSwapPair.PERMIT_TYPEHASH() (contracts/TestToken.sol#76) is not in mixedCase
Function IPancakeSwapPair.MINIMUM_LIQUIDITY() (contracts/TestToken.sol#107) is not in mixedCase
Function IPancakeSwapRouter.WETH() (contracts/TestToken.sol#147) is not in mixedCase
Parameter SpaceDoge.setInitialAddresses(address).RouterAddress (contracts/TestToken.sol#808) is not in mixedCase
Parameter SpaceDoge.setFeeAddresses(address,address,address).rewardAddress (contracts/TestToken.sol#818) is not in mixedCase
Parameter SpaceDoge.setFeeAddresses(address,address,address).operationAddress (contracts/TestToken.sol#819) is not in mixedCase
Parameter SpaceDoge.setFeeAddresses(address,address,address).poolAddress (contracts/TestToken.sol#820) is not in mixedCase
Parameter SpaceDoge.setbuyFee(uint256,uint256,uint256,uint256).rewardFee (contracts/TestToken.sol#832) is not in mixedCase
Parameter SpaceDoge.setbuyFee(uint256,uint256,uint256,uint256).operationFee (contracts/TestToken.sol#833) is not in mixedCase
Parameter SpaceDoge.setbuyFee(uint256,uint256,uint256,uint256).liquidityFee (contracts/TestToken.sol#834) is not in mixedCase
Parameter SpaceDoge.setbuyFee(uint256,uint256,uint256,uint256).poolFee (contracts/TestToken.sol#835) is not in mixedCase
Parameter SpaceDoge.setsellFee(uint256,uint256,uint256,uint256).rewardFee (contracts/TestToken.sol#845) is not in mixedCase
Parameter SpaceDoge.setsellFee(uint256,uint256,uint256,uint256).operationFee (contracts/TestToken.sol#846) is not in mixedCase
Parameter SpaceDoge.setsellFee(uint256,uint256,uint256,uint256).liquidityFee (contracts/TestToken.sol#847) is not in mixedCase
Parameter SpaceDoge.setsellFee(uint256,uint256,uint256,uint256).poolFee (contracts/TestToken.sol#848) is not in mixedCase
Parameter SpaceDoge.excludeAddressFromFee(address,bool).isExclude (contracts/TestToken.sol#876) is not in mixedCase
Variable SpaceDoge.PancakeSwapRouter (contracts/TestToken.sol#751) is not in mixedCase
Variable SpaceDoge.PancakeSwapPair (contracts/TestToken.sol#752) is not in mixedCase
Variable SpaceDoge.maxTxAmount (contracts/TestToken.sol#770) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/TestToken.sol#352)" inContext (contracts/TestToken.sol#342-355)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable IPancakeSwapRouter.addLiquidity(address,address,uint256,uint256,uint256,address,uint256).amountADesired (contracts/TestToken.sol#152) is too similar to IPancakeSwapRouter.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (contracts/TestToken.sol#153)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

SpaceDoge.burnAddress (contracts/TestToken.sol#748) should be constant
SpaceDoge.numTokensSellToAddToLiquidity (contracts/TestToken.sol#771) should be constant
SpaceDoge.swapAndLiquifyEnabled (contracts/TestToken.sol#761) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

SpaceDoge._decimals (contracts/TestToken.sol#594) should be immutable
SpaceDoge._name (contracts/TestToken.sol#596) should be immutable
SpaceDoge._symbol (contracts/TestToken.sol#595) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```



# Summary

- Owner is able to set taxes up to 100%
- Owner is able to set max buy/sell/transfer amount to 0
- Owner is able to disable trades by setting max tax to 0
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to mint new tokens

# MANUAL AUDIT

## Severity Criteria

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

Vulnerabilities are divided into three primary risk categories: **high**, **medium**, and **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			



# FINDINGS

- **High Risk Findings:2**
- **Medium Risk Findings:1**
- **Low Risk Findings:0**
- **Suggestions & discussion: 1**

## High Risk Issues:

**Centralization – Owner is able to set buy and sell taxes up to 100%**

```
function setbuyFee(
    uint256 _rewardFee,
    uint256 _operationFee,
    uint256 _liquidityFee,
    uint256 _poolfee
) external onlyOwner {
    // @audit no limitations
    buyFees.reward = _rewardFee;
    buyFees.operation = _operationFee;
    buyFees.liquidity = _liquidityFee;
    buyFees.poolfee = _poolfee;
    emit SetBuyFee(buyFees);
}

ftrace | funcSig
function setsellFee(
    uint256 _rewardFee,
    uint256 _operationFee,
    uint256 _liquidityFee,
    uint256 _poolfee
) external onlyOwner {
    // @audit no limitations
    sellFees.reward = _rewardFee;
    sellFees.operation = _operationFee;
    sellFees.liquidity = _liquidityFee;
    sellFees.poolfee = _poolfee;
    emit SetSellFee(sellFees);
}
```

**Suggestion: set a limit for max buy/sell tax**

**Centralization – Owner is able to set max buy/sell/transfer amounts to 0**

```
function setMaxTxAmount(uint maxTxAmount) external onlyOwner {  
    //@audit no limitations for maxTxAmount  
    maxTxAmount = maxTxAmount;  
}
```

**Suggestion: set a limit for max tax**



## Medium :

**Centralization – Auto-liquidity generated tokens are sent to owner's wallet (EOA wallet)**

```
function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
    _approve(address(this), address(PancakeSwapRouter), tokenAmount);

    PancakeSwapRouter.addLiquidityETH{value: ethAmount}(
        address(this),
        tokenAmount,
        0, // slippage is unavoidable
        0, // slippage is unavoidable
        owner(),
        block.timestamp
    );
}
```

**Suggestion : burn auto-liquidity generated tokens.**

## Suggestion:

**Use latest version of solidity compiler**



## 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](http://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 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 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. 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 Expelee team disclaims any liability for the resulting losses.