AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

Arb Inu

DATED : 02 APR 23'

AUDIT SUMMARY

Project name – Arb Inu

Date: 02 April, 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	1	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	1	0	0	0

USED TOOLS

Tools:

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

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

3- Slither : Static Analysis

Testnet Link: Contract has been tested on binance smart chain testnet which can be found in below link:

https://testnet.bscscan.com/token/0x3482130B967
8558C422CFb648957730e56DE015D#writeContract

Token Information

Token Name : Arb Inu

Token Symbol: AINU

Decimals: 9

Token Supply: 1,000,000,000

Token Address:
0x8E0D9E03D4baE85EFDaa7d49e5c7BBA79eB4c65F

Checksum:
1ee6badd183eb82feada954622de2b8f6fa9fb27

Owner:
0x442d23e0d5ba876cad2d15d858ffFe4525738f9B
(at time of audit)

TOKEN OVERVIEW

Fees:

Buy Fees: upto 12%

Sell Fees: upto 12%

Transfer Fees: 0%

Fees Privilige: owner

Ownership : owner

Minting: No mint function

Max Tx Amount/ Max Wallet Amount: No

Blacklist: No

Other Priviliges: exlcuding from rewards - including in rewards - changing internal swap threshold - including in fees - excluding from 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

◆ **Critical**

◆ **High-Risk**

◆ **Medium-Risk**

◆ **Low-Risk**

◆ **Gas Optimization /Suggestion**

Description

These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.

A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.

A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.

A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.

A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
◆ **Critical**	0
◆ **High-Risk**	1 (**Resolved**)
◆ **Medium-Risk**	0
◆ **Low-Risk**	0
◆ **Gas Optimization / Suggestions**	0

INHERITANCE TREE
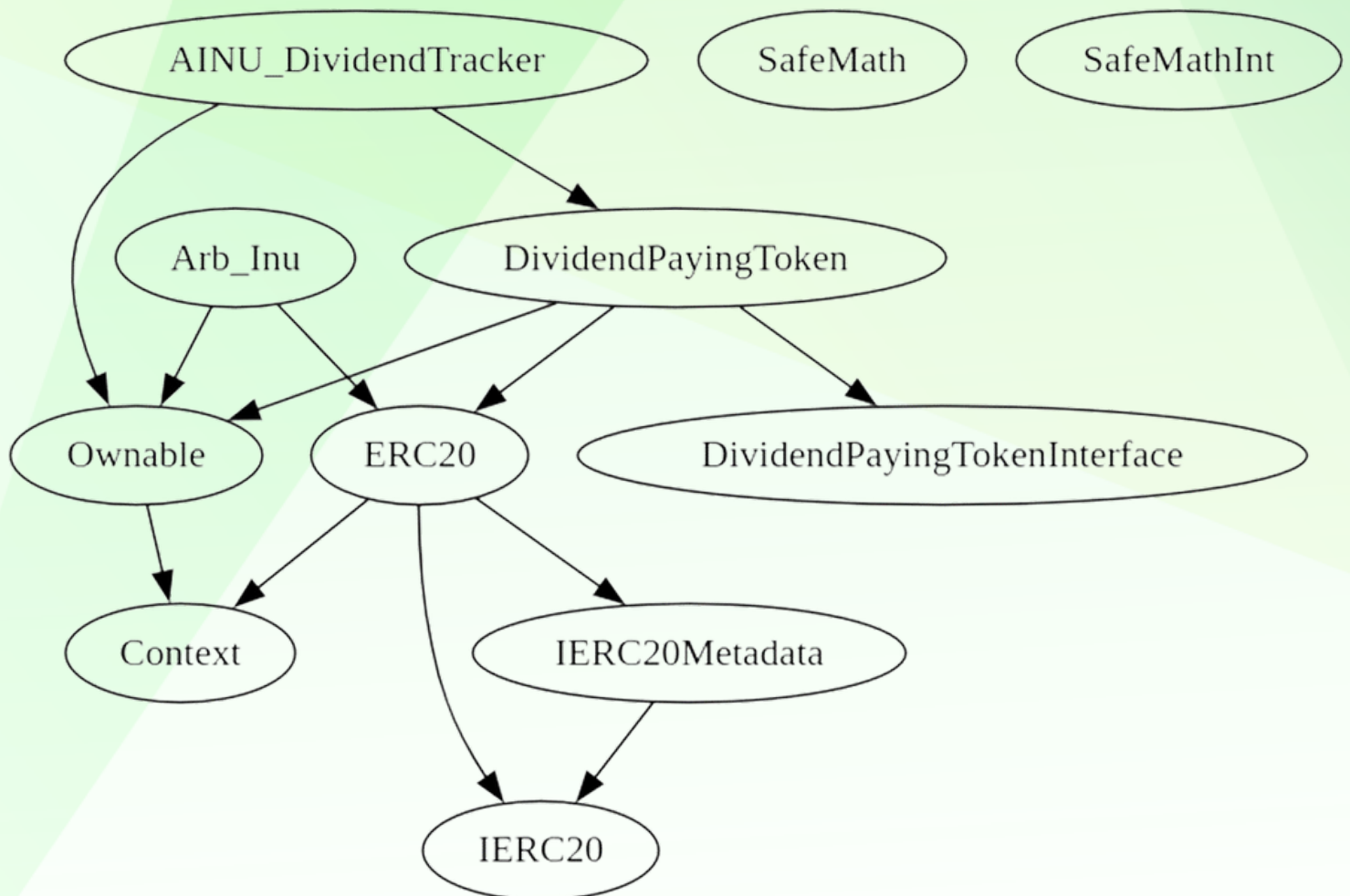
POINTS TO NOTE

- Owner is not able to set buy/sell fees more than 12% (24% max fee)
- 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 not 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**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
IERC20	Interface			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<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❗
└	increaseAllowance	Public ❗	🛑	NO❗
└	decreaseAllowance	Public ❗	🛑	NO❗
└	_transfer	Internal 🔒	🛑	
└	_tokengeneration	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
SafeMath	Library			
└	add	Internal 🔒		

CONTRACT ASSESMENT

└	sub	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	mod	Internal 🔒		
SafeMathInt	Library			
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
└	abs	Internal 🔒		
└	toUint256Safe	Internal 🔒		
SafeMathUint	Library			
└	toInt256Safe	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
IPair	Interface			
└	sync	External ❗	🛑	NO❗
IFactory	Interface			
└	createPair	External ❗	🛑	NO❗
└	getPair	External ❗		NO❗
IRouter	Interface			
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactETHForTokens	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
DividendPayingTokenInterface	Interface			
└	dividendOf	External ❗		NO❗

CONTRACT ASSESMENT

	└	distributeDividends	External ❗		💵	NO❗
	└	withdrawableDividendOf	External ❗			NO❗
	└	withdrawnDividendOf	External ❗			NO❗
	└	accumulativeDividendOf	External ❗			NO❗
DividendPayingToken	Implementation	ERC20, DividendPayingTokenInterface, Ownable		
	└	<Constructor>	Public ❗	🛑	ERC20
	└	<Receive Ether>	External ❗	💵	NO❗
	└	distributeDividends	Public ❗	💵	NO❗
	└	_withdrawDividendOfUser	Internal 🔒	🛑	
	└	setRewardToken	External ❗	🛑	onlyOwner
	└	swapBnbForCustomToken	Internal 🔒	🛑	
	└	dividendOf	Public ❗		NO❗
	└	withdrawableDividendOf	Public ❗		NO❗
	└	withdrawnDividendOf	Public ❗		NO❗
	└	accumulativeDividendOf	Public ❗		NO❗
	└	_transfer	Internal 🔒	🛑	
	└	_tokengeneration	Internal 🔒	🛑	
	└	_burn	Internal 🔒	🛑	
	└	_setBalance	Internal 🔒	🛑	
IterableMapping	Library			
	└	get	Internal 🔒		
	└	getIndexOfKey	Internal 🔒		
	└	getKeyAtIndex	Internal 🔒		
	└	size	Internal 🔒		
	└	set	Internal 🔒	🛑	
	└	remove	Internal 🔒	🛑	
Address	Library			
	└	sendValue	Internal 🔒	🛑	
Arb_Inu	Implementation	ERC20, Ownable		
	└	<Constructor>	Public ❗	🛑	ERC20
	└	<Receive Ether>	External ❗	💵	NO❗
	└	updateDividendTracker	Public ❗	🛑	onlyOwner
	└	processDividendTracker	External ❗	🛑	NO❗
	└	claim	External ❗	🛑	NO❗
	└	rescueBEP20Tokens	External ❗	🛑	onlyOwner
	└	forceSend	External ❗	🛑	NO❗
	└	excludeFromFees	Public ❗	🛑	onlyOwner
	└	excludeMultipleAccountsFromFees	Public ❗	🛑	onlyOwner

CONTRACT ASSESMENT

└	excludeFromDividends	External ❗	🛑	onlyOwner
└	setMarketingWallet	External ❗	🛑	onlyOwner
└	setSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	setBuyTaxes	External ❗	🛑	onlyOwner
└	setSellTaxes	External ❗	🛑	onlyOwner
└	setSwapEnabled	External ❗	🛑	onlyOwner
└	enableTradingEnabled	External ❗	🛑	onlyOwner
└	setAntiBotBlocks	External ❗	🛑	onlyOwner
└	setMinBalanceForDividends	External ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	setGasForProcessing	External ❗	🛑	onlyOwner
└	setClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO❗
└	getTotalDividendsDistributed	External ❗		NO❗
└	isExcludedFromFees	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	getCurrentRewardToken	External ❗		NO❗
└	dividendTokenBalanceOf	Public ❗		NO❗
└	getAccountDividendsInfo	External ❗		NO❗
└	getAccountDividendsInfoAtIndex	External ❗		NO❗
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfDividendTokenHolders	External ❗		NO❗
└	_transfer	Internal 🔒	🛑	
└	swapAndLiquify	Private 🔐	🛑	
└	swapTokensForBNB	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
AINU_DividendTracker	Implementation	Ownable, DividendPayingToken		
└	<Constructor>	Public ❗	🛑	DividendPayingToken
└	_transfer	Internal 🔒		
└	setMinBalanceForDividends	External ❗	🛑	onlyOwner
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfTokenHolders	External ❗		NO❗
└	getCurrentRewardToken	External ❗		NO❗
└	getAccount	Public ❗		NO❗
└	getAccountAtIndex	Public ❗		NO❗
└	canAutoClaim	Private 🔐		
└	setBalance	Public ❗	🛑	onlyOwner
└	process	Public ❗	🛑	|NO❗

CONTRACT ASSESMENT

| └ | processAccount | Public ❗ | 🛑 | onlyOwner |

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS

```
Function IRouter.WETH() (contracts/Token.sol#740) is not in mixedCase
Parameter DividendPayingToken.dividendOf(address)._owner (contracts/Token.sol#956) is not in mixedCase
Parameter DividendPayingToken.withdrawableDividendOf(address)._owner (contracts/Token.sol#964) is not in mixedCase
Parameter DividendPayingToken.withdrawnDividendOf(address)._owner (contracts/Token.sol#973) is not in mixedCase
Parameter DividendPayingToken.accumulativeDividendOf(address)._owner (contracts/Token.sol#984) is not in mixedCase
Constant DividendPayingToken.magnitude (contracts/Token.sol#830) is not in UPPER_CASE_WITH_UNDERSCORES
Contract Easter_Doge (contracts/Token.sol#1135-1644) is not in CapWords
Parameter Easter_Doge.setBuyTaxes(uint256,uint256,uint256)._rewards (contracts/Token.sol#1328) is not in mixedCase
Parameter Easter_Doge.setBuyTaxes(uint256,uint256,uint256)._marketing (contracts/Token.sol#1329) is not in mixedCase
Parameter Easter_Doge.setBuyTaxes(uint256,uint256,uint256)._liquidity (contracts/Token.sol#1330) is not in mixedCase
Parameter Easter_Doge.setSellTaxes(uint256,uint256,uint256)._rewards (contracts/Token.sol#1340) is not in mixedCase
Parameter Easter_Doge.setSellTaxes(uint256,uint256,uint256)._marketing (contracts/Token.sol#1341) is not in mixedCase
Parameter Easter_Doge.setSellTaxes(uint256,uint256,uint256)._liquidity (contracts/Token.sol#1342) is not in mixedCase
Parameter Easter_Doge.setSwapEnabled(bool)._enabled (contracts/Token.sol#1351) is not in mixedCase
Constant Easter_Doge.deadWallet (contracts/Token.sol#1148-1149) is not in UPPER_CASE_WITH_UNDERSCORES
Variable Easter_Doge._isExcludedFromFees (contracts/Token.sol#1172) is not in mixedCase
Contract EDoge_DividendTracker (contracts/Token.sol#1646-1896) is not in CapWords
Parameter EDoge_DividendTracker.getAccount(address)._account (contracts/Token.sol#1727) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/Token.sol#15)" inContext (contracts/Token.sol#9-18)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

Variable DividendPayingToken._withdrawDividendOfUser(address)._withdrawableDividend (contracts/Token.sol#887) is too similar to EDoge_DividendTracker.getAccount(address).withdrawableDividends (co
ntracts/Token.sol#1735)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-too-similar

Easter_Doge.setGasForProcessing(uint256) (contracts/Token.sol#1388-1399) uses literals with too many digits:
    - require(bool,string)(newValue >= 200000 && newValue <= 500000,GasForProcessing must be between 200,000 and 500,000) (contracts/Token.sol#1389-1392)
Easter_Doge.slitherConstructorVariables() (contracts/Token.sol#1135-1644) uses literals with too many digits:
    - gasForProcessing = 300000 (contracts/Token.sol#1166)
EDoge_DividendTracker.constructor() (contracts/Token.sol#1670-1675) uses literals with too many digits:
    - minimumTokenBalanceForDividends = 10000000 * (10 ** decimals()) (contracts/Token.sol#1674)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits

SafeMathInt.MAX_INT256 (contracts/Token.sol#594) is never used in SafeMathInt (contracts/Token.sol#592-649)
Easter_Doge.currentRewardToken (contracts/Token.sol#1154) is never used in Easter_Doge (contracts/Token.sol#1135-1644)
Easter_Doge.lastSell (contracts/Token.sol#1168) is never used in Easter_Doge (contracts/Token.sol#1135-1644)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variable

Easter_Doge.currentRewardToken (contracts/Token.sol#1154) should be constant
Easter_Doge.launchtax (contracts/Token.sol#1170) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

DividendPayingToken.router (contracts/Token.sol#832) should be immutable
Easter_Doge.pair (contracts/Token.sol#1139) should be immutable
Easter_Doge.router (contracts/Token.sol#1138) 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 issues found

FUNCTIONAL TESTING

Router (PCS V2):
0xD99D1c33F9fC3444f8101754aBC46c52416550D1

1- Adding Liquidity (Passed):
liquidity added on Pancakeswap V2:

https://testnet.bscscan.com/tx/0x0d0eb5dd3a51c8dd458ca0dfc34eb42b23e10c4b6994b4e08ba1fb99ab8448c9

2- Buying when trading not enabled (0%)(Passed):

https://testnet.bscscan.com/tx/0x4f931b2167f683d61e19bb0796a943940929965edf7b20254b53c099d566e534

3- Selling when trading not enabled (0%)(Passed):

https://testnet.bscscan.com/tx/0xcda898ccfb925d354c5bc9e644327489b4287ebcacda553fd11d83d9588db8de

4- Transferring when trading not enabled (0% tax) (passed):

https://testnet.bscscan.com/tx/0xa0e46ce4514e5b86d0ac8b2e2e33aa8647536a6b909db8b32c74ee8195792b76

5- Buying when trading enabled (upto 12% tax) (passed):

https://testnet.bscscan.com/tx/0xf68d601cf9868406bd12caad81d141b544bd309476f0241fa687cc70cdbb166d

FUNCTIONAL TESTING

6- Selling when trading enabled (upto 12% tax) (passed):

https://testnet.bscscan.com/tx/0xffdd74eda245215f95884e8f4c8
3c23e341d747c296ca6fddd56fb388b6029d8

7- Transferring when trading enabled (0% tax) (passed):

https://testnet.bscscan.com/tx/0x1a893f0f9314f7340f637584f6a
c7d5c14198d66dceb876ce75fd68568231ec8

8- Internal swap (passed):
As can seen in this transaction, marketing wallet received BNB
https://testnet.bscscan.com/address/0xbf38df11ce20160a1f9421d1
649dbf26a6155d5c#internaltx

9- Auto Liquidity (passed):
Auto liquidity generated tokens are burnt, as can be seen in this
transaction
https://testnet.bscscan.com/token/0xe67b7d7feb9032dd21cb274
552e0d7ee57aab24c?
a=0x0000000000000000000000000000000000000000

10- Distribution of rewards (passed):
BUSD tokens are distributed between holders, this can be seen in
this transaction
https://testnet.bscscan.com/tx/0x1a893f0f9314f7340f637584f6a
c7d5c14198d66dceb876ce75fd68568231ec8

MANUAL TESTING

Centralization - Owner must enable trading

Severity: High
Function: enableTradingEnabled
Lines: 1164
Status: 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.

```solidity
function enableTradingEnabled() external onlyOwner {
    require(!tradingEnabled, "Trading is already enabled");
    tradingEnabled = true;
    startTradingBlock = block.number;
}
```

Recommendation:
Incorporate a safety mechanism that allows investors to activate trading if a specified duration has elapsed since the conclusion of the presale.
Alleviation:
Since contract is owned by safu dev, enabling trades is guaranteed.

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