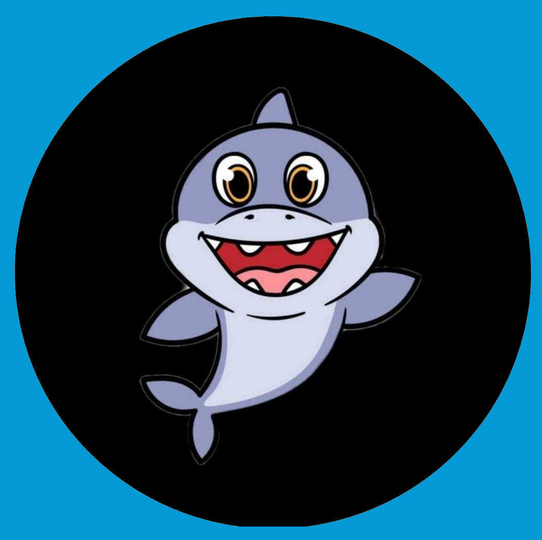
QuillAudits

Audit Report
July, 2021



Contents

Scope of Audit

The scope of this audit was to analyze and document the Family Shark Safe smart contract codebase for quality, security, and correctness.

Checked Vulnerabilities

We have scanned the smart contract for commonly known and more specific vulnerabilities. Here are some of the commonly known vulnerabilities that we considered:

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contract, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

Structural Analysis
In this step we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.
SmartCheck.

Static Analysis
Static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step a series of automated tools are used to test security of smart contracts.

Code Review / Manual Analysis
Manual Analysis or review of code was done to identify new vulnerability or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of automated analysis were manually verified.

Gas Consumption
In this step we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit
Remix IDE, Truffle, Truffle Team, Ganache, Solhint, Mythril, Slither, SmartCheck.

Issue Categories

Every issue in this report has been assigned with a severity level. There are four levels of severity and each of them has been explained below.

High severity issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality and we recommend these issues to be fixed before moving to a live environment.

Medium level severity issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems and they should still be fixed.

Low level severity issues

Low level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are severity four issues which indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Number of issues per severity

Type	High	Medium	Low	Informational
Open	0	0	3	5
Acknowledged	0	0	0	0
Closed	0	0	0	0

Introduction

During the period of **July 14, 2021 to July 19, 2021** - QuillAudits Team performed a security audit for Family Shark Safe smart contracts.

The code for the audit was provided by Family Shark Safe, and their hashes are mentioned below:

File: CoinToken.sol
Platform: BSC / Solidity
Smart Contract Online Code: https://bscscan.com/address/0x52Be45401De7fb88F9f8Fb6F336252009E55cE59#code
MD5 hash: 155F1335B04B9C28A7E4E9934780B750

Issues Found – Code Review / Manual Testing

High severity issues

No issues were found

Medium severity issues

No issues were found

Low level severity issues

1. Possible to gain ownership

Line	Code

217	function renounceOwnership() public virtual onlyOwner {
	emit OwnershipTransferred(_owner, address(0));
	_owner = address(0);
	}

Description

Possible to gain ownership after renouncing the contract ownership. The owner can renounce ownership and make contract without owner, but they can regain ownership by following the steps below:
1. Owner calls the lock function in contract to set the current owner as _previousOwner.
2. Owner calls unlock to unlock the contract and set _owner = _previousOwner.
3. Owner called renounceOwnership to leave the contract without the owner.
4. Owner calls unlock to regain ownership.

Remediation

We suggest removing these lock/unlock functions as this seems not to serve a great purpose. Otherwise, always renounce ownership first before calling the lock function.

Status: Open

2. Infinite loop

Line	Code

612	function includeInReward(address account) external onlyOwner() {
	require(_isExcluded[account], "Account is already included");
	for (uint256 i = 0; i < _excluded.length; i++) {
	if (_excluded[i] == account) {
	_excluded[i] = _excluded[_excluded.length - 1];
	_tOwned[account] = 0;
	_isExcluded[account] = false;
	_excluded.pop();
	break;
	}
	}
	}

Description

In includeInReward & _getCurrentSupply functions for loop do not have _excluded length limit , which costs more gas

Remediation

_exclude length should be limited.

Status: Open

3. Centralized risk in addLiquidity

Line	Code

841	function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
	_approve(address(this), address(uniswapV2Router), tokenAmount);
	uniswapV2Router.addLiquidityETH{value: ethAmount}(
	address(this),
	tokenAmount,
	0, // slippage is unavoidable
	0, // slippage is unavoidable
	owner(),
	block.timestamp
	);
	}

Description
In addLiquidity function, owner() gets Tokens from the Pool. At some time, The owner will accumulate significant tokens. If the _owner is an EOA (Externally Owned Account), mishandling of its private key can have devastating consequences to the project.

Remediation
addLiquidityETH function call to be replaced by address(this)receive and to restrict the management of the tokens. This will protect the tokens From being stolen if the _owner account is compromised.

Status: Open

Informational

4. Use the latest solidity version

Line	Code
7	pragma solidity ^0.8.4;

Description
Using the latest solidity will prevent any compiler-level bugs.

Remediation
Use 0.8.6, which is the latest version at the time of this audit.

Status: Open

5. Spelling mistakes

Line	Code
670	//to recieve ETH from uniswapV2Router when swaping receive() external payable {}

Description
Typing mistakes in comments.

Remediation
Please correct spellings

Status: Open

6. external instead of public

Line	Code

577	function deliver(uint256 tAmount) public {
	address sender = _msgSender();
	require(!_isExcluded[sender], "Excluded addresses cannot call this function");
	(uint256 rAmount,,,,,,) = _getValues(tAmount);
	_rOwned[sender] = _rOwned[sender].sub(rAmount);
	_rTotal = _rTotal.sub(rAmount);
	_tFeeTotal = _tFeeTotal.add(tAmount);
	}

Description
If any function is not called from inside the smart contract, then it is better to declare it as external instead of public as it saves some gas as well.

Remediation
We recommend applying OpenZeppelin ReentrancyGuard library - nonReentrant modifier for the aforementioned functions to prevent reentrancy attacks.

Status: Open

7. Missing zero address validation

Description
Variable validation is not performed in below functions :

- deliver
- swapAndLiquify
- setRouterAddress

Remediation
We suggest validating variables before the execution of the task.

Status: Open

8. Critical operation lacks event log

Description

List of Missing event log :

- deliver
- _takeDev
- _takeLiquidity
- addLiquidity
- removeAllFee
- restoreAllFee

Remediation

Please write an event log for listed events.

Status: Open

Functional test

Function Names	Testing results
lockTheSwap	Passed
name	Passed
symbol	Passed
decimals	Passed
totalSupply	Passed
balanceOf	Passed
transfer	Passed
allowance	Passed
approve	Passed
transferFrom	Passed
increaseAllowance	Passed
decreaseAllowance	Passed
isExcludedFromReward	Passed
totalFees	Passed
deliver	External instead of public, Critical operation lacks event log
reflectionFromToken	Passed
tokenFromReflection	Passed
excludeFromReward	External instead of public
includeInReward	Infinite loop
_transferBothExcluded	Passed

Function Names	Testing results
excludeFromFee	External instead of public
includeInFee	external instead of public
setTaxFeePercent	Passed
setDevFeePercent	Passed
setLiquidityFeePercent	Passed
setMaxTxPercent	External instead of public
setDevWalletAddress	External instead of public
setSwapAndLiquifyEnabled	External instead of public
receive	Passed
_reflectFee	Passed
_getValues	Passed
_getTValues	Passed
_getRValues	Passed
_getRate	Passed
_getCurrentSupply	Infinite loop
_takeLiquidity	Critical operation lacks event log
_takeDev	Critical operation lacks event log
calculateTaxFee	Passed
calculateDevFee	Passed
calculateLiquidityFee	Passed
removeAllFee	Critical operation lacks event log

Function Names	Testing results
addLiquidity	Centralized risk in addLiquidity
restoreAllFee	Critical operation lacks event log
isExcludedFromFee	Passed
_approve	Passed
_transfer	Passed
swapAndLiquify	Function input parameters lack of check
swapTokensForEth	Passed
_tokenTransfer	Passed
_transferStandard	Passed
_transferToExcluded	Passed
_transferFromExcluded	Passed
setRouterAddress	Function input parameters lack of check
setNumTokensSellToAddToLiquidity	Passed
owner	Passed
onlyOwner	Passed
renounceOwnership	Possible to gain ownership
transferOwnership	Passed
lock	Possible to gain ownership
unlock	Possible to gain ownership
_msgSender	Passed
_msgData	Passed

Automated Testing

Slither

```
INFO:Detectors:
CoinToken.addLiquidity(uint256,uint256) (CoinToken.sol#833-843) sends eth to arbitrary user
        Dangerous calls:
        - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol#835-842
)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations
INFO:Detectors:
Reentrancy in CoinToken._transfer(address,address,uint256) (CoinToken.sol#772-807):
        External calls:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#798)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#824-830)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#798)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (CoinToken.sol#713)
                - _rOwned[_devWalletAddress] = _rOwned[_devWalletAddress].add(rDev) (CoinToken.sol#721)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#867)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#877)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#868)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#889)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#619)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#890)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#879)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#621)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _rTotal = _rTotal.sub(rFee) (CoinToken.sol#666)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (CoinToken.sol#715)
                - _tOwned[_devWalletAddress] = _tOwned[_devWalletAddress].add(tDev) (CoinToken.sol#723)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (CoinToken.sol#618)
```

```
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#889)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (CoinToken.sol#619)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#890)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#879)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (CoinToken.sol#621)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _rTotal = _rTotal.sub(rFee) (CoinToken.sol#666)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (CoinToken.sol#715)
                - _tOwned[_devWalletAddress] = _tOwned[_devWalletAddress].add(tDev) (CoinToken.sol#723)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (CoinToken.sol#618)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (CoinToken.sol#888)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (CoinToken.sol#878)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (CoinToken.sol#620)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol#472-508) perf
orms a multiplication on the result of a division:
        -_maxTxAmount = (_tTotal * 5 / 1000) * 10 ** _decimals (CoinToken.sol#485)
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol#472-508) perf
orms a multiplication on the result of a division:
        -numTokensSellToAddToLiquidity = (_tTotal * 5 / 10000) * 10 ** _decimals (CoinToken.sol#486)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#divide-before-multiply
INFO:Detectors:
CoinToken.addLiquidity(uint256,uint256) (CoinToken.sol#833-843) ignores return value by uniswapV2Router.addLiquidityETH{value: ethAmount}
(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol#835-842)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
CoinToken.allowance(address,address).owner (CoinToken.sol#536) shadows:
        - Ownable.owner() (CoinToken.sol#204-206) (function)
CoinToken._approve(address,address,uint256).owner (CoinToken.sol#764) shadows:
        - Ownable.owner() (CoinToken.sol#204-206) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address).feeaddress (CoinToken.sol#47
2) lacks a zero-check on :
                - _devWalletAddress = feeaddress (CoinToken.sol#487)
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address).tokenOwner (CoinToken.sol#47
```

```
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address).feeaddress (CoinToken.sol#47
2) lacks a zero-check on :
                - _devWalletAddress = feeaddress (CoinToken.sol#487)
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address).tokenOwner (CoinToken.sol#47
2) lacks a zero-check on :
                - _owner = tokenOwner (CoinToken.sol#503)
CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address).service (CoinToken.sol#472)
lacks a zero-check on :
                - address(service).transfer(msg.value) (CoinToken.sol#504)
CoinToken.setDevWalletAddress(address)._addr (CoinToken.sol#652) lacks a zero-check on :
                - _devWalletAddress = _addr (CoinToken.sol#653)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in CoinToken._transfer(address,address,uint256) (CoinToken.sol#772-807):
        External calls:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#798)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#824-830)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#798)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _devFee = _previousDevFee (CoinToken.sol#756)
                - _devFee = 0 (CoinToken.sol#750)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _liquidityFee = _previousLiquidityFee (CoinToken.sol#757)
                - _liquidityFee = 0 (CoinToken.sol#751)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _previousDevFee = _devFee (CoinToken.sol#746)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _previousLiquidityFee = _liquidityFee (CoinToken.sol#747)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _previousTaxFee = _taxFee (CoinToken.sol#745)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
```

```
                - _liquidityFee = _previousLiquidityFee (CoinToken.sol#757)
                - _liquidityFee = 0 (CoinToken.sol#751)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _previousDevFee = _devFee (CoinToken.sol#746)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _previousLiquidityFee = _liquidityFee (CoinToken.sol#747)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _previousTaxFee = _taxFee (CoinToken.sol#745)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _tFeeTotal = _tFeeTotal.add(tFee) (CoinToken.sol#667)
        - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
                - _taxFee = _previousTaxFee (CoinToken.sol#755)
                - _taxFee = 0 (CoinToken.sol#749)
Reentrancy in CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol
#472-508):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (CoinToken.sol#
493-494)
        State variables written after the call(s):
        - _isExcludedFromFee[tokenOwner] = true (CoinToken.sol#500)
        - _isExcludedFromFee[address(this)] = true (CoinToken.sol#501)
        - _owner = tokenOwner (CoinToken.sol#503)
        - uniswapV2Router = _uniswapV2Router (CoinToken.sol#497)
Reentrancy in CoinToken.setRouterAddress(address) (CoinToken.sol#898-902):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (CoinToken.sol#
900)
        State variables written after the call(s):
        - uniswapV2Router = _uniswapV2Router (CoinToken.sol#901)
Reentrancy in CoinToken.swapAndLiquify(uint256) (CoinToken.sol#809-817):
        External calls:
        - swapTokensForEth(half) (CoinToken.sol#813)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#824-830)
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#815)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        External calls sending eth:
```

```
        State variables written after the call(s):
        - uniswapV2Router = _uniswapV2Router (CoinToken.sol#901)
Reentrancy in CoinToken.swapAndLiquify(uint256) (CoinToken.sol#809-817):
        External calls:
        - swapTokensForEth(half) (CoinToken.sol#813)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#824-830)
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#815)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#815)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        State variables written after the call(s):
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#815)
                - _allowances[owner][spender] = amount (CoinToken.sol#768)
Reentrancy in CoinToken.transferFrom(address,address,uint256) (CoinToken.sol#545-549):
        External calls:
        - _transfer(sender,recipient,amount) (CoinToken.sol#546)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#824-830)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (CoinToken.sol#546)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (CoinToken
.sol#547)
                - _allowances[owner][spender] = amount (CoinToken.sol#768)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in CoinToken._transfer(address,address,uint256) (CoinToken.sol#772-807):
        External calls:
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#798)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
```

```
        - swapAndLiquify(contractTokenBalance) (CoinToken.sol#798)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        Event emitted after the call(s):
        - Transfer(sender,recipient,tTransferAmount) (CoinToken.sol#872)
                - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
        - Transfer(sender,recipient,tTransferAmount) (CoinToken.sol#883)
                - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
        - Transfer(sender,recipient,tTransferAmount) (CoinToken.sol#894)
                - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
        - Transfer(sender,recipient,tTransferAmount) (CoinToken.sol#625)
                - _tokenTransfer(from,to,amount,takeFee) (CoinToken.sol#806)
Reentrancy in CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol
#472-508):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (CoinToken.sol#
493-494)
        External calls sending eth:
        - address(service).transfer(msg.value) (CoinToken.sol#504)
        Event emitted after the call(s):
        - Transfer(address(0),tokenOwner,_tTotal) (CoinToken.sol#505)
Reentrancy in CoinToken.swapAndLiquify(uint256) (CoinToken.sol#809-817):
        External calls:
        - swapTokensForEth(half) (CoinToken.sol#813)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#824-830)
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#815)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        External calls sending eth:
        - addLiquidity(otherHalf,newBalance) (CoinToken.sol#815)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (CoinToken.sol#769)
                - addLiquidity(otherHalf,newBalance) (CoinToken.sol#815)
        - SwapAndLiquify(half,newBalance,otherHalf) (CoinToken.sol#816)
Reentrancy in CoinToken.transferFrom(address,address,uint256) (CoinToken.sol#545-549):
```

```
                - addLiquidity(otherHalf,newBalance) (CoinToken.sol#815)
        - SwapAndLiquify(half,newBalance,otherHalf) (CoinToken.sol#816)
Reentrancy in CoinToken.transferFrom(address,address,uint256) (CoinToken.sol#545-549):
        External calls:
        - _transfer(sender,recipient,amount) (CoinToken.sol#546)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (C
oinToken.sol#824-830)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (CoinToken.sol#546)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (CoinToken.sol
#835-842)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (CoinToken.sol#769)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (C
oinToken.sol#547)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Ownable.unlock() (CoinToken.sol#235-240) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(block.timestamp > _lockTime,Contract is locked.) (CoinToken.sol#237)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (CoinToken.sol#123-127) uses assembly
        - INLINE ASM (CoinToken.sol#125)
Address._verifyCallResult(bool,bytes,string) (CoinToken.sol#175-188) uses assembly
        - INLINE ASM (CoinToken.sol#180-183)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._verifyCallResult(bool,bytes,string) (CoinToken.sol#175-188) is never used and should be removed
Address.functionCall(address,bytes) (CoinToken.sol#135-137) is never used and should be removed
Address.functionCall(address,bytes,string) (CoinToken.sol#139-141) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (CoinToken.sol#143-145) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (CoinToken.sol#147-152) is never used and should be removed
Address.functionDelegateCall(address,bytes) (CoinToken.sol#165-167) is never used and should be removed
Address.functionDelegateCall(address,bytes,string) (CoinToken.sol#169-173) is never used and should be removed
Address.functionStaticCall(address,bytes) (CoinToken.sol#154-156) is never used and should be removed
```

```
Address.functionStaticCall(address,bytes,string) (CoinToken.sol#158-162) is never used and should be removed
Address.isContract(address) (CoinToken.sol#123-127) is never used and should be removed
Address.sendValue(address,uint256) (CoinToken.sol#129-133) is never used and should be removed
Context._msgData() (CoinToken.sol#114-117) is never used and should be removed
SafeMath.div(uint256,uint256,string) (CoinToken.sol#91-96) is never used and should be removed
SafeMath.mod(uint256,uint256) (CoinToken.sol#80-82) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (CoinToken.sol#98-103) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (CoinToken.sol#20-26) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (CoinToken.sol#47-52) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (CoinToken.sol#54-59) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (CoinToken.sol#35-45) is never used and should be removed
SafeMath.trySub(uint256,uint256) (CoinToken.sol#28-33) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.0 (CoinToken.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6
solc-0.8.0 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) (CoinToken.sol#129-133):
        - (success) = recipient.call{value: amount}() (CoinToken.sol#131)
Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (CoinToken.sol#147-152):
        - (success,returndata) = target.call{value: value}(data) (CoinToken.sol#150)
Low level call in Address.functionStaticCall(address,bytes,string) (CoinToken.sol#158-162):
        - (success,returndata) = target.staticcall(data) (CoinToken.sol#160)
Low level call in Address.functionDelegateCall(address,bytes,string) (CoinToken.sol#169-173):
        - (success,returndata) = target.delegatecall(data) (CoinToken.sol#171)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Variable Ownable._owner (CoinToken.sol#194) is not in mixedCase
Variable Ownable._lockTime (CoinToken.sol#196) is not in mixedCase
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (CoinToken.sol#267) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (CoinToken.sol#268) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (CoinToken.sol#282) is not in mixedCase
Function IUniswapV2Router01.WETH() (CoinToken.sol#300) is not in mixedCase
Parameter CoinToken.setDevWalletAddress(address)._addr (CoinToken.sol#652) is not in mixedCase
Parameter CoinToken.setSwapAndLiquifyEnabled(bool)._enabled (CoinToken.sol#657) is not in mixedCase
Parameter CoinToken.calculateTaxFee(uint256)._amount (CoinToken.sol#726) is not in mixedCase
Parameter CoinToken.calculateDevFee(uint256)._amount (CoinToken.sol#732) is not in mixedCase
```

```
Parameter CoinToken.setSwapAndLiquifyEnabled(bool)._enabled (CoinToken.sol#657) is not in mixedCase
Parameter CoinToken.calculateTaxFee(uint256)._amount (CoinToken.sol#726) is not in mixedCase
Parameter CoinToken.calculateDevFee(uint256)._amount (CoinToken.sol#732) is not in mixedCase
Parameter CoinToken.calculateLiquidityFee(uint256)._amount (CoinToken.sol#738) is not in mixedCase
Variable CoinToken._devWalletAddress (CoinToken.sol#442) is not in mixedCase
Variable CoinToken._taxFee (CoinToken.sol#450) is not in mixedCase
Variable CoinToken._devFee (CoinToken.sol#452) is not in mixedCase
Variable CoinToken._liquidityFee (CoinToken.sol#454) is not in mixedCase
Variable CoinToken._maxTxAmount (CoinToken.sol#460) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (CoinToken.sol#115)" inContext (CoinToken.sol#109-118)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Reentrancy in CoinToken.constructor(string,string,uint256,uint256,uint256,uint256,uint256,address,address,address,address) (CoinToken.sol
#472-508):
        External calls:
        - address(service).transfer(msg.value) (CoinToken.sol#504)
        Event emitted after the call(s):
        - Transfer(address(0),tokenOwner,_tTotal) (CoinToken.sol#505)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (CoinToken.sol#3
04) is too similar to IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (Co
inToken.sol#305)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#866) is too similar to CoinToken._getValues(
uint256).tTransferAmount (CoinToken.sol#671)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#876) is too similar to CoinToken._getValue
s(uint256).tTransferAmount (CoinToken.sol#671)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#584) is too similar to CoinToken._getTValues(uint256)
.tTransferAmount (CoinToken.sol#680)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#584) is too similar to CoinToken._transferBothExclude
d(address,address,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#876) is too similar to CoinToken._getTValu
es(uint256).tTransferAmount (CoinToken.sol#680)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#876) is too similar to CoinToken._transfer
BothExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#866) is too similar to CoinToken._transferSt
```

```
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#876) is too similar to CoinToken._getValue
s(uint256).tTransferAmount (CoinToken.sol#671)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#584) is too similar to CoinToken._getTValues(uint256)
.tTransferAmount (CoinToken.sol#680)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#584) is too similar to CoinToken._transferBothExclude
d(address,address,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#876) is too similar to CoinToken._getTValu
es(uint256).tTransferAmount (CoinToken.sol#680)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#876) is too similar to CoinToken._transfer
BothExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#866) is too similar to CoinToken._transferSt
andard(address,address,uint256).tTransferAmount (CoinToken.sol#866)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#672) is too similar to CoinToken._getTValues(uint256).tTransferAmou
nt (CoinToken.sol#680)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#672) is too similar to CoinToken._transferBothExcluded(address,addr
ess,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#672) is too similar to CoinToken._transferBothExcluded(address,addr
ess,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#887) is too similar to CoinToken._transf
erBothExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#689) is too similar to CoinToken._
transferBothExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#876) is too similar to CoinToken._transfer
ToExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#876)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#887) is too similar to CoinToken._getTVa
lues(uint256).tTransferAmount (CoinToken.sol#680)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#876) is too similar to CoinToken._transfer
Standard(address,address,uint256).tTransferAmount (CoinToken.sol#866)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#617) is too similar to CoinToken._transf
erBothExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#866) is too similar to CoinToken._getTValues
(uint256).tTransferAmount (CoinToken.sol#680)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#866) is too similar to CoinToken._transferBo
thExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#617)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#672) is too similar to CoinToken._transferToExcluded(address,addres
s,uint256).tTransferAmount (CoinToken.sol#876)
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#689) is too similar to CoinToken._
transferToExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#876)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#887) is too similar to CoinToken._transf
erStandard(address,address,uint256).tTransferAmount (CoinToken.sol#866)
```

```
transferStandard(address,address,uint256).tTransferAmount (CoinToken.sol#866)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#584) is too similar to CoinToken._transferStandard(ad
dress,address,uint256).tTransferAmount (CoinToken.sol#866)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#584) is too similar to CoinToken._transferToExcluded(
address,address,uint256).tTransferAmount (CoinToken.sol#876)
Variable CoinToken._transferStandard(address,address,uint256).rTransferAmount (CoinToken.sol#866) is too similar to CoinToken._transferTo
Excluded(address,address,uint256).tTransferAmount (CoinToken.sol#876)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#887) is too similar to CoinToken._transf
Excluded(address,address,uint256).tTransferAmount (CoinToken.sol#876)
Variable CoinToken._transferFromExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#887) is too similar to CoinToken._transf
erFromExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#887)
Variable CoinToken._getValues(uint256).rTransferAmount (CoinToken.sol#672) is too similar to CoinToken._getValues(uint256).tTransferAmoun
t (CoinToken.sol#671)
Variable CoinToken._getRValues(uint256,uint256,uint256,uint256,uint256).rTransferAmount (CoinToken.sol#689) is too similar to CoinToken._
getValues(uint256).tTransferAmount (CoinToken.sol#671)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#617) is too similar to CoinToken._transf
erToExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#876)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#617) is too similar to CoinToken._transf
erStandard(address,address,uint256).tTransferAmount (CoinToken.sol#866)
Variable CoinToken._transferToExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#876) is too similar to CoinToken._transfer
FromExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#887)
Variable CoinToken._transferBothExcluded(address,address,uint256).rTransferAmount (CoinToken.sol#617) is too similar to CoinToken._transf
erFromExcluded(address,address,uint256).tTransferAmount (CoinToken.sol#887)
Variable CoinToken.reflectionFromToken(uint256,bool).rTransferAmount (CoinToken.sol#584) is too similar to CoinToken._getValues(uint256).
tTransferAmount (CoinToken.sol#671)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (CoinToken.sol#213-216)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (CoinToken.sol#219-223)
lock(uint256) should be declared external:
        - Ownable.lock(uint256) (CoinToken.sol#227-232)
unlock() should be declared external:
        - Ownable.unlock() (CoinToken.sol#235-240)
name() should be declared external:
        - CoinToken.name() (CoinToken.sol#510-512)
symbol() should be declared external:
        - CoinToken.symbol() (CoinToken.sol#514-516)
decimals() should be declared external:
```

```
        - Ownable.lock(uint256) (CoinToken.sol#227-232)
unlock() should be declared external:
        - Ownable.unlock() (CoinToken.sol#235-240)
name() should be declared external:
        - CoinToken.name() (CoinToken.sol#510-512)
symbol() should be declared external:
        - CoinToken.symbol() (CoinToken.sol#514-516)
decimals() should be declared external:
        - CoinToken.decimals() (CoinToken.sol#518-520)
totalSupply() should be declared external:
        - CoinToken.totalSupply() (CoinToken.sol#522-524)
transfer(address,uint256) should be declared external:
        - CoinToken.transfer(address,uint256) (CoinToken.sol#531-534)
allowance(address,address) should be declared external:
        - CoinToken.allowance(address,address) (CoinToken.sol#536-538)
approve(address,uint256) should be declared external:
        - CoinToken.approve(address,uint256) (CoinToken.sol#540-543)
transferFrom(address,address,uint256) should be declared external:
        - CoinToken.transferFrom(address,address,uint256) (CoinToken.sol#545-549)
increaseAllowance(address,uint256) should be declared external:
        - CoinToken.increaseAllowance(address,uint256) (CoinToken.sol#551-554)
decreaseAllowance(address,uint256) should be declared external:
        - CoinToken.decreaseAllowance(address,uint256) (CoinToken.sol#556-559)
isExcludedFromReward(address) should be declared external:
        - CoinToken.isExcludedFromReward(address) (CoinToken.sol#561-563)
totalFees() should be declared external:
        - CoinToken.totalFees() (CoinToken.sol#565-567)
deliver(uint256) should be declared external:
        - CoinToken.deliver(uint256) (CoinToken.sol#569-576)
reflectionFromToken(uint256,bool) should be declared external:
        - CoinToken.reflectionFromToken(uint256,bool) (CoinToken.sol#578-587)
excludeFromReward(address) should be declared external:
        - CoinToken.excludeFromReward(address) (CoinToken.sol#595-602)
excludeFromFee(address) should be declared external:
        - CoinToken.excludeFromFee(address) (CoinToken.sol#628-630)
includeInFee(address) should be declared external:
        - CoinToken.includeInFee(address) (CoinToken.sol#632-634)
setMaxTxPercent(uint256) should be declared external:
```

```
        - CoinToken.includeInFee(address) (CoinToken.sol#632-634)
setMaxTxPercent(uint256) should be declared external:
        - CoinToken.setMaxTxPercent(uint256) (CoinToken.sol#648-650)
setDevWalletAddress(address) should be declared external:
        - CoinToken.setDevWalletAddress(address) (CoinToken.sol#652-654)
setSwapAndLiquifyEnabled(bool) should be declared external:
        - CoinToken.setSwapAndLiquifyEnabled(bool) (CoinToken.sol#657-660)
isExcludedFromFee(address) should be declared external:
        - CoinToken.isExcludedFromFee(address) (CoinToken.sol#760-762)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:CoinToken.sol analyzed (10 contracts with 75 detectors), 135 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
root@server:/chetan/gaza/mycontracts#
```

SOLIDITY STATIC ANALYSIS

SOLIDITY STATIC ANALYSIS

last results for:
contracts/CoinToken.sol

Security

Transaction origin:

INTERNAL ERROR in module Transaction origin: Cannot convert undefined or null to object
Pos: not available

Check-effects-interaction:

INTERNAL ERROR in module Check-effects-interaction: Cannot convert undefined or null to object
Pos: not available

Inline assembly:

INTERNAL ERROR in module Inline assembly: Cannot convert undefined or null to object
Pos: not available

Block timestamp:

INTERNAL ERROR in module Block timestamp: Cannot convert undefined or null to object
Pos: not available

Low level calls:

INTERNAL ERROR in module Low level calls: Cannot convert undefined or null to object
Pos: not available

Selfdestruct:

INTERNAL ERROR in module Selfdestruct: Cannot convert undefined or null to object
Pos: not available

Gas & Economy

This on local calls:

INTERNAL ERROR in module This on local calls: Cannot convert undefined or null to object
Pos: not available

Delete dynamic array:

INTERNAL ERROR in module Delete dynamic array: Cannot convert undefined or null to object
Pos: not available

For loop over dynamic array:

INTERNAL ERROR in module For loop over dynamic array: Cannot convert undefined or null to object
Pos: not available

Ether transfer in loop:

INTERNAL ERROR in module Ether transfer in loop: Cannot convert undefined or null to object
Pos: not available

ERC

ERC20:

INTERNAL ERROR in module ERC20: Cannot convert undefined or null to object
Pos: not available

Miscellaneous

Constant/View/Pure functions:

INTERNAL ERROR in module Constant/View/Pure functions: Cannot convert undefined or null to object
Pos: not available

Similar variable names:

INTERNAL ERROR in module Similar variable names: Cannot convert undefined or null to object
Pos: not available

No return:

INTERNAL ERROR in module No return: Cannot convert undefined or null to object
Pos: not available

Guard conditions:

INTERNAL ERROR in module Guard conditions: Cannot convert undefined or null to object
Pos: not available

String length:

INTERNAL ERROR in module String length: Cannot convert undefined or null to object
Pos: not available

SOLHINT LINTER

```
contracts/CoinToken.sol:25:18: Error: Parse error: missing ';' at '{'
contracts/CoinToken.sol:33:18: Error: Parse error: missing ';' at '{'
contracts/CoinToken.sol:40:18: Error: Parse error: missing ';' at '{'
contracts/CoinToken.sol:52:18: Error: Parse error: missing ';' at '{'
contracts/CoinToken.sol:59:18: Error: Parse error: missing ';' at '{'
contracts/CoinToken.sol:89:18: Error: Parse error: missing ';' at '{'
contracts/CoinToken.sol:96:18: Error: Parse error: missing ';' at '{'
contracts/CoinToken.sol:103:18: Error: Parse error: missing ';' at '{'
```

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of Family Shark Safe. This audit does not provide a security or correctness guarantee of the audited smart contracts. The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the Family Shark Safe Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

Closing Summary

In this report, we have considered the security of the Family Shark Safe platform. We performed our audit according to the procedure described above.

The audit showed some low and some informational severity issues. It is recommended to fix them.

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

audits@quillhash.com